Covering Letter

18-8-2026

From

Dr. A Pandian,

Professor and Head,

Department of Electrical & Electronics Engineering,

KLEF

To

The Principal,

KL College of Engineering.

Respected Sir,

Sub: Submission of BOS Minutes July 2026

I here by attach the XXXIX Board of Studies Minutes of Department of Electrical & Electronics Engineering held on 31-7-2026 in Hybrid mode. All the changes are mentioned .Kindly consider sir.

Thanks & Regards

HOD EEE

Dr. A. PANDIAN, SMIEEE
Professor & HOD
Department of EEE
KLEF Deemed to be University
Green Fields, Vaddeswaram,
Guntur Dt., A.P., 522 502.

Scanned with OKEN Scanner

Koneru Lakshmaiah Education Foundation
(Category -1, Deemed to be University estd. u/s. 3 of the UGC Act, 1956)
❖ Approved by AICTE ❖ ISO 21001:2018 Certified
Campus: Green Fields, Vaddeswaram - 522 302, Guntur District, Andhra Pradesh, INDIA.
Phone No. +91 8645 - 350 200; www.klef.ac.in; www.klef.edu.in; www.kluniversity.in
Admin Off: 29-36-38, Museum Road, Governorpet, Vijayawada - 520 002. Ph: +91 - 866 - 3500122, 2576129

Dt: 08-07-2026

Department of Electrical & Electronics Engineering

Minutes of the XXXIX BOS Meeting

The Department XXXIX BOS meeting is held on 31st July 2026 from 11.30AM onwards in HYBRID mode.

The following members were present:

1. Dr. A Pandian-Professor & HoD, Department of EEE, KLEF- Chairman
2. Dr. Dogga Raveendra, Assistant Professor, EED, NIT, AP-External Member.
3. Er. Anmol Agarwal, CEO, SkyLamps,.-External Member.
4. Er. B.Ramesh, Accenture, Hyderabad,.-External Member.
5. Dr. K Sarada, Professor Incharge (Academics), Associate Professor, Department of EEE, KLEF- Internal Member.
6. Dr. R Bhanu Prakash, Associate Professor, Department of EEE, KLEF- Internal Member
7. Dr.J Somlal, Professor, Department of EEE, KLEF - Internal Member
8. Dr. B Loveswara Rao, Professor, Department of EEE, KLEF - Internal Member
9. Dr. P Srinivas Varma, Associate Professor, Department of EEE, KLEF- Internal Member
10. Dr. M Kiran Kumar, Associate Professor, Department of EEE, KLEF- Internal Member
11. Dr. B Jyothi, Associate Professor, Department of EEE, KLEF- Internal Member
12. Mr. D Seshi Reddy, Associate Professor, Department of EEE, KLEF- Internal Member
13. Dr. G Raja Sekhar, Associate Professor, Department of EEE, KLEF-Internal Member
14. Dr. D Sudha, Assistant professor, Department of EEE, KLEF-Internal member
15. Dr. G Swapna, Assistant professor, Department of EEE, KLEF-Internal member
16. Dr. T Teja Sreenu, Assistant Professor, Department of EEE, KLEF-Internal Member
17. Dr. S Ravi Teja, Assistant Professor, Department of EEE KLEF- Internal Member
18. B Daniel Zacharia (2401200010), PG-Student Representative-KLEF
19. B Naveen (2401200007), PG-Student Representative, KLEF
20. Sumanthika Kancharla (2300060008), 3rd year-student representative, KLEF
21. Kasireddy Varshini (2300060019), 3rd year-student representative, KLEF

Members Absent: NIl

Agenda and Resolutions

The chairman presented the following achievements of students and faculty of the department during A.Y 2025-2026

Placements & Progression:

Placement statistics for the A.Y:2025-2026 were presented to all the members.

- BOS chairman happily shared that Y22 B.Tech (EEE) students has achieved 100% placement.
- 5 number of Y23 B,Tech students were placed in TCS Company.
- Recently K Sumanthika (2200069050) of Y23 B.Tech placed in Autobacs, Japan with 20 LPA

Research & Development

Total no. of publications including Scopus and SCI indexed by the department are 30

- 06 scholars of the department were awarded with PhD degrees in the A.Y: 2025-2026 till date.
- Dr. A Pandian, Dr. B Jyothi, and Dr. G G Raja Sekhar has received Best Researcher Awards by external organizations.
- Dr. P Srinivasa Varma has received Patent Grant.
- Dr. B Jyothi and Dr. S Ravi Teja received Seed Money from KLEF
- Electrical & Electronics Engineering Department published 33 patents in A.Y: 2025-2026.
- Electrical & Electronics Engineering Department has published 42 (SCI+WOS+ Scopus) papers and 7 conference papers.

Electrical & Electronics Engineering Department is planning to organise International Conference during November- December 2026.

Er. Anmol Agarwal external BOS member appreciated the efforts of the department for providing the balanced

AGENDA ITEM-1

To approve DAC meeting minutes held on 27-07-2026.	BoS members approved the DAC minutes and recommended it to the Academic Council.

The internal and external BOS members approved the resolutions made in Department Academic Committee (DAC)

The DAC minutes in Annexure-1a
Stake holders feedback in Annexure 1-b

AGENDA ITEM-2

To consider and approve the modified curriculum and syllabus of B.Tech-EEE Y25 admitted batch	It is resolved to approve the modified curriculum and syllabus of B.Tech-EEE Y25 admitted batch. and to recommend for further approval in Academic Council.

As per the contemporary requirements Y25 B.Tech EEE curriculum is modified by introducing courses related to Machine learning and Embedded System.

Anneure-2

AGENDA ITEM-3

To consider and approve the curriculum and syllabus of B.Tech-EEE Y26 admitted batch.	It is resolved to approve the curriculum and syllabus of B.Tech-EEE Y26 admitted batch. and to recommend for further approval in Academic Council.

Along with Semester mode Trimester mode is initiated for Y26 B.Tech-EEE admitted batch

As per the consideration of Thrust Areas Courses related to Robotics is introduced.

Annexure -3

AGENDA ITEM-4

To consider and approve the modifications of syllabus for courses in Y25 B.Tech EEE curriculum	It is resolved to approve the modification of syllabus for courses in Y25 B.Tech EEE curriculum and recommend for further approval in Academic Council.

In view of contemporary requirements BOS members approved the modification of syllabus for the following courses in Y25 B.Tech EEE curriculum

Category	Code	Course Title	% of Revision	Reason
Core	25EE2202	Electric Motors, Drives & Electromechanical Systems	30%	As per the feedback from PED Research group
Core	25EE2204	Power Electronics & Wide-Bandgap Converters	15%	As per the feedback from PED Research group
Core	25EE3105	Control Systems For Electrical & Energy Applications	18%	As per feedback from Industry Expert
Core	25EE3106	Modern Power Systems & Smart Grid Technologies	40%	As per the contemporary requirements
Core	25EE3208	Power System Protection and Control	5%	As per feedback received from PS research group.

Annexure-4 a

AGENDA ITEM-5

To consider and approve the introduction of courses in Y25 and Y26 B.Tech EEE curriculum	It is resolved to approve the the introduction of courses in Y25 and Y26 B.Tech EEE curriculum and recommend for further approval in Academic Council.

In view of contemporary requirements BOS members approved the introduction of courses in Y25 and Y26 B.Tech EEE curriculum

Y25 B.Tech-EEE

Category	Code	Course Title	Reason
Core	25SC2008	Full Stack Web Development	As per the feedback received from CSE Research group
Core	25EC2206	Embedded System Design And IOT	As per the feedback received from Academic Experts
Core	25EC2308	Embedded Software Development	As per the feedback received from Academic Experts
Core	25CS3113	Embedded AI and Tiny ML	As per the feedback received from Academic Experts
Core	25EC2309	IOT Protocols and Cloud Integration	As per the feedback received from Academic Experts
Core	25EE3207	Industrial Automation, Plc & SCADA Systems	As per the feedback received from Academic Experts
Core	25CS3112	Deep Learning With Accelerated Computing	As per the feedback received from Stakeholders.
Core	25CS2207	Generative AI and AI Platform Engineering	As per the feedback received from Stakeholders.
Elective	25EMS3111	Agentic Embedded Systems & Intelligent Edge Orchestration	As per the feedback received from Stakeholders.
Elective	25EMS3221	Electric Vehicle Systems & Battery Management Engineering	As per the feedback received from Industry Experts
Elective	25EMS3231	Advanced Power Electronics & Electric Drive Systems	As per the feedback received from Stakeholders.
Elective	25EMS3231	Industrial Automation, PLC & Energy Management Systems	As per the feedback received from Stakeholders.
Elective	25EMS 3351	Industrial Automation, PLC & Energy Management Systems	As per the feedback received from Stakeholders.
Core	25EE41 10	Renewable Energy Systems & Grid Integration	As per the feedback received from Power system Research group.
Core	25EE41	Electric Vehicle Systems &	As per the feedback received from

	09	Battery Technologies	Stakeholders

Y26 B.Tech EEE

Category	Code	Course Title	Reason
Core	26EC3112E	Foundations of Robotics & Mechatronic Systems	As per the feedback received from CSE Research group
Core	26EC3214E	Robotic Perception, Computer Vision & Sensor Fusion	As per the feedback received from Academic Experts
Core	26EC3316E	Robotic Motion Planning, Navigation & Control Systems	As per the feedback received from Academic Experts
Core	26EC4117E	Autonomous Robotic Systems & Human-Robot Interaction	As per the feedback received from Academic Experts

Annexure-4b

AGENDA ITEM-6

To consider and approve the modification of syllabus AI DRIVEN ECU DESIGN FOR AUTONOMOUS VEHICLES (PE-5) for Y23 B.Tech	It is resolve to approve the modification of syllabus AI DRIVEN ECU DESIGN FOR AUTONOMOUS VEHICLES (PE-5) for Y23 B.Tech and recommend for further approval in Academic Council.

BOS members approved the modification of syllabus for the course AI DRIVEN ECU DESIGN FOR AUTONOMOUS VEHICLES (PE-5) for Y23 B.Tech.

Annexure 4c

AGENDA ITEM-7

To consider and approve the Curriculum and Syllabus for M.Tech (PEPS) for Y26 admitted students.	It is resolve to approve the Curriculum and Syllabus for M.Tech (PEPS) for Y26 admitted students.and recommend for further approval in Academic Council.

M.Tech (PEPS) for Y26 admitted students is recommended to offer in Trimester based.

New courses

Category	Course Code and Title
Core	26PE5101E Power Electronic Converters and Modulation
Core	26PE5102E Modern Power System Analysis and State Estimation
Core	26PE5103E Modelling and Analysis of Electrical Machines
Core	26PE5104E Digital Signal Processing for Power Engineers
Core	26PE5205E Renewable Energy Systems and Integration
Core	26PE5206E AI-Enabled Smart Grid Technologies
Core	26PE5308E Power Quality and Harmonics
Core	26PE5309E Power System Protection and Substation Automation
Core	26PE5310E Power System Dynamics and Stability

Core	26PE6112E Power System Operation and AI Forecasting
Elective	26PE52A1E EV Powertrain and Battery Management Systems
Elective	26PE53A2E Smart EV Charging and IoT Energy Management
Elective	26PE61A3E Motor Drives for Electric Vehicles
Elective	26PE61A4E Power Electronics for Renewable Energy
Elective	26PE52B1E Microgrid Design and AI Energy Management
Elective	26PE53B2E Battery Energy Storage Systems
Elective	26PE61B3E Power Markets and Energy Trading
Elective	26PE61B4E Grid Integration of Renewables

Revised Courses

Category	Course Code and Title
Core	26PE5207E Electric Drives
Core	26PE6111E HVDC and FACTS

Annexure 5

AGENDA ITEM-8

To consider and approve the Value Added Courses offered in A.Y: 2026-2027 Odd Sem	It is resolve to approve the the Value Added Courses offered in A.Y: 2026-2027 Odd Sem and recommend for further approval in Academic Council.

Value Added Courses offered in Odd Semester

Sl	Certification Course
1	ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING USING PYTHON [INDUSTRIAL AUTOMATION]-L2

AGENDA ITEM-9

To consider and approve PDDs of B.Tech(EEE), M.Tech (PEPS) of Y26admitted batch and Y25 B.Tech (EEE) of Y25 admitted batch	It is resolve to approve PDDs of B.Tech(EEE), M.Tech (PEPS) of Y26admitted batch and Y25 B.Tech (EEE) of Y25 admitted batch and recommend for further approval in Academic Council.

PDDs of B.Tech(EEE), M.Tech (PEPS) of Y26admitted batch and Y25 B.Tech (EEE) of Y25 admitted batch are approved.

AGENDA ITEM-10

Any other points raised by BOS members

Er. B.Ramesh, Accenture, Hyderabad,.-BOS External Member suggested to include the course related Supply Chain Management in Business Knowledge as there is requirement in industries.

The Chairperson expressed gratitude to all members for their valuable contributions and active participation. The meeting concluded with an agreement on the next steps for implementing the approved items.

Pictures of BOS meeting proceedings:



Koneru Lakshmaiah Education Foundation

(Category -1, Deemed to be University estd. u/s. 3 of the UGC Act, 1956)

Approved by AICTE ISO 21001:2018 Certified

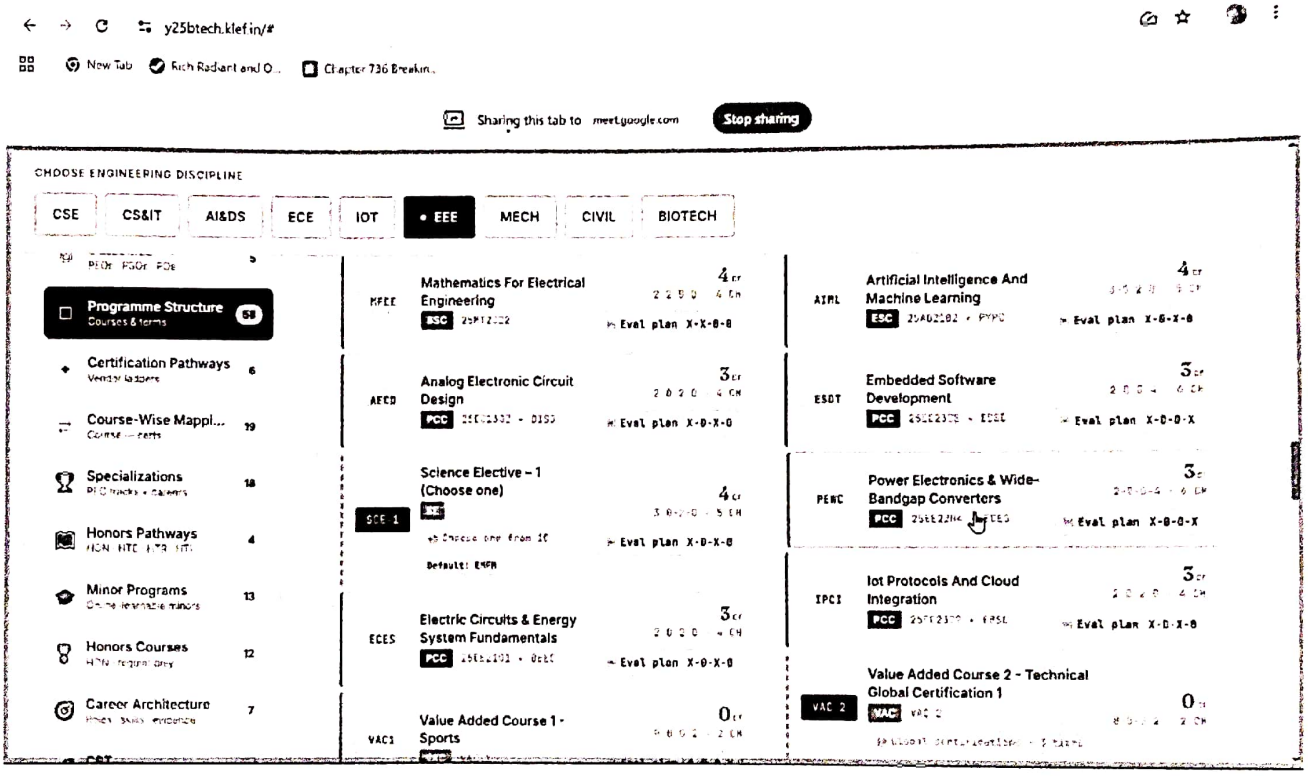
Campus: Green Fields, Vaddeswaram - 522 302, Guntur District, Andhra Pradesh, INDIA

Phone No. +91 8645 - 350 200, www.klef.ac.in, www.klef.edu.in, www.kluniversity.in

Admin Off: 29-36-38, Museum Road, Governorpet, Vijayawada - 520 002 Ph: +91 866 3500122 2576129







Dr.A. PANDIAN, SMIEEE
BOS Chairman
Professor & HOD
Department of EEE
KLEF Deemed to be University
Greeen Fields, Vaddeswaram,
Guntur Dt., A.P., 522 302.

Koneru Lakshmaiah Education Foundation
(Category -1, Deemed to be University estd. u/s. 3 of the UGC Act, 1956)
❖ Approved by AICTE ❖ ISO 21001:2018 Certified
Campus: Green Fields, Vaddeswaram - 522 302, Guntur District, Andhra Pradesh, INDIA.
Phone No. +91 8645 - 350 200; www.klef.ac.in; www.klef.edu.in; www.kluniversity.in
Admin Off: 29-36-38, Museum Road, Governorpet, Vijayawada - 520 002. Ph: +91 - 866 - 3500122, 2576129

Dt: 31-07-2026

Department of Electrical & Electronics Engineering

Minutes of the Department Advisory Committee (DAC) Meeting

The Department DAC meeting is held on 27th July2026 from 03.50PM onwards in HYBRID mode.

The following members were present:

1. Dr. A Pandian-Professor & HoD, Department of EEE, KLEF- Chairman
2. Dr.J Somlal, Professor, Department of EEE, KLEF - Internal Member
3. Dr. B Loveswara Rao, Professor, Department of EEE, KLEF - Internal Member
4. Dr. P Srinivas Varma, Associate Professor, Department of EEE, KLEF- Internal Member
5. Dr. M Kiran Kumar, Associate Professor, Department of EEE, KLEF- Internal Member
6. Dr. B Jyothi, Associate Professor, Department of EEE, KLEF- Internal Member
7. Dr. K Sarada, Associate Professor, Department of EEE, KLEF- Internal Member
8. Dr. R Bhanu Prakash, Associate Professor, Department of EEE, KLEF- Internal Member
9. Mr. D Seshi Reddy, Associate Professor, Department of EEE, KLEF- Internal Member
10. Dr. G Raja Sekhar, Associate Professor, Department of EEE, KLEF-Internal Member
11. Dr. D Sudha, Assistant professor, Department of EEE, KLEF-Internal member
12. Dr. G Swapna, Assistant professor, Department of EEE, KLEF-Internal member
13. Dr. T Teja Sreenu, Assistant Professor, Department of EEE, KLEF-Internal Member
14. Dr. S Ravi Teja, Assistant Professor, Department of EEE KLEF- Internal Member
15. B Daniel Zacharia (2401200010),PG-Student Representative, KLEF
16. B Deva Nithin (2501200001) ,PG- Student Representative, KLEF.
17. Sumanthika Kancharla (2300060008), B.Tech 3rd YEAR- Student Representative, KLEF.
18. Kasireddy Varshini (2300060019), B.Tech 3rd YEAR- Student Representative, KLEF.
19. Katta Meghana (2300060021), B.Tech 3rd YEAR- Student Representative, KLEF.

Members Absent: NIL

Agenda and Resolutions

AGENDA ITEM-1

Department Achievements for the A.Y.2025-2026	DAC chairman appreciated the efforts of faculty and the university management for their achievements done by the students.

The meeting commenced with the Chairperson presenting the minutes of the last Department Advisory Committee (DAC) meeting for review and approval. The DAC minutes included laboratory development, industry collaborations, and faculty development programs.

The chairman presented the following achievements of students and faculty of the department during A.Y 2025-2026

Placements & Progression:

Placement statistics for the A.Y:2025-2026 were presented to all the members.

Y22 B.TECH

SNO	ID NUMBER	NAME	Placed in	CTC
1	2200060002	KOTTEY MAHESH BABU	Krutanics	4.0 + 3.0 LPA
2	2200060004	PENUMATSA RAM PRASAD	Learnbuy	
3	2200060006	JONNADULA VISWANADH	Qspiders	
4	2200060009	PAPINENI SRILAXMI TRIVENI	KYB Corporation	3.36
5	2200060010	P BIKRAM KUMAR REDDY	AllDigiTech	3 LPA CTC
6	2200060012	KARNAM VISHAL	FacePrep	3 LPA CTC + RA
6	2200060012	KARNAM VISHAL	KYB Corporation	3.36
7	2200060014	SHAIK SHOAIB AHMED	Omega Healthcare	3.25 LPA
7	2200060014	SHAIK SHOAIB AHMED	Acmegrade	5 LPA
8	2200060020	YENUMULA TEJA ANKA VENKATA SUBRAHMANYAM	KYB Corporation	3.36
9	2200060021	POTLURI KAVYA SRI	KYB Corporation	3.36
10	2200060023	GOLLAPELLI GOLLAPELLI SANDEEP	Omega Healthcare	3.25 LPA
11	2200060025	PINNIBOYINA Sangeetha P	KYB Corporation	3.36
12	2200060026	Gopasani Komala	KYB Corporation	3.36
13	2200060027	Meghana Kasibhatla	TIVOLT Electric Vehicles Private Limited	
14	2200060030	VATTIKALA ASHALATHA	Hitachi Energy	6.5
15	2200069002	Moilla Sai saran Reddy	Inspireleap	4.0 + 2.5 LPA
16	2200069003	Epuri SAI SHASHANK	FacePrep	4LPA + 1LPA
16	2200069003	Epuri SAI SHASHANK	MuSigma	5LPA
17	2200069005	DACHURI NANI KISHORE	KYB Corporation	3.36
18	2200069006	SONTI VENKATA SAI VIJAY UMA MAHESH	AllDigiTech	3 LPA CTC

19	2200069007	SHAIK SAYEEF PASHA	LTI Mindtree	4.05
20	2200069009	MANEENDRA JAGARAPU	LTI Mindtree	4.05
20	2200069009	MANEENDRA JAGARAPU	Kia	5.8
21	2200069010	DEVENDER BORA	AllDigiTech	3 LPA CTC
22	2200069013	THORLAPATI VIJAY KUMAR	Omega Healthcare	3.25 LPA
23	2200069014	MAJETI NAGA NAVEEN KUMAR	AllDigiTech	3 LPA CTC
24	2200069015	MATTIREDDI GARI NEHA	AllDigiTech	3 LPA CTC
25	2200069016	VEJENDLA JOHN ALPHAUNES	Pursuit Future Technologies	
26	2200069017	Dudekula Yusuf	KYB Corporation	3.36
27	2200069019	PATAN SALAR KHAN	AllDigiTech	3 LPA CTC
28	2200069020	Galinki Nalini	Omega Healthcare	3.25 LPA
29	2200069023	KOPPULA GURU CHARAN ROYAL	KYB Corporation	3.36
30	2200069026	UMA SHANKAR TIPPANA	Aisan Industries Co. Ltd. Japan	43.83
31	2200069027	MANTADA DWARAKA VASU	KYB Corporation	3.36
32	2200069028	VANAPALLI AJAY	AllDigiTech	3 LPA CTC
33	2200069030	NARRA PRABHAT REDDY	SprintM Technologies	
34	2200069031	MUNAGAPATI KARTHIK	Inspireleap	4.0 + 2.5 LPA
35	2200069032	KRISHNA GAMPALA	AllDigiTech	3 LPA CTC
36	2200069033	PAVULURI SOMA SEKHAR	Hitachi Energy	6.5
37	2200069034	ARNIPALLI MANI SANKAR	Pursuit Future Technologies	
38	2200069035	MULUKUTLA SRI RAMA KRISHNA	Omega Healthcare	3.25 LPA
39	2200069036	PAPANA SUMANTH	Launched Global	5-7 LPA
39	2200069036	PAPANA SUMANTH	Omega Healthcare	3.25 LPA
40	2200069038	JANGITI RAMU	KYB Corporation	3.36
41	2200069039	JANGITI LAKSHMAN	Sobha Constructions, Dubai	15.19
42	2200069040	MUSHKE DEVENDAR	Inspireleap	4.0 + 2.5 LPA
43	2200069041	EDLA SANDEEP	KYB Corporation	3.36
44	2200069042	SIRIGIDI SAI RAM	Krutanics	4.0 + 3.0 LPA
44	2200069042	SIRIGIDI SAI RAM	Launched Global	5-7 LPA
45	2200069043	SRINIVAS RAHUL RAJ MALOTH	Krutanics	4.0 + 3.0 LPA
46	2200069044	HARSHAVARDHAN REDDY THATIPARTHI	Pursuit Future Technologies	
46	2200069044	HARSHAVARDHAN REDDY THATIPARTHI	AllDigiTech	3 LPA CTC
46	2200069044	HARSHAVARDHAN REDDY THATIPARTHI	Divis Laboratories	3.1 LPA
47	2200069045	ASAM MANJUNATHA REDDY	KYB Corporation	3.36
48	2200069046	Sekharapalli Vishnu Vardhan	CTS	

49	2200069047	Venu Gopal Chakala	Krutanics	4.0 + 3.0 LPA
50	2200069048	Sudha.Kushala Kumar Reddy	KYB Corporation	3.36 LPA
51	2200069049	CH.Sai Ganesh Kumar	KYB Corporation	3.36 LPA
52	2200069050	DEVINENI PRANATHI	Tiger Machines Co. ltd, Japan	27.8 LPA
53	2200069051	SIDDAVATAM CHANDRA SEKHAR REDDY	MEIL	
54	2200069052	Tarun Boya	KYB Corporation	3.36 LPA
54	2200069052	Tarun Boya	AENEXZ	4.0 + 2.5 LPA
55	2200069053	SAI PHANINDHRA JAGATHA	AllDigiTech	3 LPA CTC
56	2200069054	THADISHETTI NARESH	KYB Corporation	3.36
57	2200069056	MOHANA VIJAYA BHASKAR KARELLA	KYB Corporation	3.36
58	2200069057	UNDELA VISHNU VARDHAN REDDY	KYB Corporation	3.36
59	2200069059	MOTAKATLA ANJI REDDY	AENEXZ	4.0 + 2.5 LPA
60	2200069060	YADALA VENKATA NAGA SAI MANIKANTA HARSHITH	KYB Corporation	3.36
61	2200069061	Vinay Kumar Reddy	Hexaware	3.2 LPA
62	2200069063	Valluri Nagavenkata Gayathri	Hitachi Energy	6.5
63	2200069064	Mallu Sathveeka Reddy	Inspireleap	4.0 + 2.5 LPA
64	2200069066	DEVARASETTY VARUN SAI	LTI Mindtree	4.05
65	2200069068	POLIBOINA Yogi Preetham	Hexaware	3.2 LPA
66	2200069069	NARRA MURALI MOHAN REDDY	AllDigiTech	3 LPA CTC
67	2200069070	SAI TEJA DURGAM	Omega Healthcare	3.25 LPA
68	2200069071	Sivakumar Pellilla	Divis Laboratories	3.1 LPA
69	2200069073	KARI MADHU SHALINI	KYB Corporation	3.36
70	2200069074	LONDA SRI VIDYA SAGAR YADAV	AllDigiTech	3 LPA CTC
71	2200069075	DOMMETI TEJA KUMAR	AllDigiTech	3 LPA CTC
71	2200069075	DOMMETI TEJA KUMAR	Divis Laboratories	3.1 LPA
72	2200069076	GOTTUMUKKALA VENKATA SAI MADHAV VARMA	Divis Laboratories	3.1 LPA
72	2200069076	GOTTUMUKKALA VENKATA SAI MADHAV VARMA	AllDigiTech	3 LPA CTC
73	2200069077	SURISETTI CHARITH VARDHAN	Maruti Air Systems	3.5
74	2200069079	M HEMANTH	AllDigiTech	3 LPA CTC

Y23 B,Tech

SNO	ID NUMBER	NAME	Placed in	CTC
1	2200069050	K SUMANTHIKA	Autobacs, Japan	20 LPA
2	2300069030	KANURI DHANUSH	TCS Digital	7.09 LPA

3	2300069049	MUNTHA SATHISH	TCS Digital	7.09 LPA
4	2300060029	VAKKAPATLA SUBHASHINI	TCS Ninja	3.46 LPA
5	2300069002	MALLA PRASHANTH	TCS Ninja	3.46 LPA
6	2300069033	JADDU SATYA SURESH	TCS Ninja	3.46 LPA
7	2300060007	PIDATHALA PRAVEEN KUMAR	Trekverse	4.5 LPA

Research & Development

Total no. of publications including Scopus and SCI indexed by the department are 42 no. and 7 papers published in conference.

06 scholars of the department were awarded with PhD degrees in the A.Y: 2025-2026 till date.

Dr. A Pandian, Dr. B Jyothi, and Dr. G G Raja Sekhar has received Best Researcher Awards by external organizations.

Dr. P Srinivasa Varma has received Patent Grant.

Dr. B Jyothi, " A High-Efficiency Ferdowsi PFC Converter for Reliable and Affordable EV Charging in Rural and Urban India" and Dr. S Ravi Teja , " A Test rig for Real-time Autonomous vehicle experiments (Phase I)" received Seed Money from KLEF

Electrical & Electronics Engineering Department published 33 patents in A.Y: 2025-2026.

B Deva Nithin (2501200001) M.Tech PEPS students has done Internship in Visakhapatnam Steel Plant.

AGENDA ITEM-2

Discussion of Stakeholders feedback and ATR

The DAC discussed the feedback received from various stakeholders including students, alumni, employers, and industry experts. Key feedback was related to the inclusion of emerging technologies, increased practical exposure, software tool training, and industry-oriented courses. The Action Taken Report (ATR) highlighting curriculum updates, laboratory enhancements, guest lectures, internships, and value-added courses was presented. The committee approved the ATR and recommended continuous monitoring of feedback implementation.

Annexure 1

AGENDA ITEM-3

To approve approve the modified curriculum and syllabus of B.Tech-EEE Y25 admitted batch	It is resolved to approve the modified curriculum and syllabus of B.Tech-EEE Y25 admitted batch and to recommend the same

	to BOS for further approval.

As per the contemporary requirements Y25 B.Tech EEE curriculum is modified by introducing courses related to Machine learning and Embedded System

Machine learning thrust area

- Deep Learning With Accelerated Computing
- Generative AI And AI Platform Engineering

Embedded System thrust area

- Embedded System Design And IOT
- Embedded Software Development
- Embedded AI And Tiny ML
- IOT Protocols And Cloud Integration
- Agentic Embedded Systems & Intelligent Edge Orchestration

All the DAC members has resolved to approve the modified curriculum and syllabus of B.Tech-EEE Y25 admitted batch and to recommend the same to BOS for further approval.

Annexure 2

AGENDA ITEM-4

To consider and approve the curriculum and syllabus of B.Tech-EEE Y26 admitted batch.	It is resolve to approve the curriculum and syllabus of B.Tech-EEE Y26 admitted batch. and to recommend for further approval in BOS.

It is resolved to offer B.Tech EEE program in two modes (i.e Semester, Trimester)

As per the latest thrust areas like Robotics DAC chairman presented the Y26 B,Tech EEE curriculum along with syllabus

Robotics thrust area

- Foundations of Robotics & Mechatronic Systems
- Robotic Perception, Computer Vision & Sensor Fusion
- Robotic Motion Planning, Navigation & Control Systems
- Autonomous Robotic Systems & Human-Robot Interaction

Annexure 3

AGENDA ITEM-5

To consider and approve the modification of syllabus AI DRIVEN ECU DESIGN FOR AUTONOMOUS VEHICLES (PE-5) for Y23 B.Tech	It is resolve to approve the modification of syllabus AI DRIVEN ECU DESIGN FOR AUTONOMOUS VEHICLES (PE-5) for Y23 B.Tech and recommend for further approval in BOS..

AI DRIVEN ECU DESIGN FOR AUTONOMOUS VEHICLES syllabus is modified as it is observed 40% of the syllabus is repeated.

Annexure 4

AGENDA ITEM-6

To consider and approve the Curriculum and Syllabus for M.Tech (PEPS) for Y26 admitted students.	It is resolve to approve the Curriculum and Syllabus for M.Tech (PEPS) for Y26 admitted students.and recommend for further approval in BOS.

M.Tech (PEPS) for Y26 admitted students is recommended to offer in Trimester based.

Annexure 5

AGENDA ITEM-7

To consider and approve the Value Added Courses offered in A.Y: 2026-2027 Odd Sem	It is resolve to approve the Value Added Courses offered in A.Y: 2026-2027 Odd Sem and recommend for further approval in BOS.

Value Added Courses offered in Odd Semester

Sl	Certification Course
1	ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING USING PYTHON [INDUSTRIAL AUTOMATION]-L2

AGENDA ITEM-8

To consider and approve PDDs of B.Tech(EEE), M.Tech (PEPS) of Y26admitted batch and Y25 B.Tech (EEE) of Y25 admitted batch	It is resolve to approve PDDs of B.Tech(EEE), M.Tech (PEPS) of Y26admitted batch and Y25 B.Tech (EEE) of Y25 admitted batch and recommend for further approval in BOS

PDDs of B.Tech(EEE), M.Tech (PEPS) of Y26admitted batch and Y25 B.Tech (EEE) of Y25 admitted batch are approved.

AGENDA ITEM-9

Course Closure Minutes of A.Y:2025-2026 Even Semester

The DAC reviewed the course closure reports of the A.Y:2025-2026 Even Semester. Course files, attainment of Course Outcomes (COs), internal assessment performance, laboratory completion, and syllabus coverage were discussed. Faculty members reported satisfactory completion of syllabus and assessments. The committee approved the course closure minutes.

AGENDA ITEM-10

Any other points raised by DAC members

DAC members suggested to include Measurements and Instrumentation topics in any CO in the courses as it carriers a weight in competitive exams

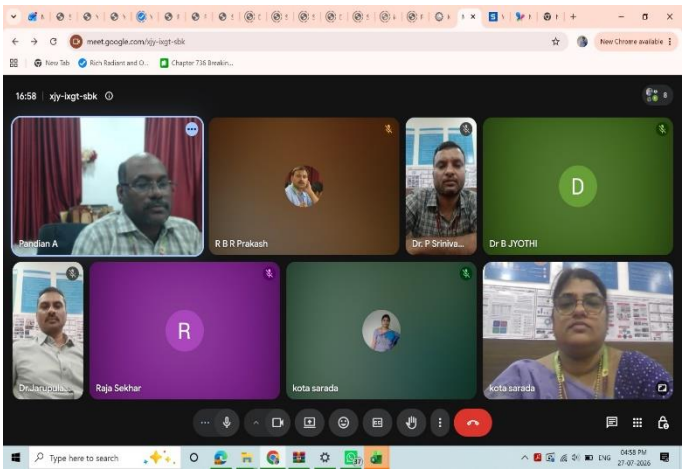
It is recommended to remove Essentials of Autonomous Systems course as pre-requisite for AI and Autonomous Systems as the concepts were covered in other courses of CSE, AIDS, CSIT students

Electrical & Electronics Engineering Department is planning to organise International Conference during November- December 2026.

The Head of the Department thanked all DAC members for their valuable inputs and constructive suggestions. The meeting ended with a resolution to continuously improve academic quality and align the curriculum with industry and societal needs.

Pictures of DAC meeting proceedings:





Koneru Lakshmaiah Education Foundation

(Category -1, Deemed to be University estd. u/s. 3 of the UGC Act, 1956)

Approved by AICTE ISO 21001:2018 Certified

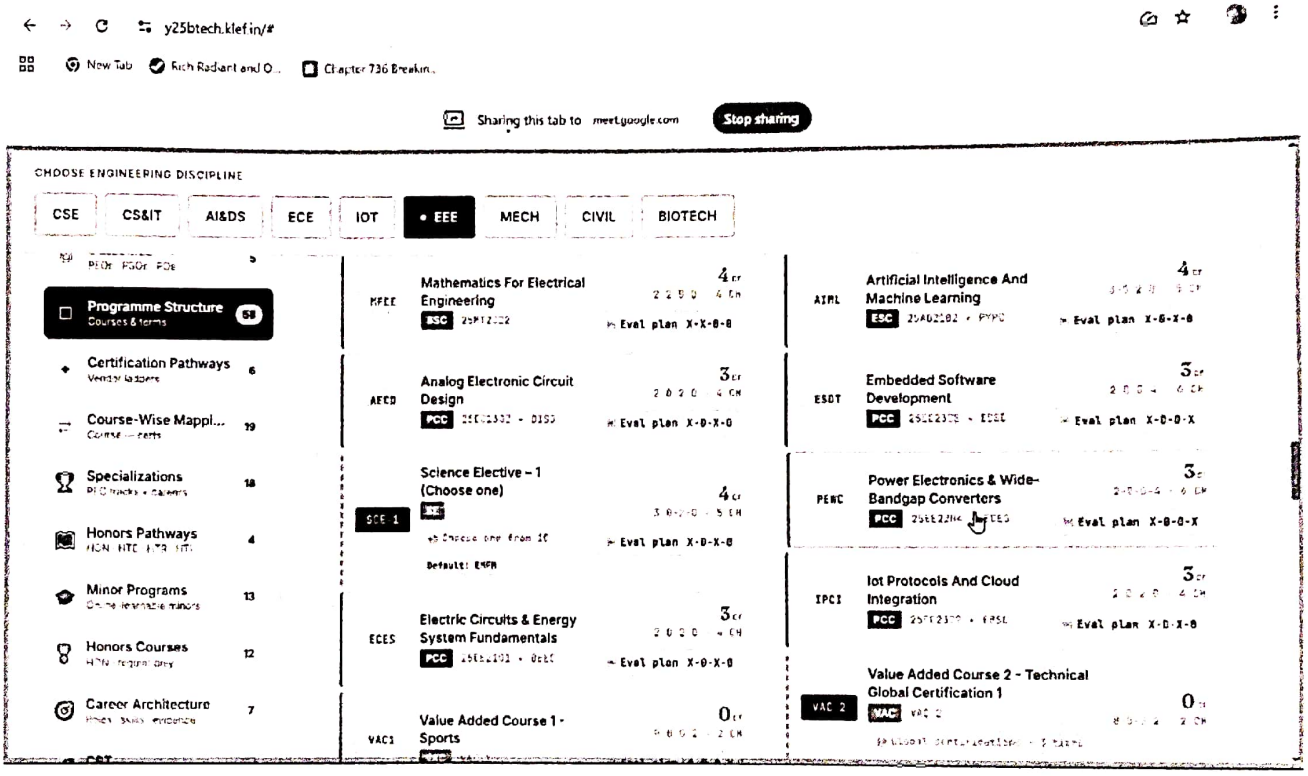
Campus: Green Fields, Vaddeswaram - 522 302, Guntur District, Andhra Pradesh, INDIA

Phone No. +91 8645 - 350 200, www.klef.ac.in, www.klef.edu.in, www.kluniversity.in

Admin Off: 29-36-38, Museum Road, Governorpet, Vijayawada - 520 002 Ph: +91 866 3500122 2576129







Dr.A. PANDIAN, SMIEEE
BOS Chairman
Professor & HOD
Department of EEE
KLEF Deemed to be University
Greeen Fields, Vaddeswaram,
Guntur Dt., A.P., 522 302.

Koneru Lakshmaiah Education Foundation
(Category -1, Deemed to be University estd. u/s. 3 of the UGC Act, 1956)
❖ Approved by AICTE ❖ ISO 21001:2018 Certified
Campus: Green Fields, Vaddeswaram - 522 302, Guntur District, Andhra Pradesh, INDIA.
Phone No. +91 8645 - 350 200; www.klef.ac.in; www.klef.edu.in; www.kluniversity.in
Admin Off: 29-36-38, Museum Road, Governorpet, Vijayawada - 520 002. Ph: +91 - 866 - 3500122, 2576129

Department of Electrical & Electronics Engineering

Stake Holders Feedback, A.Y: 2026-2027 ODD SEMESTER

Sl	Type of Stake holder	Number of feedbacks
1	Students	55
2	Parents	10
3	Alumni	10
4	Faculty	15
5	Academic peers	10
6	Industry persons	10
Total		110

SL	Stake Holder	Recommendations	Resolutions
1	Industry	Anmol Agarwal recommended to include Renewable Energy Systems & Grid Integration as core course for B.Tech EEE	Renewable Energy Systems & Grid Integration is resolved to introduce as core course from Y25 B.Tech EEE
2	Industry	B.Chandana Priya recommended to include Electric Vehicle Systems & Battery Technologies in the curriculum widely used in consumer electronics and industrial automation.	Electric Vehicle Systems & Battery Technologies is recommended to introduce in Y25 B.Tech EEE curriculum
3	Academic Peers	Dr.Dogga Raveendra recommended to include courses related to Embedded system thrust area to meet industry requirements	It is resolved to introduce courses like Embedded System Design And IOT, Embedded Software Development, Embedded AI And Tiny ML, IOT Protocols And Cloud Integration, Agentic Embedded Systems & Intelligent Edge Orchestration in Embedded system thrust area for Y25 B.Tech EEE
4	Academic Peers	Dr. Suresh Kumar Gadi recommended to introduce courses related Robotics thrust area	It is resolved to introduce courses Foundations of Robotics & Mechatronic Systems, Robotic Perception, Computer Vision & Sensor Fusion, Robotic Motion Planning, Navigation & Control Systems, Autonomous Robotic Systems & Human-Robot Interaction in Robotics thrust area thrust area for Y26 B.Tech EEE (Trimester)
5	Alumni	B. Ramesh recommended to add a course related to Supply Chain Management in Business Knowledge	As per the recommendation it is resolved to consider the introduction of course Supply Chain Management in Business Knowledge as Open Elective / management Elective for upcoming Y27 curriculum.
6	Alumni	B. Chandana Priya recommended to include Electric Vehicle	It is resolved to offer Electric Vehicle Systems & Battery Technologies as core

6	Alumni	B. Chandana Priya recommended to include Electric Vehicle Systems & Battery Technologies in the curriculum widely used in consumer electronics and industrial automation.	It is resolved to offer Electric Vehicle Systems & Battery Technologies as core course from Y25 B.Tech EEE.
7	Faculty	Dr. K Sarada recommended to include Distributed Energy Resources, Microgrids & Storage, Smart Grid concepts in core courses.	Smart-Grid Sensing & Communication, Distributed Energy Resources, Microgrids & Storage, Grid Operation, Analytics & Demand-Side Management concepts in the course Modern Power Systems & Smart Grid Technologies for Y25 B.Tech EEE
8	Faculty	PED Research group recommended to include Wide-Bandgap Devices, Modern Tooling & Indian Ecosystem in Power Electronic course into Y25 curriculum.	It is resolved to include Wide-Bandgap Devices, Modern Tooling & Indian Ecosystem concepts in Power Electronics & Wide-Bandgap Converters course for Y25 specialization.
9	Faculty	Dr. B Jyothi recommended to introduce Power Quality and Harmonics course for Y26 PG in EEE Department	It is resolved to introduce to offer Power Quality and Harmonics course for Y26 PG in EEE Department
10	Students	T Gowtham Sai Id No. 2300060018 requested to include the Edge AI, 5G+IoT, Digital Twins concepts in to curriculum for benefit of the students while doing projects.	Edge AI, 5G+IoT, Digital Twins concepts are introduced in IOT Protocols and Cloud Integration course for Y25 B.Tech EEE
11	Students	B Deva Nithin (2501200001) suggested to offer Smart EV Charging and IoT Energy Management for M.Tech students	Smart EV Charging and IoT Energy Management course is recommended to introduce for PG Curriculum for Y26 admitted batch.
12	Parents	Smt. Pinniboyina Jyothi suggested to include more courses related to for the benefit of students.	Deep Learning With Accelerated Computing, Generative AI And AI Platform Engineering courses are introduced in addition to AI and Machine learning in Y25 B.Tech Curriculum.
13	Parents	Vattikala Ravi recommended to introduce the course Energy Analytics, AI & Digital Twin Engineering	It is resolved to introduce the Energy Analytics, AI & Digital Twin Engineering course as elective in Electric Mobility & Smart Energy Engineering

Dr.A.PANDIAN, SMIEEE
Professor & HOD
Department of EEE
KLEF Deemed to be University
Greeen Fields, Vaddeswaram,
Guntur Dt., A.P., 522 302.

PROGRAMME STRUCTURE · Y25 B.TECH

Electrical & Electronics Engineering

Term-wise course structure with L-T-P-S, credits (CR), contact hours (CH) and pre-requisites. One academic year per page; summer terms follow their year.

Category-wise Credit Requirement — Regular (Semester) Model

	Category	Credits	Courses
HSS	Humanities & Social Sciences	15	6
BSC	Basic Sciences	20	5
ESC	Engineering Sciences	32	9
PCC	Professional Core	59	18
PEC	Professional Elective	15	5
OEC	Open Elective	8	2
PRI	Project / Internship	15	5
AUC	Audit Course · non-credit	Nil	4
VAC	Value Added Course · non-credit	Nil	4
	Total credits for graduation	**164**	**58**

ALL MANDATORY COURSES · Y25 B.TECH

Electrical & Electronics Engineering

Detailed content for every mandatory course — each formatted as its individual course handout (one course per page), including both Regular and PBL variants (distinct course codes) where offered, sorted A–Z by course title. Experiential PRI courses are in a separate document.

39 course handouts (Regular + PBL where offered), A–Z by title.

Engineering Chemistry

25CY1001

L-T-P-S	3-0-2-0	Credits	4	Contact Hrs/Wk	5
Semester	SCE-1, SCE-2	Mode	Regular	Category	SCE
Pre-requisite	NIL				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand structure and bonding — atomic structure, bonding, and the structure-property link.	BTL2 · Understand
CO2	Apply electrochemistry — cells, electrode potentials, batteries, and fuel cells.	BTL3 · Apply
CO3	Analyze corrosion and its control — mechanisms, protection, and material durability.	BTL4 · Analyze
CO4	Apply polymer and materials chemistry — polymers, composites, and engineering materials.	BTL3 · Apply
CO5	Apply water and environmental chemistry — water treatment, analysis, and sustainability.	BTL3 · Apply
CO6	Analyze energy materials and modern chemistry — batteries, hydrogen, green chemistry, and AI for materials.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO7	PSO1	PSO2
CO1	H	H	–	–	–	H	–
CO2	H	–	H	–	–	H	–
CO3	–	H	–	–	H	H	–
CO4	–	–	H	H	–	H	–
CO5	–	–	H	–	H	–	H
CO6	–	H	–	–	H	–	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Foundations: Structure & Bonding** Atomic structure and quantum numbers.; Chemical bonding (ionic, covalent, metallic).; Molecular orbital theory at intuition.; Intermolecular forces.; Structure-property relationships.; Phase behaviour and basics of thermodynamics.; Spectroscopy for characterization at intuition.; Lab: characterize a material and connect structure to its properties.	CO1
M2	**Electrochemistry & Energy Storage** Electrochemical cells and electrode potentials.; The Nernst equation.; Conductance and electrolytes.; Primary and secondary batteries.; Lithium-ion battery chemistry.; Fuel cells and hydrogen.; Supercapacitors at intuition.; Lab: build/characterize an electrochemical cell and measure performance.	CO2
M3	**Corrosion & Its Control** Why metals corrode — electrochemistry of corrosion.; Types of corrosion.; Factors affecting corrosion rate.; Cathodic and anodic protection.; Coatings and inhibitors.; Corrosion-resistant materials.; Corrosion in the Indian context (infrastructure).; Lab: measure a corrosion rate and test a protection method.	CO3
M4	**Polymers & Engineering Materials** Polymerization and polymer structure.; Thermoplastics, thermosets, elastomers.; Conducting and smart polymers.; Composites and reinforcement.; Engineering ceramics and glasses.; Nanomaterials at intuition.; Material selection basics.; Lab: synthesize/characterize a polymer or composite.	CO4
M5	**Water & Environmental Chemistry** Water quality parameters.; Hardness and its treatment.; Water treatment for industry/boilers.; Wastewater chemistry.; Air-pollution chemistry at intuition.; Analytical methods (titrimetry, instrumental).; Sustainability and regulation.; Lab: analyze water quality and design a treatment.	CO5

	Title & Topics	CO
M6	**Energy Materials & Modern Chemistry (AI) + Capstone** Advanced battery and beyond-Li chemistries.; Hydrogen production and the green-H2 economy.; Solar/photovoltaic materials.; Green chemistry principles.; Catalysis and process chemistry.; AI/ML-accelerated materials discovery.; The energy-transition materials landscape (India).; Lab/Showcase: present a ChemLab energy-material or anti-corrosion solution with a structure-property + sustainability analysis.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Engineering Chemistry**	P. C. Jain & Monika Jain	Dhanpat Rai Publishing	2015
2	**A Textbook of Engineering Chemistry**	Shashi Chawla	Dhanpat Rai & Co.	2017
3	**Engineering Chemistry**	S. S. Dara & S. S. Umare	S. Chand	2018
4	**Electrochemical Methods: Fundamentals and Applications, 3rd Ed.**	Allen J. Bard, Larry R. Faulkner & Henry S. White	Wiley	2022
5	**Polymer Science and Technology, 3rd Ed.**	Joel R. Fried	Prentice Hall	2014
6	**Materials Science and Engineering: An Introduction, 10th Ed.**	William D. Callister & David G. Rethwisch	Wiley	2018
7	**Green Chemistry: Theory and Practice**	Paul T. Anastas & John C. Warner	Oxford University Press	1998

Electromagnetic Fields & Engineering Materials

25EE2102

L-T-P-S	3-0-2-0	**Credits**	4	**Contact Hrs/Wk**	5
Semester	SCE-1	**Mode**	Regular	**Category**	SCE
Pre-requisite	Engineering Mathematics-I, Basic Electrical Engineering				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the structure-property relationship of engineering materials — crystal structures and defects, dielectric and insulating materials, magnetic materials and ferrites, conductors, photo-conductive materials, semiconductors, nano-materials and superconductors — in their canonical EEE applications.	BTL2 · Understand
CO2	Understand vector calculus for electromagnetics — scalars and vectors, differential length / area / volume elements in Cartesian / cylindrical / spherical coordinates, line / surface / volume integrals, gradient, divergence, curl, the Del operator, and the canonical Divergence and Stokes' theorems.	BTL2 · Understand
CO3	Apply Coulomb's law, electric-field intensity, Gauss's law and electric potential to determine the electrostatic field due to point, line, surface and volume charge distributions, and analyse the behaviour of conductors and dielectrics in electrostatic fields.	BTL3 · Apply
CO4	Apply Biot-Savart's law and Ampère's law to compute magnetostatic fields due to canonical current distributions, analyse magnetic circuits, and compute self- and mutual inductance of simple geometries used in transformers and electrical machines.	BTL3 · Apply
CO5	Analyse Faraday's law of electromagnetic induction, time-varying fields, the relationship between transformer and motional EMFs, the displacement-current term, capacitance of parallel-plate / spherical / coaxial geometries, and Maxwell's equations in point and integral form.	BTL4 · Analyze
CO6	Analyse Lorentz force on moving charges and current-carrying elements, force and torque on a closed circuit, electric and magnetic boundary conditions, and connect the analysis to contemporary applications — EV-motor magnetics, 5G / mmWave antenna engineering, EM-simulation tools, and modern materials for power electronics.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO7	PSO1
CO1	**H**	M	M	–	–	–	**H**
CO2	**H**	**H**	M	–	–	–	**H**
CO3	**H**	**H**	M	M	–	–	**H**
CO4	**H**	**H**	**H**	M	–	–	**H**
CO5	**H**	**H**	**H**	**H**	M	–	**H**
CO6	**H**	**H**	**H**	–	**H**	M	**H**

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Engineering Materials — Structure & Properties** Crystal structures — BCC, FCC, HCP; coordination number and atomic packing factor; the canonical Bravais-lattice classification.; Crystal defects — point defects (vacancies, interstitials, substitutional), line defects (dislocations), surface defects (grain boundaries); the canonical effect on conductivity and strength.; Dielectric materials — ceramic and polymer dielectrics; polarization mechanisms (electronic, ionic, orientational, space-charge); dielectric constant and loss; the canonical capacitor-dielectric applications.; Insulating materials — mica, porcelain, glass, polyethylene, XLPE, EPR; the canonical IS 7098 transformer-oil and cable-insulation context.; Magnetic materials — diamagnetic, paramagnetic, ferromagnetic, antiferromagnetic, ferrimagnetic; B-H curve, hysteresis, saturation, coercivity, remanence; the canonical soft (transformer core) vs hard (permanent magnet) classification.; Ferrites and ferromagnetic components — manganese-zinc, nickel-zinc ferrites; the canonical SMPS-transformer / inductor / RF-choke application.; Conductors and photo-conductivity — the canonical copper / aluminium classification; photoconductive materials (CdS, Si, Ge) and their canonical light-sensor / solar-cell applications.; Nano-materials and superconductors — graphene, carbon nanotubes, quantum dots; type-I vs type-II superconductors, BCS theory at intuition, high-T_c cuprates; canonical IISc / TIFR / IUAC research benchmarks.	CO1

PROGRAMME STRUCTURE · Y26 B.TECH

Electrical & Electronics Engineering

Term-wise course structure with L-T-P-S, credits (CR), contact hours (CH) and pre-requisites. One academic year per page; summer terms follow their year.

Category-wise Credit Requirement — Regular (Semester) Model

	Category	Credits	Courses
HSS	Humanities & Social Sciences	13	6
BSC	Basic Sciences	20	5
ESC	Engineering Sciences	33	9
PCC	Professional Core	58	16
PEC	Professional Elective	15	5
OEC	Open Elective	9	3
PRI	Project / Internship	15	5
AUC	Audit Course · non-credit	Nil	4
VAC	Value Added Course · non-credit	Nil	4
	Total credits for graduation	**163**	**57**

Category-wise Credit Requirement — PBL (Trimester) Model

	Category	Credits	Courses
HSS	Humanities & Social Sciences	15	7
BSC	Basic Sciences	20	5
ESC	Engineering Sciences	33	9
PCC	Professional Core	76	21
PEC	Professional Elective	15	5
OEC	Open Elective	9	3
PRI	Project / Internship	15	6
AUC	Audit Course · non-credit	Nil	4
VAC	Value Added Course · non-credit	Nil	5
	Total credits for graduation	**183**	**65**

Category-wise Credit Requirement — Lateral Entry (Semester) Model

	Category	Credits	Courses
HSS	Humanities & Social Sciences	5	2
BSC	Basic Sciences	12	3
ESC	Engineering Sciences	16	4
PCC	Professional Core	54	15
PEC	Professional Elective	15	5
OEC	Open Elective	9	3
PRI	Project / Internship	15	4
AUC	Audit Course · non-credit	Nil	3
VAC	Value Added Course · non-credit	Nil	4
BRIDGE	BRIDGE · non-credit	Nil	4
	Total credits for graduation	**126**	**47**

Regular (Semester) Track — Year 1

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
S01	26UC1137	DTIE	Design Thinking for Innovation & Entrepreneurship	HSS	0-0-4-0	2	4	-
	26UC1102	LSFE	Language Skills for Engineers	HSS	0-0-4-0	2	4	-
	26MT1101	MTCP	Mathematics for Computation	BSC	2-2-0-0	4	4	-
	26EC1202	DDCA	Digital Design & Computer Architecture	ESC	2-0-0-4	3	6	-
	26SC1101	PSPJ	Problem Solving Using Programming (Java)	ESC	2-0-0-8	4	10	-
	FLP	FLNG	Foreign Language	HSS	0-0-4-0	2	4	-
S02	26UC1204	CSFE	Communication Skills for Engineers	HSS	0-0-4-0	2	4	LSFE
	26MT1202	MTAI	Mathematics for AI	BSC	2-2-0-0	4	4	MTCP
	26EC1101	FOIS	Fundamentals of IoT & Sensors	ESC	2-0-4-0	4	6	-
	26EE1201	ECES	Electric Circuits & Energy System Fundamentals	PCC	2-0-2-4	4	8	-
	26SC1306	CFAI	Computational Foundations for AI	ESC	2-0-0-4	3	6	-
	26SC1203	DSA1	Data Structures & Algorithms – 1	ESC	2-0-0-4	3	6	PSPP
Summer 1	26IE2040	INTS	Social Internship	PRI	0-0-4-0	0	4	-
	26UC0032	SDGL	Insights to Sustainable Development Goals	AUC	2-0-0-0	0	2	-

Year 1 — Courses: 14 · Credits: 37 · Contact Hours/Week: 72

Regular (Semester) Track — Year 2

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
S03	26MT1305	MTES	Mathematics for Electrical Systems	BSC	2-2-0-0	4	4	MTCP
	26SC1308	FSWD	Full Stack Web Development	ESC	2-0-4-0	4	6	PSPP
	26SC2110	MLRN	Machine Learning	ESC	2-0-4-0	4	6	CFAI
	26EC2106	ESDI	Embedded System Design & IoT	ESC	2-0-4-0	4	6	FOIS
	26SC1305	DSA2	Data Structures & Algorithms – 2	ESC	2-0-2-4	4	8	DSA1
	SCE-1	SCE1	Science Elective – 1	BSC	3-0-2-0	4	5	-
	VAC-1	VAC1	Value Added Course 1 - Sports	VAC	0-0-0-2	0	2	-
S04	26EE2203	EMDE	Electric Motors, Drives & Electromechanical Systems	PCC	2-0-0-4	3	6	ECES
	26EE2204	EGTD	Electric Power Generation, Transmission and Distribution	PCC	3-1-0-0	4	4	ECES
	26EC2310	ESWD	Embedded Software Development	PCC	2-0-4-0	4	6	ESDI
	26EC1203	AECD	Analog Electronic Circuit Design	PCC	2-0-0-4	3	6	-
	26IN2201	IOTP	IoT Protocols & Cloud Integration	PCC	2-0-2-4	4	8	FOIS
	26EE2205	PEWB	Power Electronics & Wide-Bandgap Converters	PCC	2-0-0-4	3	6	ECES
	VAC-2	VAC2	Value Added Course 2 - Technical Global Certification 1	VAC	0-0-0-2	0	2	-
Summer 2	26IE3041	INTT	Technical Internship	PRI	0-0-4-0	0	4	-
	26UC0017	IKSL	Indian Knowledge Systems	AUC	2-0-0-0	0	2	-

Year 2 — Courses: 16 · Credits: 45 · Contact Hours/Week: 81

Regular (Semester) Track — Year 3

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
S05	26EE2306	CSEE	Control Systems for Electrical & Energy Applications	PCC	2-0-4-0	4	6	ECES
	26EE4111	REGI	Renewable Energy Systems & Grid Integration	PCC	2-0-4-0	4	6	EGTD
	26EE2307	MPSG	Modern Power Systems & Smart Grid Technologies	PCC	2-0-4-0	4	6	EGTD
	26AD2304	DLRN	Deep Learning with Accelerated Computing	PCC	2-0-0-4	3	6	MLRN
	26EC3113	EATM	Embedded AI & Tiny ML	PCC	2-0-4-0	4	6	ESWD
	PE-1	PEC1	Professional Elective – 1	PEC	2-0-0-4	3	6	-
	VAC-3	VAC3	Value Added Course 3 - Technical Global Certification 2	VAC	0-0-0-2	0	2	-
S06	26AD2202	GNAI	Generative AI & AI Platform Engineering	PCC	2-0-4-0	4	6	MLRN
	26IE4101	TMPR	Term Paper	PRI	0-0-6-0	3	6	-
	26EE3309	PSPC	Power System Protection and Control	PCC	2-0-2-0	3	4	EGTD
	26EE3208E	IAPS	Industrial Automation, PLC & SCADA Systems	PCC	2-0-4-0	4	6	CSEE
	PE-2	PEC2	Professional Elective – 2	PEC	3-0-0-0	3	3	PEC1
	PE-3	PEC3	Professional Elective – 3	PEC	2-0-0-4	3	6	PEC1
	VAC-4	VAC4	Value Added Course 4 - Technical Global Certification 3	VAC	0-0-0-2	0	2	-
Summer 3	26UC0009	ECEN	Ecology & Environment	AUC	2-0-0-0	0	2	-

Year 3 — Courses: 15 · Credits: 42 · Contact Hours/Week: 73

Regular (Semester) Track — Year 4

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
S07	26EE4110	EVBT	Electric Vehicle Systems & Battery Technologies	PCC	1-0-2-4	3	7	PEWB
	26UC0008	INCT	Indian Constitution	AUC	2-0-0-0	0	2	-
	OE-1	OEC1	Open Elective – 1	OEC	2-0-2-0	3	4	-
	SCE-2	SCE2	Science Elective – 2	BSC	3-0-2-0	4	5	-
	PE-4	PEC4	Professional Elective – 4	PEC	3-0-0-0	3	3	PEC1
	26IE4053	CAP1	Capstone Project – 1	PRI	0-0-12-0	6	12	TMPR
S08	26UC0026	HVEG	Human Values, Gender Equality & Professional Ethics	HSS	2-0-0-0	2	2	-
	OE-2	OEC2	Open Elective – 2	OEC	2-0-2-0	3	4	-
	OE-3	OEC3	Open Elective – 3	OEC	2-0-2-0	3	4	-
	ME	MGMT	Management Elective	HSS	2-0-2-0	3	4	-
	PE-5	PEC5	Professional Elective – 5	PEC	3-0-0-0	3	3	PEC1
	26IE4054	CAP2	Capstone Project – 2	PRI	0-0-12-0	6	12	CAP1

Year 4 — Courses: 12 · Credits: 39 · Contact Hours/Week: 62

Regular (Semester) Track total — Courses: 57 · Credits: 163 · Contact Hours/Week: 297

PBL (Trimester) Track — Year 1

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
T01	26SC1101E	PSPP	Problem Solving Using Programming (Java)	ESC	3-0-6-0	4	9	-
	26UC1137E	DTIE	Design Thinking for Innovation & Entrepreneurship	HSS	3-0-0-0	2	3	-
	26UC1102E	LSFE	Language Skills for Engineers	HSS	3-0-0-0	2	3	-
	26MT1101E	MTCP	Mathematics for Computation	BSC	3-3-0-0	4	6	-
	26EC1101E	FOIS	Fundamentals of IoT & Sensors	ESC	3-3-0-0	4	6	-
T02	26EC1202E	DDCA	Digital Design & Computer Architecture	ESC	3-0-0-6	3	9	-
	26SC1203E	DSA1	Data Structures & Algorithms – 1	ESC	3-0-0-6	3	9	PSPP
	26MT1202E	MTAI	Mathematics for AI	BSC	3-3-0-0	4	6	MTCP
	26EE1201E	ECES	Electric Circuits & Energy System Fundamentals	PCC	3-0-6-0	4	9	-
	26UC1204E	CSFE	Communication Skills for Engineers	HSS	3-0-0-0	2	3	LSFE
T03	26SC1305E	DSA2	Data Structures & Algorithms – 2	ESC	3-0-6-0	4	9	DSA1
	26SC1306E	CFAI	Computational Foundations for AI	ESC	3-0-0-6	3	9	MTAI
	26MT1305E	MTEE	Mathematics for Electrical Systems	BSC	3-3-0-0	4	6	MTCP
	26SC1308E	FSWD	Full Stack Web Development	ESC	3-0-6-0	4	9	PSPP
	FLP-1	FLG1	Foreign Language Proficiency – 1	HSS	3-0-0-0	2	3	-
Summer 1	26IE2040E	INTS	Social Internship	PRI	0-0-6-0	0	6	-
	26UC0032E	SDGL	Insights to Sustainable Development Goals	AUC	3-0-0-0	0	3	-

Year 1 — Courses: 17 · Credits: 49 · Contact Hours/Week: 108

PBL (Trimester) Track — Year 2

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
T04	26EC1203E	AECD	Analog Electronic Circuit Design	PCC	3-0-0-6	3	9	-
	26SC2108E	MLRN	Machine Learning	ESC	3-0-6-0	4	9	CFAI
	FLP-2	FLG2	Foreign Language Proficiency – 2	HSS	3-0-0-0	2	3	FLG1
	SCE-1	SCE1	Science Elective – 1	BSC	3-0-6-0	4	9	-
	26EC2106E	ESDI	Embedded System Design & IoT	ESC	3-0-6-0	4	9	FOIS
T05	26EE2203E	EMDE	Electric Motors, Drives & Electromechanical Systems	PCC	3-0-6-0	4	9	ECES
	26EE2204E	EGTD	Electric Power Generation, Transmission and Distribution	PCC	3-3-0-0	4	6	ECES
	26AD2202E	GNAI	Generative AI & AI Platform Engineering	PCC	3-0-6-0	4	9	MLRN
	26EE2205E	PEWB	Power Electronics & Wide-Bandgap Converters	PCC	3-0-0-6	3	9	ECES
	VAC-1	VAC1	Value Added Course 1 - Sports	VAC	0-0-0-2	0	2	-
T06	26EE2306E	CSEE	Control Systems for Electrical & Energy Applications	PCC	3-0-6-0	4	9	ECES
	26EC2310E	ESWD	Embedded Software Development	PCC	3-0-6-0	4	9	ESDI
	26EE2307E	MPSG	Modern Power Systems & Smart Grid Technologies	PCC	3-0-6-0	4	9	EGTD
	26IN2201E	IOTP	IoT Protocols & Cloud Integration	PCC	3-0-6-0	4	9	FOIS
	VAC-2	VAC2	Value Added Course 2 - Technical Global Certification 1	VAC	0-0-0-2	0	2	-
Summer 2	26IE3041E	INTT	Technical Internship	PRI	0-0-6-0	0	6	-
	26UC0017E	IKSL	Indian Knowledge Systems	AUC	3-0-0-0	0	3	-

Year 2 — Courses: 17 · Credits: 48 · Contact Hours/Week: 121

PBL (Trimester) Track — Year 3

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
T07	26EC3112E	RBFN	Foundations of Robotics & Mechatronic Systems	PCC	3-0-0-6	3	9	EBSD
	26AD2304E	DLRN	Deep Learning with Accelerated Computing	PCC	3-0-0-6	3	9	MLRN
	26EC3113E	EATM	Embedded AI & Tiny ML	PCC	3-0-6-0	4	9	DLRN
	PE-1	PEC1	Professional Elective – 1	PEC	3-0-0-6	3	9	-
	VAC-3	VAC3	Value Added Course 3 - Technical Global Certification 2	VAC	0-0-0-2	0	2	-
T08	26EE3208E	IAPS	Industrial Automation, PLC & SCADA Systems	PCC	3-0-6-0	4	9	CSEE
	26EC3214E	RBPR	Robotic Perception, Computer Vision & Sensor Fusion	PCC	3-0-6-0	4	9	RBFN
	PE-2	PEC2	Professional Elective – 2	PEC	3-0-0-6	3	9	PEC1
	PE-3	PEC3	Professional Elective – 3	PEC	3-0-0-6	3	9	PEC1
	VAC-4	VAC4	Value Added Course 4 - Technical Global Certification 3	VAC	0-0-0-2	0	2	-
T09	26EE3309E	PSPC	Power System Protection and Control	PCC	3-0-0-6	3	9	EGTD
	26EC3316E	RBMT	Robotic Motion Planning, Navigation & Control Systems	PCC	3-0-6-0	4	9	RBFN
	PE-4	PEC4	Professional Elective – 4	PEC	3-0-0-6	3	9	PEC1
	PE-5	PEC5	Professional Elective – 5	PEC	3-0-0-6	3	9	PEC1
	VAC-5	VAC5	Value Added Course 5 - Technical Global Certification 4	VAC	0-0-0-2	0	2	-
Summer 3	26IE4042E	INTR	Research / Industrial Internship	PRI	0-0-0-0	0	0	-
	26UC0009E	ECEN	Ecology & Environment	AUC	3-0-0-0	0	3	-

Year 3 — Courses: 17 · Credits: 40 · Contact Hours/Week: 117

PBL (Trimester) Track — Year 4

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
T10	26EE4110E	EVBT	Electric Vehicle Systems & Battery Technologies	PCC	3-0-0-6	3	9	PEWB
	26EE4111E	REGI	Renewable Energy Systems & Grid Integration	PCC	3-0-6-0	4	9	EGTD
	26EC4117E	ARBS	Autonomous Robotic Systems & Human-Robot Interaction	PCC	3-0-3-0	3	6	RBPR
	26EC4118E	AGEM	Agentic Embedded Systems & Intelligent Edge Orchestration	PCC	3-0-0-6	3	9	ESDI
	26IE4101E	TMPR	Term Paper	PRI	0-3-3-0	3	6	-
T11	26IE4053E	CAP1	Capstone Project – 1	PRI	0-0-12-12	6	12	TMPR
	SCE-2	SCE2	Science Elective – 2	BSC	3-0-6-0	4	9	-
	OE-1	OEC1	Open Elective – 1	OEC	3-0-3-0	3	6	-
	OE-2	OEC2	Open Elective – 2	OEC	3-0-3-0	3	6	-
	26UC0008E	INCT	Indian Constitution	AUC	3-0-0-0	0	3	-
T12	26IE4054E	CAP2	Capstone Project – 2	PRI	0-0-12-12	6	12	CAP1
	26UC0026E	HVEG	Human Values, Gender Equality & Professional Ethics	HSS	3-0-0-0	2	3	-
	OE-3	OEC3	Open Elective – 3	OEC	3-0-3-0	3	6	-
	ME	MGMT	Management Elective	HSS	3-0-3-0	3	6	-

Year 4 — Courses: 14 · Credits: 46 · Contact Hours/Week: 102

PBL (Trimester) Track total — Courses: 65 · Credits: 183 · Contact Hours/Week: 448

Lateral Entry (Semester) Track — Year 2

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
S03	26BR0301	PSPY	Problem Solving using Programming (Python)	BRIDGE	0-0-4-0	0	4	None
	26BR0302	MENG	Mathematics for Engineers	BRIDGE	0-0-4-0	0	4	None
	26MT1305	MTES	Mathematics for Electrical Systems	BSC	2-2-0-0	4	4	MTCP
	26SC1308	FSWD	Full Stack Web Development	ESC	2-0-4-0	4	6	PSPP
	26SC2110	MLRN	Machine Learning	ESC	2-0-4-0	4	6	CFAI
	26EC2106	ESDI	Embedded System Design & IoT	ESC	2-0-4-0	4	6	FOIS
	26SC1305	DSA2	Data Structures & Algorithms – 2	ESC	2-0-2-4	4	8	DSA1
	SCE-1	SCE1	Science Elective – 1	BSC	3-0-2-0	4	5	-
	VAC-1	VAC1	Value Added Course 1 - Sports	VAC	0-0-0-2	0	2	-
S04	26BR0401	LCSE	Language and Communication Skills for Engineers	BRIDGE	0-0-4-0	0	4	None
	26BR0402	DSTR	Data Structures	BRIDGE	0-0-4-0	0	4	None
	26EE2203	EMDE	Electric Motors, Drives & Electromechanical Systems	PCC	2-0-0-4	3	6	ECES
	26EE2204	EGTD	Electric Power Generation, Transmission and Distribution	PCC	3-1-0-0	4	4	ECES
	26EC2310	ESWD	Embedded Software Development	PCC	2-0-4-0	4	6	ESDI
	26EC1203	AECD	Analog Electronic Circuit Design	PCC	2-0-0-4	3	6	-
	26IN2201	IOTP	IoT Protocols & Cloud Integration	PCC	2-0-2-4	4	8	FOIS
	26EE2205	PEWB	Power Electronics & Wide-Bandgap Converters	PCC	2-0-0-4	3	6	ECES
	VAC-2	VAC2	Value Added Course 2 - Technical Global Certification 1	VAC	0-0-0-2	0	2	-
Summer 2	26IE3041	INTT	Technical Internship	PRI	0-0-4-0	0	4	-
	26UC0017	IKSL	Indian Knowledge Systems	AUC	2-0-0-0	0	2	-

Year 2 — Courses: 20 · Credits: 45 · Contact Hours/Week: 97

Lateral Entry (Semester) Track — Year 3

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
S05	26EE2306	CSEE	Control Systems for Electrical & Energy Applications	PCC	2-0-4-0	4	6	ECES
	26EE4111	REGI	Renewable Energy Systems & Grid Integration	PCC	2-0-4-0	4	6	EGTD
	26EE2307	MPSG	Modern Power Systems & Smart Grid Technologies	PCC	2-0-4-0	4	6	EGTD
	26AD2304	DLRN	Deep Learning with Accelerated Computing	PCC	2-0-0-4	3	6	MLRN
	26EC3113	EATM	Embedded AI & Tiny ML	PCC	2-0-4-0	4	6	ESWD
	PE-1	PEC1	Professional Elective – 1	PEC	2-0-0-4	3	6	-
	VAC-3	VAC3	Value Added Course 3 - Technical Global Certification 2	VAC	0-0-0-2	0	2	-
S06	26AD2202	GNAI	Generative AI & AI Platform Engineering	PCC	2-0-4-0	4	6	MLRN
	26IE4101	TMPR	Term Paper	PRI	0-0-6-0	3	6	-
	26EE3309	PSPC	Power System Protection and Control	PCC	2-0-2-0	3	4	EGTD
	26EE3208E	IAPS	Industrial Automation, PLC & SCADA Systems	PCC	2-0-4-0	4	6	CSEE
	PE-2	PEC2	Professional Elective – 2	PEC	3-0-0-0	3	3	PEC1
	PE-3	PEC3	Professional Elective – 3	PEC	2-0-0-4	3	6	PEC1
	VAC-4	VAC4	Value Added Course 4 - Technical Global Certification 3	VAC	0-0-0-2	0	2	-
Summer 3	26UC0009	ECEN	Ecology & Environment	AUC	2-0-0-0	0	2	-

Year 3 — Courses: 15 · Credits: 42 · Contact Hours/Week: 73

Lateral Entry (Semester) Track — Year 4

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
S07	26EE4110	EVBT	Electric Vehicle Systems & Battery Technologies	PCC	1-0-2-4	3	7	PEWB
	26UC0008	INCT	Indian Constitution	AUC	2-0-0-0	0	2	-
	OE-1	OEC1	Open Elective – 1	OEC	2-0-2-0	3	4	-
	SCE-2	SCE2	Science Elective – 2	BSC	3-0-2-0	4	5	-
	PE-4	PEC4	Professional Elective – 4	PEC	3-0-0-0	3	3	PEC1
	26IE4053	CAP1	Capstone Project – 1	PRI	0-0-12-0	6	12	TMPR
S08	26UC0026	HVEG	Human Values, Gender Equality & Professional Ethics	HSS	2-0-0-0	2	2	-
	OE-2	OEC2	Open Elective – 2	OEC	2-0-2-0	3	4	-
	OE-3	OEC3	Open Elective – 3	OEC	2-0-2-0	3	4	-
	ME	MGMT	Management Elective	HSS	2-0-2-0	3	4	-
	PE-5	PEC5	Professional Elective – 5	PEC	3-0-0-0	3	3	PEC1
	26IE4054	CAP2	Capstone Project – 2	PRI	0-0-12-0	6	12	CAP1

Year 4 — Courses: 12 · Credits: 39 · Contact Hours/Week: 62

Lateral Entry (Semester) Track total — Courses: 47 · Credits: 126 · Contact Hours/Week: 232

	Title & Topics	CO
M2	**Vector Calculus for Electromagnetics** Scalars and vectors — scalar / vector / mixed triple products; canonical applications in EM (E·dl, J·dS, A×B).; Coordinate systems — Cartesian (x, y, z), cylindrical (ρ, φ, z), spherical (r, θ, φ); the canonical transformation matrices.; Differential length, area and volume elements — the canonical dl, dS, dv expressions in each coordinate system.; Line, surface and volume integrals — the canonical computation steps; canonical examples (work done, flux through a surface, charge in a volume).; Gradient — ∇φ and its geometric interpretation; the canonical "direction of steepest ascent".; Divergence — ∇·A and its geometric interpretation; the canonical "net outward flux per unit volume".; Curl — ∇×A and its geometric interpretation; the canonical "rotational tendency at a point".; Integral theorems — Divergence theorem ($\iiint \nabla \cdot A\, dv = \oint A \cdot dS$) and Stokes' theorem ($\iint \nabla \times A \cdot dS = \oint A \cdot dl$); the canonical step toward Maxwell-equation integral forms.	CO2
M3	**Electrostatics** Coulomb's law — force between point charges; vector form; the canonical inverse-square scaling and superposition.; Electric-field intensity E and electric-flux-density D — their relationship via permittivity $D = \varepsilon E$; $\varepsilon = \varepsilon_0 \varepsilon_r$; the canonical free-space and dielectric distinctions.; Gauss's law — integral and differential form; the canonical "choose a Gaussian surface" problem-solving technique; applications to spherical / cylindrical / planar symmetry.; Electric field and potential due to point, line, surface and volume charge distributions — the canonical worked examples (infinite-line, ring, disk, sphere, parallel plates).; Conductors in electric field — the canonical "E = 0 inside, normal at surface"; surface-charge density; the canonical Faraday-cage shielding.; Dielectrics in electric field — polarization P, bound-charge density; the canonical $D = \varepsilon_0 E + P$ relationship; dielectric breakdown and dielectric strength.; Capacitance — parallel-plate, spherical, cylindrical, coaxial geometries; the canonical formulas; energy stored ½ C V^2; canonical capacitor-bank applications in power-factor correction.; Lab — measure capacitance of a parallel-plate capacitor with varying dielectric; verify $C = \varepsilon_0 \varepsilon_r A / d$ and ε_r values for common materials.	CO3
M4	**Magnetostatics** Magnetic-field intensity H and magnetic-flux-density B — their relationship via permeability $B = \mu H$; $\mu = \mu_0 \mu_r$; the canonical free-space and magnetic-material distinctions.; Biot-Savart's law — magnetic field due to a current element; the canonical worked examples (straight wire, circular loop, solenoid).; Ampère's law — the canonical $\oint H \cdot dl = I_{enclosed}$; applications to infinite line, toroid, infinite sheet of current.; Magnetic scalar and vector potentials — the canonical step toward retarded-potential and antenna theory in later courses.; Magnetic circuits — magnetomotive force, reluctance, the canonical analogy to electric-circuit Ohm's law; applications to transformer and motor design.; Self-inductance — $L = N\Phi / I$; the canonical worked examples (solenoid, toroid, coaxial cable).; Mutual inductance — M between two coils; the canonical k coupling coefficient; canonical applications in transformers and induction-coupled chargers (EV wireless charging at intuition).; Lab — measure self- and mutual-inductance of an air-core and an iron-core transformer pair; verify $M^2 \le L_1 L_2$ and $k = M / \sqrt{(L_1 L_2)}$.	CO4
M5	**Time-Varying Fields & Maxwell's Equations** Faraday's law of electromagnetic induction — $\oint E \cdot dl = -\, d\Phi_B / dt$; Lenz's law and the canonical sign convention.; Time-varying fields — the canonical step from static to dynamic electromagnetics.; Convection and conduction currents — $J_{conv} = \rho v$, $J_{cond} = \sigma E$; the canonical Ohm's law in point form.; Transformer EMF and motional EMF — the canonical decomposition; applications in transformer (transformer EMF) and DC machine (motional EMF) operation.; Displacement current — $J_d = \partial D / \partial t$; the canonical Maxwell correction to Ampère's law; the canonical "capacitor-current-without-conduction" insight.; Capacitance of parallel-plate, spherical and coaxial geometries revisited — the canonical engineering-design formulas; canonical coaxial-cable (RG-58, RG-6) characteristic-impedance computation.; Maxwell's equations in point form — $\nabla \cdot D = \rho$; $\nabla \cdot B = 0$; $\nabla \times E = -\partial B / \partial t$; $\nabla \times H = J + \partial D / \partial t$; the canonical 4-equation form.; Maxwell's equations in integral form — the canonical step using Divergence and Stokes' theorems; the canonical step toward wave-equation derivation in Antenna & Wave Propagation.	CO5
M6	**Magnetic Forces, Boundary Conditions & Contemporary Applications** Force on a moving charge — the canonical Lorentz-force law $F = q(E + v \times B)$; applications in mass spectrometry, cyclotron, CRT-display deflection at intuition.; Force on a current element / between current elements — the canonical $F = IL \times B$; applications in motor torque and rail-gun intuition.; Force and torque on a closed current-carrying circuit — the canonical $\tau = m \times B$; magnetic dipole moment $m = NIA$; canonical step toward motor-and-generator analysis.; Magnetic dipole — the canonical analogy to electric dipole; far-field magnetic-field expression.; Boundary conditions — the canonical tangential-E continuity, normal-D discontinuity for electric fields; tangential-H discontinuity (with surface current), normal-B continuity for magnetic fields.; RF, wireless and 5G / mmWave context — the canonical 24 / 28 / 39 / 60 GHz mmWave bands; antenna engineering at intuition; canonical Indian-context 5G rollout (Reliance Jio, Bharti Airtel 2023-2026 deployment).; EM-simulation tools — ANSYS HFSS, CST Studio Suite, Keysight ADS for RF / microwave; canonical use cases (PCB-EMI verification, antenna design, motor cogging-torque simulation).; Contemporary materials and applications — rare-earth magnets (NdFeB, SmCo) and EV-motor / wind-turbine context; SiC / GaN substrates for power electronics; nano-materials for EMI shielding; canonical Indian materials-engineering research (IIT-Madras, IISc, ARCI Hyderabad).	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Elements of Electromagnetics**	Matthew N. O. Sadiku	Oxford University Press	2018 (7th Ed.)
2	**Engineering Electromagnetics**	William H. Hayt Jr. & John A. Buck	McGraw-Hill Education	2019 (9th Ed.)
3	**Field and Wave Electromagnetics**	David K. Cheng	Pearson Education	2014 (2nd Ed.)
4	**Materials Science and Engineering: An Introduction**	William D. Callister Jr. & David G. Rethwisch	Wiley India Pvt. Ltd.	2018 (10th Ed.)
5	**Electromagnetic Fields and Wave Theory**	G. S. N. Raju	Pearson Education	2013 (2nd Ed.)
6	**Fundamentals of Engineering Electromagnetics**	Sunil Bhooshan	Oxford Higher Education	2012
7	**Engineering Electromagnetics**	Nathan Ida	Springer (India) Pvt. Ltd.	2015 (3rd Ed.)

Analog Electronic Circuit Design

25EC1302

L-T-P-S	2-0-2-0	Credits	3	Contact Hrs/Wk	4
Semester	S03	Mode	Regular	Category	PCC
Pre-requisite	DISD				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply BJT and MOSFET small-signal analysis — h-parameter and small-signal models, common-emitter / common-collector / common-base configurations, biasing techniques — to design discrete-transistor amplifiers.	BTL3 · Apply
CO2	Apply operational-amplifier circuit design — ideal vs real op-amp parameters, slew rate, GBW, the canonical configurations (inverting, non-inverting, summing, difference, instrumentation), single-supply biasing — to signal conditioning.	BTL3 · Apply
CO3	Apply active filter design — Sallen-Key low-pass / high-pass, multiple-feedback band-pass, notch (twin-T) filters, Butterworth / Chebyshev / Bessel responses — to specified frequency-domain requirements.	BTL3 · Apply
CO4	Analyze feedback circuits — negative feedback advantages (linearity, bandwidth, noise reduction), feedback topologies (series-shunt, shunt-series, series-series, shunt-shunt), oscillation conditions (Barkhausen) — to design stable amplifiers.	BTL4 · Analyze
CO5	Apply data converters — DAC architectures (R-2R, weighted), ADC architectures (flash, SAR, sigma-delta), sampling theorem, anti-aliasing filters — to mixed-signal design.	BTL3 · Apply
CO6	Analyze and apply modern analog-design tooling — LTspice simulation discipline, KiCad PCB layout for analog (grounding, shielding, layout-induced noise), the modern analog-IC landscape (Analog Devices, Texas Instruments), AI-augmented analog-design workflow.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	–	H	–	H
CO2	H	–	H	–	H
CO3	H	–	H	–	H
CO4	H	H	H	–	H
CO5	H	–	H	–	H
CO6	H	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**BJT and MOSFET Small-Signal Analysis** Recap from BEEC — BJT and MOSFET basics, biasing fundamentals.; Small-signal models — h-parameter model for BJT, hybrid-pi model; the linearization-around-DC-operating-point principle.; Common-emitter amplifier — voltage gain derivation, input/output impedance, frequency response at intuition.; Common-collector (emitter follower) — unity gain, low output impedance; the canonical buffer.; Common-base — high frequency, current buffer; niche use cases.; MOSFET small-signal — gm, ro, the parallel structure to BJT.; Biasing techniques — voltage divider biasing, emitter degeneration, current-source biasing; the stability discussion.; A first lab: simulate a CE amplifier in LTspice; measure gain, input/output impedance; compare against hand calculation.	CO1

	Title & Topics	CO
M2	**Op-Amp Circuit Design** Real op-amp parameters — input offset voltage, input bias current, slew rate, gain-bandwidth product, CMRR, PSRR.; The canonical op-amp ICs — LM741 (legacy teaching), TL072 (modern audio), OPA2134 (precision), AD8602 (low-noise instrumentation); reading datasheets.; Inverting and non-inverting amplifiers revisited (from BEEC) — the closed-loop frequency response.; Summing and difference amplifiers — the canonical signal-mixing and differential-input use cases.; Instrumentation amplifier — the canonical 3-op-amp topology; CMRR and gain trim; pre-packaged ICs (INA826, AD620).; Single-supply op-amp design — biasing inputs to mid-rail; the rail-to-rail input/output op-amp choice.; A lab: design a 100x-gain instrumentation amplifier with INA826 for ECG signals; verify on bench against textbook value; demonstrate CMRR by feeding common-mode noise.	CO2
M3	**Active Filter Design** Filter classifications — low-pass / high-pass / band-pass / band-reject; Butterworth / Chebyshev / Bessel responses.; Sallen-Key topology — second-order low-pass and high-pass; the canonical campus design.; Multiple-feedback (MFB) topology — second-order; lower sensitivity to component tolerances than Sallen-Key.; Higher-order filters — cascading second-order sections; the canonical 4th-order Butterworth.; Notch filters — twin-T topology for 50Hz powerline rejection; the canonical biomedical front-end use.; State-variable filter at intuition — the universal LP/HP/BP topology.; Filter design tools — Texas Instruments' FilterPro, Analog Devices' Filter Wizard; the productivity gain.; A lab: design and simulate a band-pass filter for ECG (0.5Hz - 100Hz) using Sallen-Key low-pass and high-pass cascade; verify in LTspice; build on breadboard; measure frequency response.	CO3
M4	**Feedback and Stability** Negative feedback fundamentals — gain reduction, bandwidth extension, distortion reduction, noise reduction.; Feedback topologies — series-shunt (voltage in, voltage out), shunt-series, series-series, shunt-shunt; the canonical four configurations.; Loop gain analysis — Aβ, the canonical Bode plot of $\|A\beta\|$.; Stability — phase margin, gain margin, the 60-degree phase-margin rule of thumb.; Compensation techniques — dominant-pole, lead-lag; when each fits.; Oscillator design — Barkhausen criterion (Aβ = 1 at the desired frequency); the canonical Wien-bridge oscillator.; Modern oscillator-IC alternatives — DDS chips (AD9833), the contrast with discrete-transistor oscillators.; A lab: simulate a Wien-bridge oscillator in LTspice; observe startup behavior; build on breadboard; measure frequency stability over 5 minutes.	CO4
M5	**Data Converters** DAC architectures — R-2R ladder, weighted-resistor, segmented; the canonical 12-bit and 16-bit DAC types.; ADC architectures — flash (fastest, expensive), SAR (canonical for embedded — STM32 ADC is SAR), sigma-delta (highest resolution, slow).; Sampling theorem — Nyquist criterion; the consequence of violation (aliasing) and the canonical examples.; Anti-aliasing filters — required before any sampling; the canonical first-order RC + second-order Sallen-Key combination.; Quantization noise — the SQNR formula 6.02N + 1.76 dB; what it means for the practical design.; Modern integrated ADCs / DACs — the STM32 12-bit SAR ADC, the canonical industrial 24-bit sigma-delta (ADS1256), the precision DAC alternatives.; A lab: simulate sampling a 1kHz sine wave at 2kHz and 8kHz; observe alias frequency; design and build an anti-alias filter; demonstrate clean sampling with the filter.	CO5
M6	**Modern Analog Design Tooling** LTspice mastery — schematic capture discipline, transient and AC analysis, parametric sweeps, Monte Carlo for tolerance analysis.; KiCad PCB layout for analog — proper grounding (analog-digital ground separation), star routing, shielding, the canonical mistakes.; PCB fabrication for the Indian student — JLCPCB ($2 for 5 boards), PCBPower (Indian house), OSH Park (premium); the canonical workflow.; Modern analog-IC landscape — Analog Devices (precision, RF), Texas Instruments (broadest portfolio), Maxim (now ADI), Linear Technology (now ADI), Microchip; reading parametric search engines on their websites.; Indian semiconductor design centres — TI India (Bengaluru), ADI India (Bangalore + Hyderabad), Cadence India, Synopsys India; the placement-corridor hubs.; AI coding assistants for analog — Claude / GPT-4 for circuit-topology brainstorming, component-value selection; the discipline of always running through LTspice before committing.; When AI assistants fail in analog design — incorrect biasing assumptions, missed loading effects, wrong frequency-compensation choices, and ignoring real-world parasitics; the eye for these only develops by reading lots of human-engineered designs.; A final-week lab: design, simulate, layout, fabricate, assemble, test a complete signal-conditioning front-end (recipe given for one of: ECG, EMG, force-sensor, photodiode); produce a test report.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Microelectronic Circuits**	Adel S. Sedra, Kenneth C. Smith, Tony Chan Carusone & Vincent Gaudet	Oxford University Press	2020 (8th Ed.)
2	**The Art of Electronics**	Paul Horowitz & Winfield Hill	Cambridge University Press	2015 (3rd Ed.)

#	Title	Authors	Publisher	Year
3	**Op Amps for Everyone**	Bruce Carter & Ron Mancini	Newnes	2017 (5th Ed.)
4	**Analog Integrated Circuit Design**	Tony Chan Carusone, David A. Johns & Kenneth W. Martin	Wiley	2011 (2nd Ed.)
5	**Electronic Devices and Circuit Theory**	Robert L. Boylestad & Louis Nashelsky	Pearson	2012 (11th Ed.)
6	**Practical Electronics for Inventors**	Paul Scherz & Simon Monk	McGraw-Hill	2016 (4th Ed.)
7	**KiCad Like a Pro**	Peter Dalmaris	Tech Explorations	2023 (3rd Ed.)

Artificial Intelligence And Machine Learning

25AD2102

L-T-P-S	3-0-2-0	Credits	4	Contact Hrs/Wk	5
Semester	S04	Mode	Regular	Category	ESC
Pre-requisite	PYPG				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the scope and history of AI — symbolic vs statistical paradigms, search as the foundational AI problem-solving framework, the canonical AI sub-fields (search, planning, knowledge representation, learning, perception) — and place modern AI in this lineage.	BTL2 · Understand
CO2	Apply probability and statistics — Bayes' theorem, probability distributions, hypothesis testing, the bias-variance trade-off — as the mathematical underpinning of modern ML.	BTL3 · Apply
CO3	Apply supervised learning — linear and logistic regression, decision trees, random forests, gradient boosting, SVMs, the train/validation/test discipline, cross-validation, evaluation metrics — using scikit-learn at production fluency.	BTL3 · Apply
CO4	Apply unsupervised learning — k-means, hierarchical clustering, DBSCAN, PCA, t-SNE, UMAP — to discover structure in unlabelled data.	BTL3 · Apply
CO5	Apply neural network foundations — perceptron, multilayer perceptron, backpropagation, activation functions, loss functions, the train-loop pattern — using PyTorch on small classification problems.	BTL3 · Apply
CO6	Analyze the modern AI landscape — large language models, generative AI, retrieval-augmented generation, the API-vs-self-hosted trade-offs, prompt engineering basics, AI safety considerations — to make informed choices about applying AI to real problems.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO6	PO8	PSO1
CO1	H	H	–	–	–	–	H
CO2	H	H	–	–	–	–	H
CO3	H	–	H	H	–	–	H
CO4	H	–	H	–	–	–	H
CO5	H	–	H	H	–	–	H
CO6	H	–	H	–	M	M	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**What AI Is — Search and Classical AI** Defining AI: the four canonical perspectives (acting humanly, thinking humanly, acting rationally, thinking rationally); the rational-agent framing as the modern dominant view.; A brief history: 1950s symbolic AI, 1980s expert systems, the AI winters, the 2012 deep-learning revolution, the 2022 LLM moment.; Agents and environments: PEAS framework (Performance, Environment, Actuators, Sensors); fully vs partially observable, deterministic vs stochastic, episodic vs sequential.; Uninformed search: BFS, DFS, uniform-cost search, depth-limited, iterative deepening; complexity comparison.; Informed search: A*, the heuristic function, admissibility and consistency; the canonical 8-puzzle and pathfinding examples.; Adversarial search: minimax, alpha-beta pruning; relevance for game AI (chess, Go before AlphaGo).; Constraint satisfaction at intuition level: backtracking with constraint propagation; the Sudoku and N-queens canonical problems.; A first lab: implement BFS, DFS, A* on a 2D grid pathfinding problem; visualise the explored states; observe heuristic effects.	CO1

	Title & Topics	CO
M2	**Probability, Statistics, and Knowledge Representation** Probability fundamentals revisited: random variables, joint/marginal/conditional distributions; the Bayes theorem stated.; Bayesian thinking: prior, likelihood, posterior; the canonical "disease test" example showing why posterior probability matters.; Common distributions: Bernoulli, Binomial, Categorical, Gaussian, Poisson; when each models reality.; Descriptive statistics revisited at the ML angle: mean, median, std, percentiles; the importance of looking at distributions before modelling.; The bias-variance trade-off introduced: high-bias models underfit; high-variance models overfit; the U-shape of test error.; Hypothesis testing at intuition level: null vs alternative, p-values, Type I vs Type II error; the modern criticisms of NHST.; Knowledge representation at intuition level: propositional logic, first-order logic, ontologies; why classical KR was eclipsed by statistical learning but still matters in domain-heavy AI.; A lab: an EDA notebook on a real Indian dataset (e.g., census or RBI data) — load with pandas, visualise distributions, compute summary stats, identify quality issues.	CO2
M3	**Supervised Learning — Regression and Classification** The supervised learning setup: features X, labels y, the goal of learning a function f: X → y from examples.; Linear regression: ordinary least squares, gradient descent vs the closed-form solution, regularisation (L1=Lasso, L2=Ridge, ElasticNet).; Logistic regression: the binary-classification workhorse; the sigmoid function, log-loss, decision boundary intuition.; Decision trees and random forests: information gain, Gini impurity, bagging, the random-feature trick that makes random forests work.; Gradient boosting: XGBoost / LightGBM / CatBoost as the dominant tabular-ML algorithms in industry today.; Support Vector Machines at intuition level: the maximum-margin idea, the kernel trick, when SVMs win.; The ML lifecycle hygiene: train/validation/test split, k-fold cross-validation, stratified sampling, data leakage as the common silent bug.; Evaluation metrics: MSE/RMSE for regression; accuracy/precision/recall/F1/ROC-AUC for classification; the canonical confusion matrix.; A lab: a complete supervised-learning notebook on the Telco churn dataset — EDA, feature engineering, baseline logistic regression, gradient-boosting tuned with grid search, evaluation, business interpretation.	CO3
M4	**Unsupervised Learning and Dimensionality Reduction** The unsupervised learning setup: features X, no labels; the goal is to discover structure.; k-Means clustering: the algorithm, the k-selection problem (elbow, silhouette), strengths and limits.; Hierarchical clustering: agglomerative vs divisive, dendrograms, linkage criteria.; DBSCAN: density-based clustering; handles arbitrary shapes and noise; the eps + min-samples hyperparameter pair.; Gaussian mixture models at intuition level: soft clustering as probabilistic generalisation of k-means.; Principal Component Analysis (PCA): variance-maximisation directions; eigendecomposition mental model; explained-variance ratio for choosing dimensionality.; t-SNE and UMAP: nonlinear visualisation techniques; how to interpret them honestly (and the canonical mistakes).; Anomaly detection at preview: isolation forest, one-class SVM, autoencoder-based — the techniques that surface in fraud and quality assurance.; A lab: a clustering + dimensionality-reduction notebook on a customer-segmentation dataset — k-means with k tuned, PCA visualisation, UMAP visualisation, business interpretation of clusters.	CO4
M5	**Neural Networks — Foundations** The biological-vs-artificial neuron mental model: dendrites/axons → weights/sums/activation; honest about the analogy's limits.; The perceptron: history (Rosenblatt 1958), the algorithm, the limit (Minsky-Papert XOR proof), the multi-layer fix.; Multilayer perceptron (MLP): forward pass through layers; matrix-multiplication mental model.; Activation functions: sigmoid, tanh, ReLU and its variants (LeakyReLU, GELU, SiLU); why ReLU dominated post-2012.; Loss functions: MSE for regression, cross-entropy for classification; what each penalises.; Backpropagation: chain rule applied through layers; computational-graph mental model.; Training loop pattern: epoch → batch → forward → loss → backward → optimizer.step(); the canonical PyTorch idiom.; A first PyTorch lab: a 3-layer MLP on MNIST/Fashion-MNIST — full train loop, evaluate on test set, plot loss curve, tune learning rate; the bridge to DLAC depth.; Honest framing: deep learning is not a separate subject — it's neural networks at scale (more layers, more data, more compute, more architectural sophistication). The framework landscape: PyTorch dominant, TensorFlow on the way out, JAX in research; choose PyTorch for new work.	CO5

	Title & Topics	CO
M6	**The Modern AI Landscape** The transformer architecture at intuition level: attention as the key innovation; the contrast with CNNs and RNNs (depth lives in DLAC and NLPL).; Pre-training and fine-tuning paradigm: huge unsupervised pre-training, task-specific fine-tuning; the GPT/BERT lineage in 30 seconds.; Large Language Models (LLMs) — what they are: parameter-heavy transformer-based models trained on web-scale text; the capability tier (small models like Phi-3, mid like Llama-3-8B, large like GPT-4 / Claude / Gemini).; Generative AI applications: text generation, code generation, image generation (Stable Diffusion / DALL-E / Midjourney at intuition level); the 2026-current capability frontier.; Retrieval-Augmented Generation (RAG) at intuition level: combining LLM generation with retrieved context from a vector database; why RAG is the dominant production pattern (depth in GALE).; API vs self-hosted: when to use OpenAI/Anthropic/Google APIs vs deploying your own model; the cost, latency, privacy, customisation trade-offs.; Prompt engineering basics: zero-shot, few-shot, chain-of-thought; the discipline of testing prompts as you would test code.; AI safety and ethics in practice: bias, hallucination, privacy, the EU AI Act and India's DPDP Act at preview; the engineer's responsibility.; A lab: use the HuggingFace transformers library to (a) call a pre-trained sentiment classifier on Indian-language tweets, (b) call a small open-source LLM via the pipelines API, (c) build a tiny RAG over 50 PDF documents using sentence-transformers + FAISS; observe what works, where it fails, and the engineer's role in deploying it correctly.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Artificial Intelligence — A Modern Approach**	Stuart Russell & Peter Norvig	Pearson	2021 (4th Ed.)
2	**Pattern Recognition and Machine Learning**	Christopher M. Bishop	Springer	2006
3	**Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow**	Aurélien Géron	O'Reilly	2022 (3rd Ed.)
4	**An Introduction to Statistical Learning — with Applications in Python**	Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani & Jonathan Taylor	Springer	2023
5	**Deep Learning**	Ian Goodfellow, Yoshua Bengio & Aaron Courville	MIT Press	2016
6	**Designing Machine Learning Systems — An Iterative Process for Production-Ready Applications**	Chip Huyen	O'Reilly	2022
7	**AI Engineering — Building Applications with Foundation Models**	Chip Huyen	O'Reilly	2024

Basic Electrical And Electronic Circuits

25EC1101

L-T-P-S	3-0-2-0	Credits	4	Contact Hrs/Wk	5
Semester	S01	Mode	Regular	Category	ESC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply Ohm's law and Kirchhoff's laws — KCL, KVL, mesh analysis, nodal analysis — to analyze DC resistive circuits.	BTL3 · Apply
CO2	Apply network theorems — Thevenin, Norton, superposition, maximum power transfer — to simplify circuit analysis.	BTL3 · Apply
CO3	Apply AC circuit analysis — phasors, impedance, RLC circuits, resonance, power in AC circuits — to single-phase systems.	BTL3 · Apply
CO4	Apply semiconductor device fundamentals — PN diode, BJT, MOSFET — to small-signal and switching circuits at intuition level.	BTL3 · Apply
CO5	Apply operational amplifiers — ideal op-amp, inverting/non-inverting/summing/difference amplifiers, basic filters — to signal-conditioning circuits.	BTL3 · Apply
CO6	Apply digital logic fundamentals and laboratory measurement — Boolean algebra, basic gates, multimeters, oscilloscopes, function generators, modern simulation tools (LTspice / Multisim) — to analyze and design simple circuits.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	H	–	–	H
CO2	H	H	–	–	H
CO3	H	H	–	–	H
CO4	H	–	H	–	H
CO5	H	–	H	–	H
CO6	H	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**DC Circuit Analysis** Charge, current, voltage, power — fundamental quantities and units.; Resistors — color coding, series/parallel combinations, voltage and current divider rules.; Ohm's law and power dissipation — V=IR, P=VI=$I^2R=V^2/R$.; Kirchhoff's Current Law (KCL) — sum of currents at a node equals zero.; Kirchhoff's Voltage Law (KVL) — sum of voltages around a loop equals zero.; Mesh analysis and nodal analysis — systematic methods for circuits with multiple sources.; A first lab: build a 4-node DC circuit on breadboard; measure node voltages with multimeter; verify against hand-calculated KCL/KVL solutions.	CO1
M2	**Network Theorems** Thevenin's theorem — replacing any linear circuit with V_th and R_th; the canonical computation.; Norton's theorem — current-source dual of Thevenin.; Superposition theorem — for linear circuits with multiple sources.; Maximum power transfer theorem — when load resistance equals source impedance.; Source transformations — converting between Thevenin and Norton equivalents.; Star-delta (Y-Δ) transformation — for simplifying resistor networks.; A lab: derive Thevenin equivalent of a 6-resistor 2-source circuit on paper; verify by measurement on breadboard; demonstrate maximum power transfer with a load-resistance sweep.	CO2

	Title & Topics	CO
M3	**AC Circuit Analysis** AC fundamentals — sinusoidal waveforms, frequency, period, peak/RMS/average values.; Phasors — representing sinusoids as complex numbers; the canonical Euler-formula bridge.; Impedance — resistor, capacitor, inductor in the frequency domain; series and parallel combinations.; RLC circuits — series and parallel; the canonical resonance condition.; Power in AC circuits — real, reactive, apparent power; the power factor; why it matters in industrial loads.; Three-phase systems at intuition — Y and Δ connections; balanced loads.; A lab: build series RLC circuit; sweep frequency with function generator; observe resonance peak on oscilloscope; verify resonance frequency $f = 1/(2\pi\sqrt{LC})$.	CO3
M4	**Semiconductor Devices** Semiconductor physics at intuition — intrinsic vs extrinsic, p-type and n-type, the PN junction.; PN junction diode — forward and reverse bias, I-V characteristic, Zener diodes.; Diode applications — half-wave and full-wave rectifiers, voltage regulators, clipping/clamping circuits.; Bipolar Junction Transistor (BJT) — NPN/PNP construction, common-emitter operation, biasing fundamentals.; MOSFET — N-channel/P-channel, enhancement-mode operation, switching applications.; Comparison of BJT vs MOSFET — input impedance, switching speed, when each wins.; A lab: build a half-wave rectifier with capacitive filter on breadboard; observe input vs output waveforms on oscilloscope; characterize BJT in CE configuration.	CO4
M5	**Operational Amplifiers** Ideal op-amp — infinite open-loop gain, infinite input impedance, zero output impedance; the two golden rules.; Negative feedback — virtual short and virtual ground; the foundation for all op-amp circuits.; Inverting amplifier — gain = -Rf/Rin; the canonical configuration.; Non-inverting amplifier — gain = 1 + Rf/Rin; voltage follower as a special case.; Summing and difference amplifiers — the standard signal-conditioning building blocks.; Active filters — first-order low-pass and high-pass; cutoff frequency; the connection to AECD downstream.; A lab: design and build inverting amplifier with gain -10 using LM741 / TL072 op-amp; verify gain by measurement; observe DC offset and slew-rate effects.	CO5
M6	**Digital Logic and Modern Tooling** Number systems — binary, octal, hexadecimal; conversions; signed-number representations (two's complement).; Boolean algebra — basic identities, De Morgan's laws, Karnaugh maps for simplification.; Logic gates — AND, OR, NOT, NAND, NOR, XOR, XNOR; truth tables and gate-level implementations.; Combinational circuits — adders, multiplexers, decoders, encoders; the building-block view.; Sequential circuits at intuition — flip-flops, latches; the bridge to DISD downstream.; Modern simulation tooling — LTspice (free, dominant analog simulator), Multisim, KiCad with ngspice; the productivity gain over breadboard-only.; AI-augmented circuit design — Claude / ChatGPT for circuit-design brainstorming and component selection; the discipline of always verifying with actual simulation or hardware before trusting the suggestion.; A final-week lab: design a 1-bit full adder in LTspice using transistor-level gates; verify truth table by simulation; build it on breadboard; demonstrate to the panel.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Fundamentals of Electric Circuits**	Charles K. Alexander & Matthew N. O. Sadiku	McGraw-Hill	2020 (7th Ed.)
2	**Microelectronic Circuits**	Adel S. Sedra, Kenneth C. Smith, Tony Chan Carusone & Vincent Gaudet	Oxford University Press	2020 (8th Ed.)
3	**The Art of Electronics**	Paul Horowitz & Winfield Hill	Cambridge University Press	2015 (3rd Ed.)
4	**Digital Design — With an Introduction to the Verilog HDL, VHDL, and SystemVerilog**	M. Morris Mano & Michael D. Ciletti	Pearson	2018 (6th Ed.)
5	**Engineering Circuit Analysis**	William H. Hayt, Jack E. Kemmerly & Steven M. Durbin	McGraw-Hill	2018 (9th Ed.)
6	**Op-Amps and Linear Integrated Circuits**	Ramakant A. Gayakwad	Pearson	2015 (4th Ed.)
7	**Practical Electronics for Inventors**	Paul Scherz & Simon Monk	McGraw-Hill	2016 (4th Ed.)

Communication Skills For Engineers

25UC1204

L-T-P-S	0-0-4-0	Credits	2	Contact Hrs/Wk	4
Semester	S02	Mode	Regular	Category	HSS
Pre-requisite	LSFE				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the communication process, models, barriers, and non-verbal cues.	BTL2 · Understand
CO2	Apply professional written communication — emails, reports, and correspondence.	BTL3 · Apply
CO3	Apply presentation and public-speaking skills with confidence and impact.	BTL3 · Apply
CO4	Apply interpersonal, team, and conflict-handling communication.	BTL3 · Apply
CO5	Apply employability communication — resumes, interviews, and group discussions.	BTL3 · Apply
CO6	Analyze and practise communication in digital/AI-mediated, global settings.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	–	M	**H**	–	M	–
CO2	–	M	**H**	–	M	–
CO3	–	**H**	**H**	–	M	–
CO4	–	**H**	**H**	–	M	–
CO5	–	–	**H**	**H**	–	M
CO6	M	–	**H**	–	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Communication Fundamentals** The communication process and models.; Verbal and non-verbal communication.; Barriers and how to overcome them.; Listening as a communication skill.; Feedback and clarity.; Tone, register, and audience.; Body language and paralanguage.; Cross-cultural awareness. **Labs:** Non-verbal-communication observation lab with structured feedback.; Listening-and-feedback role-play with a communication-barrier scenario. **Case Study:** Case Study — a communication breakdown on a project team: analyse a real case where barriers/assumptions derailed a project, and identify the fixes.	CO1
M2	**Professional & Workplace Communication** Email etiquette and structure.; Business correspondence (letters, memos).; Technical reports and documentation.; Meeting communication and minutes.; Netiquette and tone online.; Concise, reader-focused writing.; Cross-cultural workplace communication.; Common professional-writing pitfalls. **Labs:** Draft a professional report/memo for a workplace scenario and peer-review it.; Email-tone lab: rewrite the same message for three different audiences. **Case Study:** Case Study — a real escalation email thread: analyse how tone and structure affected the outcome, then rewrite the key message for a better result.	CO2
M3	**Oral Communication & Presentations** Public-speaking fundamentals.; Structuring a presentation.; Designing effective visual aids.; Voice, pace, and body language.; Engaging an audience.; Handling questions and nerves.; Storytelling in technical talks.; Virtual-presentation skills. **Labs:** Deliver a 5-minute technical presentation with slides and receive rubric feedback.; Impromptu-speaking lab on engineering topics with peer evaluation. **Case Study:** Case Study — a famous engineering/product pitch or TED talk: analyse what made the delivery effective (or not) and extract reusable techniques.	CO3

	Title & Topics	CO
M4	**Interpersonal & Team Communication** Group discussion skills.; Teamwork and collaborative communication.; Assertiveness vs aggression vs passivity.; Conflict resolution.; Negotiation basics.; Emotional intelligence in communication.; Giving and receiving feedback.; Leadership communication. **Labs:** Structured group-discussion lab with observer feedback.; Conflict-resolution / negotiation role-play with reflection. **Case Study:** Case Study — a team conflict in a real project: analyse the interpersonal dynamics and design a communication strategy to resolve it.	CO4
M5	**Employability Communication** Resume and cover-letter writing.; LinkedIn and professional profiles.; Interview communication (HR + technical).; Group discussions for selection.; Telephonic and video interviews.; Personal branding and elevator pitch.; Answering behavioural questions (STAR).; Communicating under pressure. **Labs:** Build a resume + elevator pitch and present it for feedback.; Mock interview + group discussion lab with structured evaluation. **Case Study:** Case Study — a mock-placement scenario: analyse strong vs weak interview/GD responses from real placement contexts and what distinguishes them.	CO5
M6	**Communication in the Digital & AI Age** Virtual and remote communication.; Collaboration tools and async communication.; Data storytelling and visual communication.; AI communication tools — use and misuse.; Maintaining authenticity and voice.; Managing digital presence and reputation.; Ethics and credibility online.; Communicating across global teams. **Labs:** Data-storytelling lab: turn a dataset into a clear one-slide message.; Async + AI-assisted communication lab with an ethics reflection. **Case Study:** Case Study — remote-team communication: analyse a real distributed-team case (e.g., an IT services project) and how communication practices made or broke it.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Technical Communication: Principles and Practice, 3rd Ed.**	Meenakshi Raman & Sangeeta Sharma	Oxford University Press	2015
2	**Business Communication, 2nd Ed.**	Meenakshi Raman & Prakash Singh	Oxford University Press	2012
3	**Crucial Conversations: Tools for Talking When Stakes Are High, 2nd Ed.**	Kerry Patterson, Joseph Grenny, Ron McMillan & Al Switzler	McGraw-Hill	2011
4	**The 7 Habits of Highly Effective People**	Stephen R. Covey	Free Press	1989
5	**Soft Skills: An Integrated Approach to Maximise Personality**	Gajendra Singh Chauhan & Sangeeta Sharma	Wiley	2016
6	**English for Technical Communication**	Aysha Viswamohan	Tata McGraw-Hill	2008
7	**Talk Like TED: The 9 Public-Speaking Secrets of the World's Top Minds**	Carmine Gallo	St. Martin's Press	2014

Computer Organization And Architecture

25EC1204

L-T-P-S	3-0-0-0	**Credits**	3	**Contact Hrs/Wk**	3
Semester	S02	**Mode**	Regular	**Category**	PCC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the von Neumann and Harvard architecture models, the role of CPU/memory/I-O, the canonical instruction-execution cycle, and the levels-of-abstraction view (gates → datapath → ISA → assembly → high-level language).	BTL2 · Understand
CO2	Apply instruction-set architecture concepts — RISC-V as the canonical teaching ISA, instruction formats (R/I/S/B/U/J), addressing modes, assembly-language fundamentals — to read and write small assembly programs.	BTL3 · Apply
CO3	Analyze the single-cycle and pipelined datapath — register file, ALU, control signals, the 5-stage pipeline (IF/ID/EX/MEM/WB), pipeline hazards (data, control, structural), forwarding, branch prediction at intuition.	BTL4 · Analyze
CO4	Analyze the memory hierarchy — cache organization (direct-mapped, set-associative, fully-associative), write policies, replacement policies, virtual memory and TLB at intuition, the cache-conscious-programming connection.	BTL4 · Analyze
CO5	Apply I/O fundamentals — programmed I/O, interrupt-driven I/O, DMA, bus arbitration, the canonical peripheral interfaces (USB, PCIe at intuition, storage interfaces) — to reason about I/O-bound performance.	BTL3 · Apply
CO6	Analyze modern architecture extensions — multicore and SMT/hyperthreading at intuition, SIMD vector instructions (AVX/NEON), GPU architecture at intuition, the Indian semiconductor design ecosystem context (Bengaluru, Hyderabad, Noida design centres).	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	H	–	–	H
CO2	H	–	H	–	H
CO3	H	H	H	–	H
CO4	H	H	H	H	H
CO5	H	–	H	–	H
CO6	H	H	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Architecture Foundations** The levels-of-abstraction picture — transistors → gates → registers → datapath → ISA → assembly → high-level language → application; why each layer hides complexity below.; The von Neumann architecture — single memory holding both code and data; the canonical limitations and the canonical workarounds.; The Harvard architecture — separate code and data memories; why most modern CPUs are "modified Harvard" (split L1 caches).; The instruction-execution cycle — fetch, decode, execute, memory-access, write-back; the canonical workhorse.; Performance metrics — clock frequency, IPC, CPU time = (instructions × CPI) / clock-rate; Amdahl's law at intuition.; The Indian semiconductor context — TSMC fab dependency, India's design-centre strength (200K+ engineers), the policy push toward local manufacturing (Tata-PSMC, Micron Sanand).; A first lab: trace the execution of one instruction through the von Neumann cycle on paper; verify against a RISC-V simulator (Venus or Ripes).	CO1

	Title & Topics	CO
M2	**Instruction-Set Architecture (RISC-V)** Why RISC-V — open ISA replacing MIPS as the canonical teaching choice; the rapid commercial adoption (SiFive, Western Digital, Alibaba T-Head); India's SHAKTI processor at IIT Madras.; RISC-V base ISA RV32I — 47 instructions across R/I/S/B/U/J formats.; Registers — 32 general-purpose registers, the calling convention (a0-a7 for arguments, t0-t6 for temporaries), the canonical zero register x0.; Instruction formats — bit-by-bit layout of R/I/S/B/U/J formats; how the decoder identifies each.; Addressing modes — register, immediate, displacement (the canonical RISC three).; Assembly programming basics — load/store, arithmetic, branches, function calls (jal/jalr); writing a small assembly program.; A lab: write and run on the Venus or Ripes RISC-V simulator a small assembly program (e.g., compute factorial recursively); inspect register and memory state at each step.	CO2
M3	**Datapath, Control, Pipelining** Single-cycle datapath — register file, ALU, multiplexers, the canonical RISC-V 5-stage layout (IF/ID/EX/MEM/WB).; Control signals — derived from instruction-decode; the truth table approach.; The single-cycle limitation — clock period bound by the slowest instruction; why pipelining wins.; Pipelining — overlapping the 5 stages; ideal speedup and the realistic gap due to hazards.; Data hazards — RAW (read-after-write) as the dominant kind; forwarding/bypassing as the canonical hardware solution.; Control hazards — branches; branch prediction at intuition (static, dynamic, modern neural); the cost of mispredictions.; Structural hazards — when hardware resources double-book; why split I/D L1 caches help.; A lab: trace a 5-instruction sequence through the 5-stage pipeline on paper; identify the hazards; show how forwarding resolves them.	CO3
M4	**Memory Hierarchy** The memory wall — CPU speed grew ~50%/year, DRAM ~7%/year; caches as the canonical fix.; Cache fundamentals — locality (temporal, spatial), block/line size, hit and miss penalties.; Cache organization — direct-mapped, set-associative (2-way, 4-way, 8-way, ...), fully-associative; trade-offs.; Cache replacement policies — LRU as the gold standard, random / FIFO as cheap alternatives; modern hardware approximations of LRU.; Cache write policies — write-through vs write-back, write-allocate vs no-write-allocate.; Multi-level caches — L1/L2/L3 hierarchy in modern CPUs; inclusive vs exclusive caches at intuition.; Virtual memory at intuition — pages, page tables, TLB; the integration with cache (PIPT vs VIVT).; Cache-conscious programming — array-of-structs vs struct-of-arrays for SIMD-friendly code, blocking/tiling for matrix multiply, the connection to the 100x speedup that motivated the BLAS / cuBLAS world.; A lab: in C, implement matrix-multiply 3 ways — naive triple-loop, loop-interchange for cache, block-tiled; measure each on 1024×1024 matrices; observe the order-of-magnitude difference.	CO4
M5	**Input/Output Systems** Programmed I/O — CPU-driven, polling-based; the simplicity and the wasted cycles.; Interrupt-driven I/O — peripherals notify when ready; the modern default for keyboards, mice, storage.; DMA (Direct Memory Access) — peripheral talks directly to memory, bypassing CPU; canonical for disk and network transfers.; Buses — the historical view (ISA, PCI), the modern point-to-point (PCIe); arbitration and burst transfers.; Storage interfaces — SATA vs NVMe; why NVMe-over-PCIe revolutionized SSD performance.; USB at intuition — host-controller architecture, USB-2/3/4 speed evolution.; Network interface integration — how NICs DMA packets into memory, the canonical kernel-bypass technologies (DPDK, RDMA at preview).; A lab: in Linux, observe interrupt rates with /proc/interrupts during heavy I/O (running a network test); discuss the contributions of each interrupt source.	CO5
M6	**Modern Architecture and the Frontier** Multicore architectures — symmetric multiprocessing (SMP), shared-memory programming model, the cache-coherence problem (MESI protocol at intuition).; Simultaneous multithreading (SMT) / hyperthreading — sharing pipeline resources between threads; when SMT helps and hurts.; SIMD vector instructions — AVX-512 on x86, NEON/SVE on ARM; the SIMD-friendly loop pattern.; GPU architecture at intuition — SIMT model, warp/wavefront, the massive thread count, why GPUs dominate ML; depth in DLAC.; NPUs and AI accelerators — Apple Neural Engine, Google TPU, NVIDIA Tensor cores; the canonical workload (matrix-multiply at low precision); India's indigenous design (Sarvam, AGNIT, AI4Bharat compute).; RISC-V vector extension at intuition — the open ISA path to vector / AI workloads; the Indian SHAKTI program.; The performance frontier in 2026 — the end of Dennard scaling, the move to specialized accelerators, the chiplets architecture (AMD Ryzen, Apple M-series).; A final-week lab: write the same matrix-multiply kernel in 3 forms — scalar C, hand-vectorized with AVX intrinsics, GPU CUDA (provided template); measure throughput on a lab machine; present the speedup story.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Computer Organization and Design — RISC-V Edition**	David A. Patterson & John L. Hennessy	Morgan Kaufmann	2020 (2nd Ed.)

#	Title	Authors	Publisher	Year
2	**Computer Architecture — A Quantitative Approach**	John L. Hennessy & David A. Patterson	Morgan Kaufmann	2019 (6th Ed.)
3	**Computer Organization and Architecture — Designing for Performance**	William Stallings	Pearson	2022 (11th Ed.)
4	**Structured Computer Organization**	Andrew S. Tanenbaum & Todd Austin	Pearson	2012 (6th Ed.)
5	**Digital Design and Computer Architecture — RISC-V Edition**	Sarah L. Harris & David Money Harris	Morgan Kaufmann	2021
6	**The RISC-V Reader — An Open Architecture Atlas**	David Patterson & Andrew Waterman	Strawberry Canyon	2017
7	**Programming Massively Parallel Processors — A Hands-on Approach**	David B. Kirk & Wen-mei W. Hwu	Morgan Kaufmann	2022 (4th Ed.)

Control Systems For Electrical & Energy Applications

25EE3105

L-T-P-S	3-0-2-0	Credits	4	Contact Hrs/Wk	5
Semester	S05	Mode	Regular	Category	PCC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply mathematical modelling (differential equations, transfer functions, block-diagram and signal-flow reduction) to represent electrical and electromechanical systems.	BTL3 · Apply
CO2	Analyze time-domain behaviour — transient and steady-state response, system order, and steady-state error — of first- and second-order control systems.	BTL4 · Analyze
CO3	Analyze stability using the Routh–Hurwitz criterion and the root-locus method, and relate pole locations to performance.	BTL4 · Analyze
CO4	Analyze frequency-domain behaviour via Bode and Nyquist plots, gain and phase margins, to assess relative stability and robustness.	BTL4 · Analyze
CO5	Apply classical controller and compensator design — PID tuning, lead/lag compensation — to meet transient and steady-state specifications for an electrical loop.	BTL3 · Apply
CO6	Apply state-space representation and an introduction to digital control to model and control modern electrical/energy systems, including a simulated case study.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PSO1
CO1	H	H	–	H
CO2	H	H	–	H
CO3	H	H	–	H
CO4	H	H	–	H
CO5	H	–	H	H
CO6	H	–	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**System Modelling and Transfer Functions** Open-loop vs closed-loop control; the feedback principle and why electrical systems need it.; Modelling electrical and mechanical elements; analogies between domains.; The Laplace transform as the control engineer's tool; transfer functions.; Block-diagram algebra and reduction.; Signal-flow graphs and Mason's gain formula.; Modelling a DC motor (armature + mechanical) as a transfer function.; Modelling a DC-DC converter's control-to-output transfer function (topology owned by PEWC).; Standard test inputs: impulse, step, ramp.	CO1
M2	**Time-Domain Analysis** First-order systems: time constant, step response, settling.; Second-order systems: damping ratio, natural frequency, the response families.; Transient specifications: rise time, peak time, overshoot, settling time.; Steady-state error and the system type (0, 1, 2); error constants.; Effect of adding poles and zeros on the response.; Dominant-pole approximation.; Time response of a motor speed loop.; Reading and interpreting simulated step responses.	CO2
M3	**Stability — Routh–Hurwitz and Root Locus** The concept of stability; bounded-input bounded-output.; Characteristic equation and pole locations.; Routh–Hurwitz criterion; special cases (zero rows, zero in first column).; Root-locus construction rules.; Effect of gain on closed-loop pole movement.; Designing gain from the root locus to meet a damping spec.; Stability margins from the root locus.; Root-locus of a motor position loop.	CO3

	Title & Topics	CO
M4	**Frequency-Domain Analysis** Frequency response and the sinusoidal transfer function.; Bode plots: magnitude and phase, asymptotic construction.; Gain margin and phase margin from Bode plots.; The Nyquist stability criterion and encirclements.; Relative stability and robustness interpretation.; Bandwidth and its link to speed of response.; Correlation between time- and frequency-domain specs.; Frequency analysis of a converter voltage loop.	CO4
M5	**Controller and Compensator Design** The PID controller: proportional, integral, derivative actions and their effects.; PID tuning: Ziegler–Nichols and frequency-response-based tuning.; Practical PID: anti-windup, derivative filtering, set-point weighting.; Lead compensator design (phase margin improvement).; Lag compensator design (steady-state error improvement).; Lead-lag compensation.; Designing a PI current/voltage controller for a converter loop.; Verifying the design against transient and steady-state specs.	CO5
M6	**State-Space and Digital Control** State-space representation: state, input, output equations.; State-space from transfer functions and vice versa.; Controllability and observability at concept level.; State-feedback and pole placement (introduction).; Sampling and the z-transform; discrete transfer functions.; Digital PID and discretisation of controllers.; Introduction to modern control in energy systems (grid-following/grid-forming control concept).; A simulated capstone control loop tying the methods together.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Modern Control Engineering**	Katsuhiko Ogata	Pearson	2010 (5th Ed.)
2	**Control Systems Engineering**	Norman S. Nise	Wiley	2019 (8th Ed.)
3	**Automatic Control Systems**	Farid Golnaraghi & Benjamin C. Kuo	Wiley	2017 (10th Ed.)
4	**Feedback Control of Dynamic Systems**	Gene F. Franklin, J. David Powell & Abbas Emami-Naeini	Pearson	2019 (8th Ed.)
5	**Digital Control of Dynamic Systems**	Franklin, Powell & Workman	Addison-Wesley	1997 (3rd Ed.)

Data Structures

25SC1203

L-T-P-S	3-0-2-4	Credits	5	Contact Hrs/Wk	9
Semester	S02	Mode	Regular	Category	ESC
Pre-requisite	PSTC				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply asymptotic notation (Big-O, Big-Theta) and best/average/worst-case reasoning to compare algorithm efficiency at an introductory level.	BTL3 · Apply
CO2	Apply array-based and linked-list data structures to implement stacks, queues, and deques and to solve standard computational problems.	BTL3 · Apply
CO3	Apply hash-based data structures (hash tables, hash sets) using separate chaining and open addressing to design key-value stores.	BTL3 · Apply
CO4	Apply binary tree and binary search tree operations (insert, search, delete, traversal) and an introduction to AVL balancing to organise hierarchical data.	BTL3 · Apply
CO5	Apply elementary and divide-and-conquer sorting algorithms (insertion, selection, merge, quick) and binary search on sorted data.	BTL3 · Apply
CO6	Apply data-structure choice reasoning to pick an appropriate structure for a given problem profile, and produce a working implementation supported by tests.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PSO1
CO1	**H**	M	–	–	–	M
CO2	**H**	–	M	–	–	**H**
CO3	**H**	–	–	–	–	M
CO4	**H**	–	M	–	–	M
CO5	**H**	M	–	–	–	M
CO6	–	M	–	M	L	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Asymptotic Analysis Basics** Counting operations and growth rates; the intuition behind Big-O.; Big-O, Big-Omega, Big-Theta — definitions and standard examples.; Best-case, average-case, and worst-case analysis.; Common growth-rate hierarchy: constant, log, linear, n log n, quadratic, cubic, exponential.; Reading complexity from code: simple loops, nested loops, recursion preview.; Space complexity at the introductory level.; Empirical timing as a complement to theoretical analysis.; A first set of worked examples comparing two algorithms.	CO1
M2	**Arrays and Dynamic Arrays** Static arrays: declaration, allocation, indexing.; Dynamic arrays: geometric resizing, amortised append cost.; Implementing a list-like structure from scratch.; Two-dimensional arrays and matrix operations.; Standard array problems: rotation, reversal, partition.; In-place vs out-of-place transformations.; When arrays are the right answer.; Cache locality at an intuitive level.	CO2
M3	**Linked Lists, Stacks and Queues** Singly linked list: node structure, head pointer, traversal, insert, delete.; Doubly linked list and tail-tracking.; Stack ADT: push, pop, peek; LIFO use cases.; Expression evaluation with stacks; balanced-parentheses checking.; Queue ADT: enqueue, dequeue; FIFO use cases.; Deque and double-ended access.; Implementations: array-backed vs linked-list-backed; trade-offs.; When linked lists are the right answer (and when they are not).	CO2

	Title & Topics	CO
M4	**Hashing** Hash function design: uniformity, determinism.; Hashing integer and string keys.; Collision resolution by separate chaining.; Collision resolution by open addressing — linear probing.; Load factor and resizing.; Hash sets and hash maps as standard library structures.; When to choose a hash table.; Common pitfalls: bad hash functions, equals/hashCode contract violations.	CO3
M5	**Trees and Binary Search Trees** Tree terminology: root, parent, child, leaf, depth, height.; Binary trees: representation in arrays vs as linked nodes.; Tree traversals: pre-order, in-order, post-order, level-order.; Binary search tree (BST): insert, search, delete operations.; BST height and the importance of balance; the unbalanced-BST problem.; AVL trees at an introductory level: balance factor, single and double rotations.; When to choose a BST vs a hash table.; Common operations: minimum, maximum, successor, predecessor.	CO4
M6	**Sorting and Searching** Linear search and binary search; loop-invariant correctness.; Binary search variants: first occurrence, last occurrence.; Insertion sort, selection sort, bubble sort.; Stability and in-place properties.; Merge sort and the merge step.; Quicksort and pivot selection.; Sorting comparison: time, space, stability, in-place.; Standard library sorting and how it differs from textbook quicksort.	CO5

Reference Books

#	Title	Authors	Publisher	Year
1	**Data Structures and Algorithms in Java**	Michael T. Goodrich, Roberto Tamassia & Michael H. Goldwasser	Wiley	2014 (6th Ed.)
2	**Data Structures and Algorithm Analysis in Java**	Mark Allen Weiss	Pearson	2011 (3rd Ed.)
3	**Algorithms**	Robert Sedgewick & Kevin Wayne	Addison-Wesley	2011 (4th Ed.)
4	**Introduction to Algorithms**	Cormen, Leiserson, Rivest & Stein (CLRS)	MIT Press	2022 (4th Ed.)
5	**Data Structures Through Java**	V. V. Muniswamy	I. K. International Publishing House	2009
6	**Classic Data Structures**	D. Samanta	PHI Learning	2009 (2nd Ed.)

Deep Learning With Accelerated Computing

25CS3112

L-T-P-S	2-0-0-4	Credits	3	Contact Hrs/Wk	6
Semester	S05	Mode	Regular	Category	PCC
Pre-requisite	AIML				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply neural-network foundations — perceptrons, multi-layer networks, backpropagation, common activation functions, weight initialization, regularization (dropout, batch norm) — to train classification and regression models.	BTL3 · Apply
CO2	Apply convolutional neural networks — convolution, pooling, the canonical architectures (LeNet, AlexNet, VGG, ResNet at intuition), transfer learning — to image-classification problems.	BTL3 · Apply
CO3	Apply sequence models — RNN, LSTM, GRU at intuition, the transformer attention mechanism, encoder/decoder architectures — to sequence-modeling tasks.	BTL3 · Apply
CO4	Apply GPU-accelerated training — CUDA at intuition, PyTorch GPU programming, mixed-precision training, gradient accumulation, distributed training basics — to scale model training beyond single-GPU memory.	BTL3 · Apply
CO5	Apply inference optimization — quantization (INT8, FP16), pruning at intuition, ONNX as the cross-framework format, TensorRT / Triton serving — to deploy models for production latency.	BTL3 · Apply
CO6	Analyze the modern model-development lifecycle — experiment tracking with Weights and Biases / MLflow, the API-vs-self-hosted decision, foundation models as starting points, the path to LLM fine-tuning depth (NLPL).	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	H	–	–	H
CO2	H	–	H	–	H
CO3	H	–	H	–	H
CO4	H	–	H	H	H
CO5	H	–	H	H	H
CO6	H	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Neural Network Foundations** The perceptron — linear classifier, decision boundary, the limitation (XOR problem); the historical context.; Multi-layer perceptrons — composition of linear-and-nonlinear layers; the universal approximation theorem at intuition.; Backpropagation — chain rule applied to compute gradients of loss w.r.t. all weights; the canonical hand-derivation on a 2-layer network.; Activation functions — sigmoid, tanh, ReLU, GELU, Swish; the canonical choice (ReLU) and when alternatives win.; Loss functions — MSE for regression, cross-entropy for classification; the canonical pairings.; Optimization revisited — SGD, momentum, Adam (from MFAI but applied here); learning-rate schedules.; Regularization — dropout (intuition), batch normalization, weight decay; the bias-variance lens.; A first lab: in PyTorch, train a small MLP on MNIST classification; vary depth, width, dropout rate; observe the train/val curves.	CO1

	Title & Topics	CO
M2	**Convolutional Neural Networks** Why convolution — translation invariance, parameter sharing, local connectivity; the canonical motivation for image data.; Convolution operation — kernels, stride, padding; the connection to image processing classical filters.; Pooling — max pooling vs average pooling; downsampling and the receptive-field expansion.; CNN architectures — LeNet (1989), AlexNet (2012), VGG (2014), ResNet (2015); the pattern of "deeper is better with skip connections".; Transfer learning — using pre-trained ImageNet weights, fine-tuning vs feature-extraction; the canonical PyTorch hub pattern.; Modern vision — Vision Transformers (ViT), CLIP, foundation models for vision; honest framing on when CNN still wins.; A lab: fine-tune a pre-trained ResNet-50 on CIFAR-10; compare against training from scratch; observe the convergence and accuracy difference.	CO2
M3	**Sequence Models and Transformers** Recurrent neural networks (RNN) — the canonical motivation for sequence data; vanilla RNN, vanishing gradients.; LSTM and GRU at intuition — gated alternatives; the canonical use cases (language modeling, time-series).; Attention mechanism — query, key, value; the canonical formula softmax(QK^T/sqrt(d))V (revisited from MFAI).; Self-attention vs cross-attention — the computational structure; the connection to multi-head attention.; The transformer architecture — encoder + decoder; positional embeddings; layer normalization.; Why transformers won — parallelization across sequence positions, scalability to hundreds of millions / billions of parameters.; Foundation models at intuition — GPT, BERT, T5; the pre-train-and-fine-tune paradigm; the bridge to NLPL/GAPE.; A lab: in PyTorch, implement a small transformer from scratch (3-layer, 8-head) for a toy translation task; verify against torch.nn.Transformer.	CO3
M4	**GPU-Accelerated Training** Why GPUs for DL — massive parallelism, matrix-multiply throughput; the SIMT model from COAA recap.; CUDA at intuition — kernels, threads, blocks, grids; the canonical "Hello CUDA" example; depth in DLAC PBL.; PyTorch GPU programming — model.cuda(), tensor.to(device); the canonical patterns and the canonical pitfalls (data on wrong device).; Mixed-precision training — FP16 forward, FP32 master weights; PyTorch torch.cuda.amp; the canonical 2-3x speedup on H100/A100.; Gradient accumulation — simulating larger batch sizes when GPU memory is the limit.; Distributed training basics — data parallel via PyTorch DDP; the cross-GPU all-reduce pattern; the bridge to multi-node training (depth in NLPL).; Profiling — torch.profiler, nvtop, the canonical bottleneck-finding workflow.; A lab: train a CNN on CIFAR-10 once on CPU, once on GPU; measure the speedup; enable mixed precision; measure again.	CO4
M5	**Inference Optimization and Deployment** The training-vs-inference asymmetry — training cares about throughput, inference cares about latency.; Quantization — INT8 and FP16 quantization; post-training quantization vs quantization-aware training; the canonical 2-4x speedup.; Pruning at intuition — removing low-importance weights; structured vs unstructured pruning.; Knowledge distillation at intuition — student model learning from teacher; the canonical ~10x parameter reduction.; ONNX — Open Neural Network Exchange; the canonical cross-framework deployment format.; TensorRT and Triton — NVIDIA's production-grade inference stack; the canonical deployment for low-latency requirements.; Serving frameworks — TorchServe, TF Serving, vLLM (LLM-specific); the contemporary 2026 landscape.; A lab: take a trained ResNet-50; convert to ONNX; deploy via Triton in a Docker container; measure latency vs PyTorch eager mode.	CO5
M6	**Modern Model Development Lifecycle** Experiment tracking — Weights and Biases (W&B), MLflow, Neptune; the canonical "version control for ML experiments" stack.; Hyperparameter optimization — Bayesian optimization, Optuna, the canonical W&B sweeps integration.; Foundation models as starting points — Hugging Face Hub as the model-zoo; the from-scratch-vs-fine-tune-vs-API decision.; API-vs-self-hosted decision — when GPT-4 / Claude / Gemini APIs win; when self-hosted Llama / Qwen / Mistral wins; the cost / latency / privacy / customization matrix.; The path forward to LLM depth — NLPL covers from-scratch transformer training and vLLM serving; GAPE covers the full GenAI ecosystem at survey depth; AAAE covers agents.; India's AI compute landscape — Yotta NM1, AI4Bharat compute, IndiaAI mission GPU procurement; the implications for Indian AI engineers.; AI-augmented ML development — Claude / Copilot / Cursor for ML code; the discipline of running every generated training loop through profiling.; A final-week lab: train a small image classifier on a real Indian dataset (e.g., Indian crop disease, Indian sign language); track in W&B; quantize for deployment; deploy a demo via Streamlit / Gradio.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Deep Learning**	Ian Goodfellow, Yoshua Bengio & Aaron Courville	MIT Press	2016
2	**Dive into Deep Learning**	Aston Zhang, Zachary C. Lipton, Mu Li & Alexander J. Smola	Cambridge University Press	2023
3	**Deep Learning with PyTorch**	Eli Stevens, Luca Antiga & Thomas Viehmann	Manning	2020

#	Title	Authors	Publisher	Year
4	**Neural Networks and Deep Learning — A Textbook**	Charu C. Aggarwal	Springer	2023 (2nd Ed.)
5	**Programming Massively Parallel Processors — A Hands-on Approach**	David B. Kirk & Wen-mei W. Hwu	Morgan Kaufmann	2022 (4th Ed.)
6	**Designing Machine Learning Systems — An Iterative Process for Production-Ready Applications**	Chip Huyen	O'Reilly	2022
7	**Efficient Deep Learning — Fast and Accurate Inference**	Gaurav Menghani	Manning	2024

Design Thinking For Innovation

25UC1203

L-T-P-S	0-0-4-0	Credits	2	Contact Hrs/Wk	4
Semester	S01	Mode	Regular	Category	HSS
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand design-thinking principles, mindsets, and the innovation process.	BTL2 · Understand
CO2	Apply empathy methods to understand users and uncover real needs.	BTL3 · Apply
CO3	Analyze and frame problems into actionable, human-centred definitions.	BTL4 · Analyze
CO4	Apply ideation techniques to generate and select innovative solutions.	BTL3 · Apply
CO5	Apply prototyping methods to make ideas tangible and testable.	BTL3 · Apply
CO6	Analyze test feedback to iterate, and apply AI to amplify design thinking.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO2	PO3	PO5	PO6	PO9	PSO1	PSO2
CO1	–	**H**	–	M	–	M	–
CO2	–	–	–	**H**	M	M	–
CO3	**H**	**H**	–	–	–	M	–
CO4	–	**H**	–	–	**H**	–	M
CO5	–	**H**	M	–	–	–	M
CO6	–	**H**	**H**	–	–	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Introduction to Design Thinking** What innovation is and isn't.; Human-centred design philosophy.; The design-thinking mindset.; The double-diamond and 5-stage models.; Design thinking vs traditional problem-solving.; Where design thinking applies.; Famous design-thinking successes.; Roles, teams, and the creative process. **Labs:** Design-thinking warm-up lab: redesign an everyday object for a specific user.; Map a chosen problem onto the double-diamond and present the framing. **Case Study:** Case Study — a human-centred breakthrough (e.g., the Embrace infant warmer / a frugal Indian innovation): analyse how empathy-led design solved a problem cheaper and better.	CO1
M2	**Empathize** User research fundamentals.; Conducting user interviews.; Observation and contextual inquiry.; Empathy maps.; Personas.; User journey maps.; Identifying latent needs.; Synthesising qualitative insight. **Labs:** Conduct and synthesise two user interviews into an empathy map.; Build a persona + journey map for a real user of a chosen service. **Case Study:** Case Study — empathy in practice (a product redesigned after watching real users, e.g., an ATM/UPI flow for first-time users): analyse what observation revealed that surveys missed.	CO2
M3	**Define** From insights to problem statements.; Point-of-view (POV) statements.; 'How Might We' questions.; Reframing the problem.; Prioritising needs.; Defining design challenges and scope.; Success criteria.; Avoiding solution bias. **Labs:** Convert empathy findings into POV + 'How Might We' statements.; Problem-reframing lab: generate three alternative framings of one challenge. **Case Study:** Case Study — problem reframing that changed everything (e.g., 'faster elevators' → 'less boring wait'): analyse how redefining the problem unlocked the real solution.	CO3

	Title & Topics	CO
M4	**Ideate** Principles of effective brainstorming.; Divergent vs convergent thinking.; Ideation techniques (SCAMPER, mind-mapping).; Lateral and analogical thinking.; Building creative confidence.; Idea clustering and selection.; Overcoming fixation and bias.; Prioritising ideas to prototype. **Labs:** Facilitated brainstorm lab generating 50+ ideas, then cluster and select.; SCAMPER lab to transform an existing product into new concepts. **Case Study:** Case Study — an ideation-driven product (a startup's pivot from a brainstormed insight): analyse how structured ideation produced a winning concept.	CO4
M5	**Prototype** Why prototype — thinking with your hands.; Fidelity — low to high.; Paper and physical prototyping.; Digital and clickable prototypes.; Storyboarding and scenarios.; Minimum viable prototypes.; Rapid, cheap iteration.; Communicating a prototype. **Labs:** Build a low-fidelity paper prototype of a solution concept.; Create a clickable/digital prototype and a storyboard of its use. **Case Study:** Case Study — prototyping that saved money (a company that tested a cheap prototype before building): analyse how early prototyping avoided a costly failure.	CO5
M6	**Test, Iterate & AI-Amplified Design** Usability testing basics.; Gathering and interpreting feedback.; Iteration and the build-measure-learn loop.; Design sprints.; Generative AI for ideation and prototyping.; AI-assisted user research and synthesis.; Human judgement vs AI in design.; Scaling and implementing innovation. **Labs:** Run a usability test on your prototype and log iteration insights.; AI-amplified design lab: use generative AI to expand ideas/prototypes, then critique. **Case Study:** Case Study — AI in the design process (a team using generative AI to prototype faster): analyse where AI accelerated design thinking and where human empathy stayed essential.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Change by Design: How Design Thinking Transforms Organizations and Inspires Innovation**	Tim Brown	Harper Business	2009
2	**The Design of Everyday Things, Revised Ed.**	Don Norman	Basic Books	2013
3	**Designing for Growth: A Design Thinking Tool Kit for Managers**	Jeanne Liedtka & Tim Ogilvie	Columbia Business School Publishing	2011
4	**Sprint: How to Solve Big Problems and Test New Ideas in Just Five Days**	Jake Knapp, John Zeratsky & Braden Kowitz	Simon & Schuster	2016
5	**Creative Confidence: Unleashing the Creative Potential Within Us All**	Tom Kelley & David Kelley	Crown Business	2013
6	**The Field Guide to Human-Centered Design**	IDEO.org	IDEO.org	2015
7	**This Is Service Design Thinking**	Marc Stickdorn & Jakob Schneider	Wiley	2011

Design Tools Workshop

25ME1103

L-T-P-S	0-0-4-0	Credits	2	Contact Hrs/Wk	4
Semester	S02	Mode	Regular	Category	ESC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply branch-relevant computer-aided design fundamentals — UI/UX wireframing for CSE-family, electrical/electronic schematics for ECE/EEE, mechanical/civil CAD for MECH/CIVIL, lab-notebook + bioinformatics workflow design for BIOTECH.	BTL3 · Apply
CO2	Apply version control with Git/GitHub — clone, add, commit, push, pull, branch, merge, pull requests — to track design and code artifacts collaboratively.	BTL3 · Apply
CO3	Apply design documentation tooling — Markdown for technical docs, draw.io / Excalidraw for diagrams, Notion / Obsidian for knowledge management — to communicate engineering work clearly.	BTL3 · Apply
CO4	Apply AI-augmented design tools — Claude / ChatGPT for design brainstorming and review, Cursor / Copilot for code design, Midjourney / DALL-E at intuition for visual mockups, Figma AI features — to accelerate design workflows.	BTL3 · Apply
CO5	Apply standard engineering documentation conventions — UML for CSE-family software, ER diagrams for data modeling, schematic-symbol standards for ECE/EEE, ISO/IS dimensioning for MECH/CIVIL — to produce professional design artifacts.	BTL3 · Apply
CO6	Analyze and present a complete branch-relevant design portfolio — at least 4 design artifacts produced with at least 4 distinct tools, version-controlled, documented, peer-reviewed.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO3	PO5	PO9	PO10	PSO1
CO1	H	H	H	–	–	H
CO2	H	–	H	M	–	H
CO3	H	–	–	M	H	H
CO4	H	–	H	–	–	H
CO5	H	–	–	M	H	H
CO6	H	H	H	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Branch-Relevant Design Tool 1** CSE/CSIT/AIDS/IOT branches: Figma — UI wireframing, components, prototyping; the modern UX-design canonical tool.; ECE/EEE branches: AutoCAD Electrical — schematic capture, the canonical electrical-symbol standards.; MECH/CIVIL branches: AutoCAD 2D drafting — drawing setup, layers, dimensioning, the IS/ISO standards.; BIOTECH branch: ELN (Electronic Lab Notebook) tools — Benchling, LabArchives; the modern alternative to paper lab notebooks.; Lab work: build the first design artifact in the chosen tool — for CSE that's a Figma wireframe of a small app; for MECH a 2D mechanical part drawing; for BT a documented protocol in Benchling.	CO1
M2	**Version Control with Git and GitHub** Why version control — preserving history, enabling collaboration, the canonical "I broke it, can I get back?" use case.; Git fundamentals — clone, add, commit, push, pull; the staging-area mental model.; Branching and merging — feature branches, merge vs rebase at intuition; resolving merge conflicts.; GitHub workflow — fork, pull request, code review, CI/CD via GitHub Actions at preview level.; Documentation in repos — README.md, LICENSE, CONTRIBUTING.md; what a well-maintained repo looks like.; Lab work: every student creates a GitHub portfolio repo; commits the M1 artifact with proper README; submits a PR to a course-shared repo for peer review.	CO2

	Title & Topics	CO
M3	**Documentation and Diagramming Tools** Markdown — syntax, the canonical use cases (READMEs, wikis, academic notes, blog posts).; Diagramming — draw.io / Excalidraw / Mermaid for architecture diagrams, flow charts, sequence diagrams.; Knowledge management tools — Notion, Obsidian, Logseq; the personal-knowledge-graph mental model.; Branch-specific documentation conventions — UML for CSE, ER for data work, schematic for ECE/EEE, drawing-sheet for MECH/CIVIL, protocol-and-results for BT.; Lab work: every student produces a documentation-only artifact for a previous-module deliverable — a Markdown README + one diagram in draw.io / Mermaid.	CO3
M4	**AI-Augmented Design Tools** Claude / ChatGPT for design brainstorming — generating UX flows, wireframe ideas, schematic-design alternatives, lab-protocol variations.; Code-design assistance via Cursor / Copilot — function signatures, boilerplate, test stubs.; Visual mockup generation — Midjourney, DALL-E, Stable Diffusion for design mood boards (CSE design students); architectural concept sketches (CIVIL).; Figma AI features (CSE-family) — auto-layout suggestions, content-fill, component-suggestion.; The discipline of treating AI output as a draft — verification, refinement, rejection of bad outputs.; Lab work: take the M1 artifact, regenerate or significantly improve it using AI tools; document the prompts used and the changes accepted/rejected; reflect in writing on what the AI got wrong.	CO4
M5	**Branch-Relevant Design Tool 2 (Advanced)** CSE/CSIT/AIDS/IOT branches: Penpot or Figma at intermediate depth — components, design tokens, accessibility-aware design.; ECE/EEE branches: KiCad / Multisim — schematic-to-PCB workflow at intuition; circuit simulation.; MECH branch: Fusion 360 or SolidWorks basics — 3D parametric modeling, the canonical part-modeling workflow.; CIVIL branch: AutoCAD 3D + introduction to BIM (Revit at intuition); structural drawings.; BIOTECH branch: Benchling sequence design + Snapgene; the canonical molecular-biology design tooling.; Lab work: build the second design artifact in the advanced tool; iterative refinement against feedback; commit to GitHub portfolio.	CO5
M6	**Portfolio Assembly and Peer Review** Portfolio structure — at least 4 artifacts spanning at least 4 distinct tools; the README that frames each artifact.; Peer review process — every student reviews 2 peers' portfolios; constructive feedback discipline.; Self-reflection — what tool clicked vs didn't; the path forward in subsequent design-heavy courses (FOWD/FDFU for CSE-family; ENGP for MECH/CIVIL; BFSA for BT).; Presenting design work — the canonical portfolio website (free templates) for CSE-family; the canonical drawing-set for MECH/CIVIL.; Final assembly week — incorporating peer feedback; final commit to GitHub; final live demo to the panel.; Lab work: complete portfolio assembly + submit final repository; 15-minute live walk-through of two artifacts of the panel's choosing.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Refactoring UI**	Adam Wathan & Steve Schoger	Self-published	2018
2	**Don't Make Me Think — A Common Sense Approach to Web Usability**	Steve Krug	New Riders	2014 (3rd Ed.)
3	**Pro Git**	Scott Chacon & Ben Straub	Apress (free online)	2014 (2nd Ed.)
4	**Engineering Drawing — Plane and Solid Geometry**	N. D. Bhatt	Charotar Publishing	2024 (54th Ed.)
5	**Mastering AutoCAD 2024 and AutoCAD LT 2024**	George Omura & Brian C. Benton	Sybex	2023
6	**KiCad Like a Pro — Mastering the World's Favorite Open-Source PCB Design Tool**	Peter Dalmaris	Tech Explorations	2023 (3rd Ed.)
7	**Tools for Thought — How AI Is Reshaping Design Workflows**	Various authors (community-curated)	Online (free)	2024+

Digital System Design

25EC1102

L-T-P-S	3-0-2-0	Credits	4	Contact Hrs/Wk	5
Semester	S01	Mode	Regular	Category	ESC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply Boolean algebra and number systems — binary/octal/hex, signed representations (sign-magnitude, 1's complement, 2's complement), Boolean identities, De Morgan's laws, K-map minimization — to simplify digital expressions.	BTL3 · Apply
CO2	Apply combinational logic design — gates, encoders, decoders, multiplexers, demultiplexers, adders (half/full, carry-look-ahead), comparators, ALU at intuition — to build practical combinational circuits.	BTL3 · Apply
CO3	Analyze sequential logic design — latches, flip-flops (D, T, JK, SR), the storage element distinction, clock and timing, setup/hold time at intuition.	BTL4 · Analyze
CO4	Apply finite state machine design — Moore vs Mealy, state diagrams, state-transition tables, state encoding, the canonical FSM design flow — to specify and implement controllers.	BTL3 · Apply
CO5	Apply Verilog/SystemVerilog HDL — modules, signals, blocking vs non-blocking assignments, always blocks, parameterization, testbenches — to design and simulate digital systems.	BTL3 · Apply
CO6	Apply FPGA prototyping fundamentals and the modern HDL toolchain — Vivado / Quartus, simulation (Verilator, Icarus), synthesis at intuition, the open-source FPGA toolchain (Yosys + nextpnr) — to deploy a small design on real hardware.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	H	–	–	H
CO2	H	–	H	–	H
CO3	H	H	H	–	H
CO4	H	–	H	–	H
CO5	H	–	H	H	H
CO6	H	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Boolean Algebra and Number Systems** Number systems — binary, octal, hexadecimal; conversions in both directions; the canonical exam questions.; Signed-number representations — sign-magnitude, 1's complement, 2's complement; why 2's complement won.; Floating-point representation at intuition — IEEE 754 single and double precision; the connection to PSTC float behavior.; Boolean algebra — postulates, identities (AND/OR/NOT laws), De Morgan's laws; canonical algebraic simplification.; Standard forms — sum-of-products (SOP), product-of-sums (POS), canonical SOP/POS, minterms and maxterms.; K-map minimization — 2/3/4-variable maps, prime implicants, essential prime implicants; the canonical procedure.; Quine-McCluskey method at intuition — the systematic algorithm for variable counts where K-maps don't scale.; A first lab: minimize 5 SOP expressions via K-map by hand; verify with an online K-map tool; build one minimized expression with 7400-series ICs on breadboard.	CO1
M2	**Combinational Logic Design** Logic gates — AND, OR, NOT, NAND, NOR, XOR, XNOR; the universal-gate property of NAND and NOR.; Encoders and decoders — 8-to-3 priority encoder, 3-to-8 decoder; the canonical use cases.; Multiplexers and demultiplexers — 2-to-1, 4-to-1, 8-to-1 muxes; tree structures for larger.; Adders — half adder, full adder, ripple-carry adder, carry-look-ahead at intuition; the propagation-delay implication.; Subtractors and comparators — 2's-complement adder/subtractor, magnitude comparator.; ALU design at intuition — composing arithmetic and logic units with mux-controlled output; the bridge to COAA.; Hazards in combinational circuits — static and dynamic hazards; mitigation techniques.; A lab: design and simulate a 4-bit ripple-carry adder in Verilog using gate-level constructs; write a testbench that exercises all 256 input combinations.	CO2

	Title & Topics	CO
M3	**Sequential Logic Design** Latches vs flip-flops — the level-triggered vs edge-triggered distinction; why edge-triggered is the modern default.; D flip-flop — the canonical workhorse; truth table; the master-slave structure.; T, JK, SR flip-flops — characteristic equations; conversions among them.; Setup time, hold time, clock-to-Q delay at intuition — the timing-budget mental model.; Registers — parallel-load, serial-in / serial-out (shift registers), parallel-in / parallel-out.; Counters — synchronous vs asynchronous, up/down counters, mod-N counters; the canonical applications.; Memory elements at intuition — SRAM cell (6T) and DRAM cell (1T1C); the distinction from flip-flops; the bridge to COAA memory hierarchy.; A lab: design and simulate a 4-bit synchronous counter in Verilog; verify on Vivado simulator; deploy to a Basys-3 / Nexys A7 / iCE40 board if available.	CO3
M4	**Finite State Machines** FSM as an abstract model — set of states + transitions + outputs; the universality for sequential systems.; Moore vs Mealy machines — output-on-state vs output-on-transition; the trade-offs.; State diagrams and state-transition tables — the canonical specification artifacts.; State encoding — binary, one-hot, gray code; the trade-offs (logic complexity vs flop count vs glitch immunity).; State minimization at intuition — equivalent-state detection.; FSM design flow — spec → diagram → table → encoded table → next-state and output equations → implementation.; Canonical FSM examples — sequence detector, traffic-light controller, vending machine, simple data-path controller.; A lab: design a sequence-detector FSM that recognizes "1011" in a serial bit stream; implement in Verilog; verify with testbench; observe waveforms.	CO4
M5	**Verilog HDL** Why HDL — describing hardware in text; the contrast with schematic capture; why Verilog and VHDL won.; Verilog basics — modules, ports, parameters, the always block, the assign statement.; Data types — wire vs reg, vector signals, parameters, packed/unpacked arrays.; Procedural blocks — always @(*) for combinational, always @(posedge clk) for sequential; the canonical templates.; Blocking vs non-blocking assignments — the canonical bug source; the rule of thumb (= for combinational, <= for sequential).; Hierarchy — module instantiation, port connection (positional vs named), generate blocks at intuition.; Testbenches — initial blocks, $monitor, $display, $finish; the simulation lifecycle.; SystemVerilog at intuition — the modern Verilog-2005 superset; logic type, always_comb / always_ff / always_latch; why industry has moved this way.; A lab: design a 4×4 unsigned multiplier in Verilog; write an exhaustive testbench (256 cases); verify in Icarus or Verilator; observe the synthesizer's gate count.	CO5
M6	**FPGA Prototyping and Modern Toolchains** FPGA architecture at intuition — LUTs, flip-flops, BRAM, DSP slices; the comparison to ASIC.; The FPGA design flow — RTL coding, simulation, synthesis, place-and-route, bitstream generation, programming.; Vivado (Xilinx) / Quartus (Intel) — the dominant commercial flows; the canonical campus-lab boards (Basys-3, Nexys A7).; Open-source FPGA tools — Yosys for synthesis, nextpnr for place-and-route, IceStorm for Lattice iCE40; the Project Trellis for ECP5; the IndianRISCV community using these tools.; Verilator — the cycle-accurate simulator; orders of magnitude faster than commercial tools; the modern verification choice.; Timing analysis at intuition — setup-time analysis, fmax determination; why the synthesis report matters.; Modern HDL tooling and AI-assisted hardware design — Claude / GPT-4 for HDL boilerplate generation; the discipline of always running through the simulator and synthesis tool.; A final-week lab: take the M4 sequence-detector and deploy it to a real FPGA board (or to Verilator with VCD waveform); demonstrate detection on board switches; submit the synthesis report.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Digital Design — With an Introduction to the Verilog HDL, VHDL, and SystemVerilog**	M. Morris Mano & Michael D. Ciletti	Pearson	2018 (6th Ed.)
2	**Digital Design and Computer Architecture — RISC-V Edition**	Sarah L. Harris & David Money Harris	Morgan Kaufmann	2021
3	**FPGA Prototyping by SystemVerilog Examples — Xilinx MicroBlaze MCS SoC Edition**	Pong P. Chu	Wiley	2018 (2nd Ed.)
4	**SystemVerilog for Design — A Guide to Using SystemVerilog for Hardware Design and Modeling**	Stuart Sutherland, Simon Davidmann & Peter Flake	Springer	2006 (2nd Ed.)
5	**Fundamentals of Digital Logic with Verilog Design**	Stephen Brown & Zvonko Vranesic	McGraw-Hill	2013 (3rd Ed.)
6	**Verilog HDL — A Guide to Digital Design and Synthesis**	Samir Palnitkar	Pearson	2003 (2nd Ed.)
7	**Modern VLSI Design — IP-Based Design**	Wayne Wolf	Pearson	2008 (4th Ed.)

Discrete Mathematics

25MT1002

L-T-P-S	2-2-0-0	Credits	4	Contact Hrs/Wk	4
Semester	S01	Mode	Regular	Category	BSC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply propositional and predicate logic — truth tables, equivalences, quantifiers, the canonical proof techniques (direct, contrapositive, contradiction, induction) — to construct rigorous arguments.	BTL3 · Apply
CO2	Apply set theory, relations, and functions — set operations, equivalence relations, partial orders, function composition, bijections — to model abstract structures.	BTL3 · Apply
CO3	Apply counting techniques — sum/product rules, permutations, combinations, pigeonhole principle, inclusion-exclusion, generating functions at intuition — to combinatorial problems.	BTL3 · Apply
CO4	Apply mathematical induction (weak and strong) and solve linear recurrence relations — homogeneous and non-homogeneous, characteristic equation method.	BTL3 · Apply
CO5	Apply graph theory — graphs and trees, traversals, paths and cycles, Euler and Hamiltonian circuits, planarity, graph coloring — to model and solve discrete problems.	BTL3 · Apply
CO6	Apply modular arithmetic and basic number theory — Euclidean algorithm, modular inverses, Fermat's little theorem, RSA at intuition.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PSO1
CO1	H	H	–	H
CO2	H	H	–	H
CO3	H	H	–	H
CO4	H	H	–	H
CO5	H	H	–	H
CO6	H	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Logic and Proofs** Propositional logic — propositions, connectives, truth tables, tautologies and contradictions.; Logical equivalences — De Morgan's laws, distributive laws, the canonical equivalences with worked transformations.; Predicate logic — universal and existential quantifiers, nested quantifiers, the canonical mistakes when quantifier order is swapped.; Rules of inference — modus ponens, modus tollens, hypothetical syllogism; constructing valid arguments.; Proof techniques — direct, contrapositive, contradiction, by cases; recognising which technique fits.; Mathematical induction (preview of M4) — strong vs weak, the canonical use cases.; Module problem set: 15 mixed problems on logic equivalences, quantifier arguments, and small proofs in chosen techniques.	CO1
M2	**Sets, Relations, Functions** Sets — definitions, set-builder notation, subset and proper subset, power sets, Cartesian products.; Set operations — union, intersection, complement, symmetric difference; Venn diagrams.; Relations — binary relations, properties (reflexive, symmetric, antisymmetric, transitive), equivalence relations and the partition theorem.; Partial orders and lattices at intuition — Hasse diagrams, the canonical examples (subset, divisibility).; Functions — domain, codomain, range, injective/surjective/bijective, composition, inverse functions.; Cardinality at intuition — finite vs countable vs uncountable; Cantor's diagonal argument.; Module problem set: 15 problems on relation properties, function characterization, set algebra.	CO2

	Title & Topics	CO
M3	**Counting and Combinatorics** Counting fundamentals — sum and product rules; the canonical "how many ways" problems.; Permutations and combinations — n!, P(n,r), C(n,r); ordered vs unordered selection.; Pigeonhole principle — basic and generalised forms; canonical applications.; Inclusion-exclusion principle — for two, three, n sets; canonical applications (derangements).; Binomial theorem and identities — Pascal's triangle, Vandermonde's identity.; Generating functions at intuition — ordinary and exponential GFs; the connection to recurrence solving.; Module problem set: 15 problems with mixed counting techniques.	CO3
M4	**Induction and Recurrences** Mathematical induction — weak form, base and inductive step; canonical examples (sum-of-first-n, sum-of-squares).; Strong induction — when weak isn't enough; canonical examples (every n>1 has prime factorisation).; Linear homogeneous recurrence relations — characteristic equation method; Fibonacci closed-form derivation.; Linear non-homogeneous recurrences — particular solutions, undetermined coefficients.; Recurrence-to-closed-form via generating functions at intuition.; Connection to algorithm analysis — master theorem (depth in DAAA); why divide-and-conquer recurrences arise.; Module problem set: 15 problems on inductive proofs and recurrence solving.	CO4
M5	**Graphs and Trees** Graphs — vertices and edges, directed vs undirected, weighted vs unweighted; representations (adjacency list, matrix).; Graph properties — degree, paths, cycles, connectivity, components; the handshaking lemma.; Special graphs — complete, bipartite, regular, planar; Euler's formula V-E+F=2.; Euler and Hamiltonian circuits — definitions, necessary and sufficient conditions for Euler.; Trees — definition, properties (n-1 edges for n vertices, unique paths); rooted and binary trees.; Graph coloring — chromatic number, four-color theorem at intuition; applications (scheduling, register allocation).; Module problem set: 12 problems on graph reasoning, traversal, and small-graph coloring.	CO5
M6	**Number Theory and Crypto Hooks** Divisibility and the Division Algorithm — quotient and remainder uniqueness; gcd via Euclidean algorithm.; Modular arithmetic — congruence relation, modular addition / multiplication / exponentiation; the ring Z_n.; Extended Euclidean algorithm — finding modular inverses; Bezout's identity application.; Fermat's little theorem and Euler's theorem — statements, proof sketches, applications in fast modular exponentiation.; Chinese Remainder Theorem at intuition — splitting modular problems across coprime factors.; RSA cryptosystem at intuition — public-key crypto via modular arithmetic; the connection to TLS / network security (deeper in CNAS); why discrete math matters concretely for the engineer.; Module problem set: 12 problems on Euclidean / extended-Euclidean / modular-exponentiation; one problem implementing toy RSA on paper for n=33.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Discrete Mathematics and Its Applications**	Kenneth H. Rosen	McGraw-Hill	2018 (8th Ed.)
2	**Discrete Mathematics with Applications**	Susanna S. Epp	Cengage Learning	2019 (5th Ed.)
3	**Concrete Mathematics — A Foundation for Computer Science**	Ronald L. Graham, Donald E. Knuth & Oren Patashnik	Addison-Wesley	1994 (2nd Ed.)
4	**Mathematics for Computer Science**	Eric Lehman, F. Thomson Leighton & Albert R. Meyer	MIT (free PDF)	2018
5	**A Book of Abstract Algebra**	Charles C. Pinter	Dover	2010 (2nd Ed.)
6	**Discrete Mathematics — An Open Introduction**	Oscar Levin	Self-published (free)	2024 (4th Ed.)
7	**How to Prove It — A Structured Approach**	Daniel J. Velleman	Cambridge University Press	2019 (3rd Ed.)

Ecology And Environment

25UC0009

L-T-P-S	2-0-0-0	**Credits**	0	**Contact Hrs/Wk**	2
Semester	2S	**Mode**	Regular	**Category**	AUC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand ecosystem structure, function, and ecological balance.	BTL2 · Understand
CO2	Understand biodiversity, its value, and conservation approaches.	BTL2 · Understand
CO3	Apply principles of sustainable natural-resource use.	BTL3 · Apply
CO4	Analyze environmental pollution — sources, effects, and control.	BTL4 · Analyze
CO5	Analyze climate change and global environmental issues.	BTL4 · Analyze
CO6	Analyze environmental management, law, and AI/IoT-enabled monitoring.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO5	PO6	PO7	PSO1	PSO2
CO1	–	–	M	**H**	M	–
CO2	–	–	M	**H**	M	–
CO3	–	–	**H**	**H**	M	–
CO4	M	–	–	**H**	–	M
CO5	–	–	**H**	**H**	–	M
CO6	–	M	–	**H**	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Ecosystems & Ecology** Ecosystem structure and function.; Energy flow and trophic levels.; Food chains, webs, and ecological pyramids.; Biogeochemical cycles (C, N, P, water).; Ecological succession.; Population and community ecology.; Major biomes and Indian ecosystems.; Ecological balance and stability. **Labs:** Field/observation survey of a local ecosystem and its food web.; Analyse sample ecological data (species counts) to assess ecosystem health. **Case Study:** Case Study — a degraded vs restored ecosystem (e.g., a revived lake/wetland in India): analyse how ecological structure broke down and how restoration brought it back.	CO1
M2	**Biodiversity & Conservation** Levels of biodiversity.; India as a megadiverse nation.; Biodiversity hotspots (Western Ghats, Himalaya).; Threats — habitat loss, invasive species, poaching.; IUCN status and endangered species.; In-situ conservation (parks, reserves).; Ex-situ conservation.; Ecosystem services and their value. **Labs:** Biodiversity survey of a campus/local area with species logging.; Map threats and design a conservation plan for a local species/habitat. **Case Study:** Case Study — a conservation success/failure in India (e.g., Project Tiger / a vanishing species): analyse the threats, the intervention, and the outcome.	CO2

	Title & Topics	CO
M3	**Natural Resources & Sustainability** Renewable vs non-renewable resources.; Water resources and scarcity.; Forest and land resources.; Energy resources and the transition.; Mineral and food resources.; Resource depletion and over-exploitation.; Carrying capacity and ecological footprint.; Sustainable resource management. **Labs:** Calculate your ecological/water footprint and propose reductions.; Audit a resource (water/energy) use on campus and recommend savings. **Case Study:** Case Study — a water/resource crisis in India (e.g., a city's 'day zero' or groundwater depletion): analyse the causes and sustainable solutions.	CO3
M4	**Environmental Pollution** Air pollution — sources and effects.; Water pollution and quality.; Soil contamination.; Noise pollution.; Solid-waste management.; E-waste and plastic pollution.; Pollution and human health.; Pollution-control measures and technology. **Labs:** Measure/interpret a local pollution indicator (air AQI / water quality).; Design a waste-segregation or pollution-reduction plan for a setting. **Case Study:** Case Study — Delhi's air pollution / a polluted river (Ganga/Yamuna): analyse the sources, health impacts, and the policy + tech responses.	CO4
M5	**Climate Change & Global Issues** Greenhouse effect and global warming.; Causes and evidence of climate change.; Impacts — sea level, extremes, agriculture.; Ozone depletion.; Acid rain and trans-boundary issues.; International agreements (Kyoto, Paris).; India's climate commitments (NDCs, net-zero 2070).; Mitigation and adaptation. **Labs:** Analyse climate/temperature trend data for a region.; Build a carbon-footprint estimate and a mitigation plan for a unit. **Case Study:** Case Study — a climate impact in India (e.g., extreme heat, floods, or a coastal threat): analyse the climate linkage and the adaptation/mitigation response.	CO5
M6	**Environmental Management, Law & AI** Environmental Impact Assessment (EIA).; Environmental laws of India (Air, Water, EP Acts).; Pollution Control Boards (CPCB/SPCB).; Green technology and clean production.; Circular economy.; AI/IoT for environmental monitoring.; Remote sensing for environment.; Environmental ethics and the engineer's role. **Labs:** Conduct a mini-EIA for a hypothetical project.; Design an AI/IoT-based environmental-monitoring concept for a problem. **Case Study:** Case Study — AI/IoT environmental monitoring (e.g., sensor-based air-quality or satellite forest monitoring): analyse how technology improves environmental management.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Textbook of Environmental Studies for Undergraduate Courses, 2nd Ed.**	Erach Bharucha	Universities Press (UGC)	2013
2	**Environmental Studies: From Crisis to Cure, 4th Ed.**	R. Rajagopalan	Oxford University Press	2016
3	**Environmental Science: Earth as a Living Planet, 9th Ed.**	Daniel B. Botkin & Edward A. Keller	Wiley	2014
4	**Ecology and Environment, 12th Ed.**	P. D. Sharma	Rastogi Publications	2017
5	**Environmental Science: A Global Concern, 14th Ed.**	William P. Cunningham & Mary Ann Cunningham	McGraw-Hill	2017
6	**Silent Spring**	Rachel Carson	Houghton Mifflin	2002
7	**The Sixth Extinction: An Unnatural History**	Elizabeth Kolbert	Henry Holt	2014

Electric Circuits & Energy System Fundamentals

25EE2101

L-T-P-S	2-0-2-0	**Credits**	3	**Contact Hrs/Wk**	4
Semester	S03	**Mode**	Regular	**Category**	PCC
Pre-requisite	BEEC				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the basic laws of electric circuits — Kirchhoff's current law, Kirchhoff's voltage law, passive sign convention, source transformation, and the canonical network theorems (superposition, Thevenin, Norton, maximum power transfer, Millman, Y-Δ transformation) for DC resistive networks.	BTL2 · Understand
CO2	Apply systematic mesh-current and nodal-voltage analysis (including super-mesh, super-node, dependent-source handling, matrix formulation [Z][I]=[V] / [Y][V]=[I]) to solve multi-source resistive and reactive networks, with PSPICE / LTspice verification.	BTL3 · Apply
CO3	Apply phasor analysis to AC circuits with R, L, C elements — compute impedance / admittance, analyse series and parallel RLC resonance, calculate average / reactive / apparent power and power factor, and design simple power-factor-correction networks.	BTL3 · Apply
CO4	Analyse two-port-network parameters — Z, Y, h (hybrid) and ABCD (transmission) parameters — their inter-conversion, reciprocity and symmetry properties, and the interconnection rules (series, parallel, cascade) for characterising amplifiers, filters and transmission lines.	BTL4 · Analyze
CO5	Apply transient analysis to first-order RC and RL circuits, and to second-order RLC circuits (over-damped, critically damped, under-damped), using time-domain (differential-equation) and Laplace-transform (s-domain) methods, with MATLAB / Simulink verification.	BTL3 · Apply
CO6	Analyse balanced and unbalanced three-phase circuits — star / delta connection, line vs phase quantities, power computation, two-wattmeter method — and connect the analysis to the Indian-grid 11 kV / 415 V distribution system and the upcoming microgrid / EV-charging-network context.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PSO1
CO1	**H**	**H**	M	L	–	–	**H**
CO2	**H**	**H**	**H**	M	M	–	**H**
CO3	**H**	**H**	**H**	M	–	–	**H**
CO4	**H**	**H**	**H**	**H**	–	–	**H**
CO5	**H**	**H**	**H**	**H**	M	–	**H**
CO6	**H**	**H**	**H**	M	M	L	**H**

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**DC Circuit Fundamentals & Network Theorems** Kirchhoff's Current Law (KCL) and Voltage Law (KVL); passive sign convention; power absorption vs delivery; canonical R, L, C element relations.; Independent and dependent sources — voltage- and current-controlled; the canonical four dependent-source types (VCVS, CCVS, VCCS, CCCS).; Source transformation — V-source + series-R ⇔ I-source + shunt-R; Millman's theorem for parallel-voltage-source reduction.; Superposition theorem — deactivation rules (V → short, I → open); summing individual responses; the canonical limitation to linear circuits.; Thevenin's theorem — V_th (open-circuit voltage) and R_th (dead-network resistance); Norton's equivalent and its relation to Thevenin.; Maximum power transfer theorem — R_L = R_th condition; the canonical efficiency = 50% at MPT; engineering vs power-system design trade-off.; Star-delta (Y-Δ) transformation — conversion formulas; bridge-circuit simplification; the canonical Wheatstone-bridge and three-phase-load applications.; A first lab — verify Thevenin and Norton equivalents on a bench DC resistive network; compare hand-calculation to multimeter measurement.	CO1

	Title & Topics	CO
M2	**Mesh & Nodal Analysis with Computer-Aided Verification** Mesh-current analysis — mesh currents, KVL equations, super-mesh handling when a current source is shared between two meshes.; Nodal-voltage analysis — reference-node selection, KCL equations, super-node handling when a voltage source is between two non-reference nodes.; Matrix formulation — [Z][I]=[V] for mesh analysis; [Y][V]=[I] for nodal analysis; solution by Cramer's rule and by matrix inversion (MATLAB \ operator).; Dependent sources in mesh and nodal analysis — writing constraint equations; the canonical modified matrix entry rules.; Network-graph theory — tree, co-tree, link; fundamental loops and cutsets; the canonical systematic generation of KVL / KCL equations from a graph.; Computer-aided circuit analysis — PSPICE / LTspice netlist syntax; .OP (operating point), .DC (DC sweep), .TRAN (transient) directives; the canonical "verify-then-build" engineering discipline.; Indian-context note — the canonical free LTspice (Analog Devices), Multisim (NI), PSIM (Powersim) educational licences widely used in Indian EEE labs.; A lab — solve a non-trivial 4-mesh network by hand; reproduce in LTspice; compare branch currents and node voltages.	CO2
M3	**AC Circuit Analysis & Resonance** Phasor representation of sinusoidal sources — complex-number form, Euler's formula, phasor diagrams; peak / RMS / average value conversion.; Impedance — $Z_R = R$, $Z_L = j\omega L$, $Z_C = 1/(j\omega C)$; series and parallel combinations; the canonical frequency-dependence intuition.; AC nodal and mesh analysis — same formulation as DC but with complex impedances and phasor sources; matrix solution.; Series RLC resonance — resonant frequency $\omega_0 = 1/\sqrt{(LC)}$, quality factor $Q = \omega_0 L/R$, bandwidth $BW = \omega_0/Q$; the canonical applications in radio-tuning and filter design.; Parallel RLC resonance — the canonical high-Q approximation; admittance vs frequency; the canonical band-stop and tank-circuit applications.; Power in AC circuits — average power $P = \frac{1}{2} V_m I_m \cos\varphi$, reactive power Q, apparent power $S = V_{rms} \cdot I_{rms}$; the canonical power triangle.; Power-factor correction — parallel capacitor sizing $C = Q / (\omega \cdot V^2)$; the canonical reactive-power reduction in industrial loads and the canonical Indian-DISCOM penalty regime for low PF.; Lab — build a series RLC resonant circuit on a breadboard; sweep frequency; observe and plot the canonical resonance peak and 3-dB bandwidth.	CO3
M4	**Two-Port Network Parameters** Z-parameters (open-circuit impedance) — $V_1 = Z_{11} I_1 + Z_{12} I_2$; $V_2 = Z_{21} I_1 + Z_{22} I_2$; the canonical reciprocity condition $Z_{12} = Z_{21}$.; Y-parameters (short-circuit admittance) — $I_1 = Y_{11} V_1 + Y_{12} V_2$; the canonical measurement procedure; matrix relation $[Y] = [Z]^{-1}$.; H-parameters (hybrid) — $V_1 = h_{11} I_1 + h_{12} V_2$; $I_2 = h_{21} I_1 + h_{22} V_2$; the canonical BJT-small-signal-model application.; ABCD (transmission) parameters — the canonical cascading property used in transmission-line analysis; chain-matrix multiplication.; Inter-conversion of two-port parameters — Z ⇔ Y ⇔ H ⇔ ABCD; reciprocal and symmetric networks; the canonical 4 × 4 conversion table.; Interconnection of two-port networks — series-series (Z parameters add), parallel-parallel (Y parameters add), cascade (ABCD parameters multiply); the canonical filter-design implication.; Image impedance and propagation constant — characteristic impedance Z_0 for matching networks; the canonical step toward transmission-line theory in semester-5 Electromagnetics.; Lab — measure Z- and ABCD-parameters of a passive T-network and a π-network; verify the cascade-multiplication rule against direct overall measurement.	CO4
M5	**Transient Analysis — First & Second Order Circuits** First-order RC circuit — zero-input (natural) response, zero-state (forced) response; time constant $\tau = RC$; the canonical step-response e^{-t}/τ shape.; First-order RL circuit — time constant $\tau = L/R$; step response; exponential decay; the canonical complete-response = transient + steady-state decomposition.; Initial- and final-value theorem — $x(0^+)$ from initial-condition continuity (V_C, I_L); $x(\infty)$ from DC-circuit analysis; the canonical step-by-step approach.; Second-order series RLC — characteristic equation $s^2 + 2\alpha s + \omega_0^2 = 0$; classification — overdamped, critically damped, underdamped, undamped.; Underdamped response — oscillation frequency $\omega_d = \sqrt{(\omega_0^2 - \alpha^2)}$; envelope decay; damping ratio $\zeta = \alpha/\omega_0$; the canonical bode-plot connection.; Laplace-transform method — s-domain impedances (sL for inductor, 1/(sC) for capacitor); transformed circuit; partial-fraction expansion; inverse Laplace.; Simulink modelling of RLC transients — the canonical Simscape Electrical workflow; comparison of analytical, hand-numerical and Simulink results.; Lab — capture step response of a bench RC, RL and underdamped RLC circuit on an oscilloscope; measure τ and ζ; correlate with hand-calculated values.	CO5
M6	**Three-Phase Circuits & Grid Context** Three-phase generation — balanced 120°-displaced voltage set; positive and negative phase sequence; the canonical R-Y-B phase convention used by Indian utilities.; Star (Y) connection — $V_{line} = \sqrt{3} \cdot V_{phase}$; $I_{line} = I_{phase}$; the canonical role of the neutral wire and the 4-wire LT distribution system.; Delta (Δ) connection — $V_{line} = V_{phase}$; $I_{line} = \sqrt{3} \cdot I_{phase}$; the canonical 3-wire delta industrial-feeder configuration.; Power in balanced three-phase — $P = \sqrt{3} \cdot V_L \cdot I_L \cdot \cos\varphi$; $Q = \sqrt{3} \cdot V_L \cdot I_L \cdot \sin\varphi$; $S = \sqrt{3} \cdot V_L \cdot I_L$; the canonical "√3 in everything" formula sheet.; Two-wattmeter method — $W_1 = V_{RB} \cdot I_R \cdot \cos(30° + \varphi)$; $W_2 = V_{YB} \cdot I_Y \cdot \cos(30° - \varphi)$; $P = W_1 + W_2$; the canonical lab-instrument-set used in Indian EEE labs.; Unbalanced three-phase loads — nodal analysis with neutral; Millman's method; the canonical phasor-diagram visualisation under unbalance.; Indian-grid 11 kV / 415 V distribution — the canonical R-Y-B + N four-wire LT line; the canonical phase-balancing concern at DISCOM substations.; Lab — measure three-phase power on a balanced and an unbalanced lab load using the two-wattmeter setup; verify $P = W_1 + W_2$ and infer power-factor sign from W_1 / W_2 ratio.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Engineering Circuit Analysis**	William H. Hayt, Jack E. Kemmerly & Steven M. Durbin	McGraw-Hill Education	2018 (9th Ed.)
2	**Fundamentals of Electric Circuits**	Charles K. Alexander & Matthew N. O. Sadiku	McGraw-Hill Education	2021 (7th Ed.)
3	**Network Analysis**	M. E. Van Valkenburg	Prentice Hall of India	2014 (3rd Ed.)
4	**Electric Circuits**	James W. Nilsson & Susan A. Riedel	Pearson	2019 (11th Ed.)
5	**Circuit Theory: Analysis and Synthesis**	A. Chakrabarti	Dhanpat Rai Publications	2018 (7th Ed.)
6	**Linear Circuit Analysis: Time Domain, Phasor and Laplace Transform Approaches**	Raymond A. DeCarlo & Pen-Min Lin	Oxford University Press	2012 (2nd Ed.)

Electric Motors, Drives & Electromechanical Systems

25EE2202

L-T-P-S	3-0-2-0	Credits	4	Contact Hrs/Wk	5
Semester	S04	Mode	Regular	Category	PCC
Pre-requisite	BEEC				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply transformer fundamentals — ideal vs real transformer, equivalent circuit, OC / SC tests, voltage regulation, efficiency, three-phase connections (Y-Y, Y-Δ, Δ-Y, Δ-Δ) and vector groups — to design and analysis problems on single-phase and three-phase transformers including auto-transformers and instrument transformers.	BTL3 · Apply
CO2	Apply DC-machine fundamentals — construction, EMF equation, types (separately-excited, shunt, series, compound), torque–speed characteristics, starting methods, armature-voltage and field-weakening speed control — to canonical traction, manufacturing-drive and high-precision-control problems.	BTL3 · Apply
CO3	Apply three-phase induction-motor analysis — construction, rotating magnetic field, slip, equivalent circuit, torque–slip characteristic, starting methods (DOL, star–delta, autotransformer, soft-starter, VFD) and IS / IEC IE2 / IE3 / IE4 / IE5 energy-efficiency standards — to industrial-motor sizing and selection problems.	BTL3 · Apply
CO4	Apply synchronous-machine fundamentals — construction (salient-pole vs cylindrical-rotor), EMF generation, armature reaction, equivalent circuit, voltage regulation (synchronous-impedance, MMF, ZPF methods), parallel operation, V- and inverted-V-curves — to generator-and-motor analysis in power-plant and industrial-drive contexts.	BTL3 · Apply
CO5	Analyse special electric machines — BLDC motor, PMSM, switched-reluctance motor, stepper motor, universal motor and linear motor — their construction, operating principles, characteristic curves and canonical EV / automation / aerospace applications, with attention to the rare-earth supply-chain context.	BTL4 · Analyze
CO6	Analyse modern motor-drive systems — EV traction-motor design, regenerative-braking strategies, sensorless vector control, motor-drive simulation tools (MATLAB Simscape Electrical, PSIM, PLECS), the AI-augmented motor-control workflow, and the Indian EV-and-drives industry ecosystem.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO7	PO11	PSO1
CO1	H	H	H	M	–	–	H
CO2	H	H	H	–	–	–	H
CO3	H	H	H	M	M	–	H
CO4	H	H	H	–	–	–	H
CO5	H	H	H	H	M	–	H
CO6	H	H	H	H	M	M	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Transformers — Single-Phase, Three-Phase, Special-Purpose** Ideal transformer — turns ratio, voltage transformation, current transformation; the canonical $V_1/V_2 = N_1/N_2 = I_2/I_1$ set.; Real transformer — winding resistances, leakage reactance, magnetising branch; the canonical T-equivalent circuit and approximate equivalent circuit.; Open-circuit and short-circuit tests — the canonical lab procedure to extract equivalent-circuit parameters; canonical IS 2026 test norms.; Voltage regulation — definition, all-day efficiency, condition for maximum efficiency; the canonical ½×full-load result.; Three-phase transformers — Y-Y, Y-Δ, Δ-Y, Δ-Δ connections; vector groups (Yd11, Dyn11); the canonical 30° phase-shift and "don't parallel different vector groups" rule.; Auto-transformer — the canonical "copper-saving" advantage; canonical starting-of-induction-motor application; safety implications.; Instrument transformers — current transformer (CT), potential transformer (PT); the canonical metering and protection applications; canonical Indian standards (IS 2705, IS 3156).; A first lab — perform OC and SC tests on a 1 kVA single-phase lab transformer; extract equivalent-circuit parameters; predict full-load voltage regulation; verify by direct measurement.	CO1

	Title & Topics	CO
M2	**DC Machines — Construction, EMF, Characteristics & Control** DC machine construction — stator (field winding), rotor (armature), commutator and brushes; the canonical mechanical-to-electrical (generator) and electrical-to-mechanical (motor) conversion.; EMF equation E = (Φ Z N P) / (60 A); the canonical generator-action equation derivation.; Types of DC machines — separately-excited, shunt, series, compound (cumulative / differential, long-shunt / short-shunt); the canonical application map for each.; Torque equation T = (Φ Z I_a P) / (2π A); speed–torque characteristics for each type; the canonical T-N curves.; Starting methods — 3-point starter, 4-point starter; the canonical reason DC motors require external starters (high starting current at low back-EMF).; Speed-control methods — armature-voltage control (Ward-Leonard system), field-weakening control; canonical historical applications.; DC machine in 2026 context — declining new applications (BLDC has replaced DC in most consumer / EV / appliance uses); still dominant in Indian Railways legacy locomotives and high-precision research applications.; Lab — characterise a small DC shunt motor; measure speed–torque curves at different field currents; verify against textbook predictions.	CO2
M3	**Three-Phase Induction Motors** Construction — stator (3-phase winding), rotor (squirrel-cage vs wound-rotor / slip-ring); the canonical "workhorse of industrial motors" (~70% of industrial electrical-energy consumption).; Rotating magnetic field — the canonical three-phase-currents-produce-rotating-field result; synchronous speed N_s = 120f/P.; Slip and rotor frequency — s = (N_s − N_r)/N_s; canonical relation to rotor frequency f_r = s·f.; Equivalent circuit — referred to stator; the canonical T-equivalent; per-phase-impedance approach.; Torque-slip characteristic — starting torque, maximum (pull-out) torque, full-load torque; the canonical complete T-s curve.; Starting methods — DOL (Direct On-Line), star-delta starter, autotransformer starter, soft-starter, VFD; the canonical industrial-selection map.; Speed-control methods — pole-changing (consequential-pole, Dahlander), rotor-resistance (slip-ring motors only), V/f control via VFD; the canonical modern dominance of VFD-based control.; Energy-efficiency standards — IE1 / IE2 / IE3 / IE4 / IE5 per IS 12615 / IEC 60034-30-1; canonical Indian BEE 5-star rating; the canonical 5-star IE3+ induction motor as the 2026 industrial standard.; Lab — perform no-load and blocked-rotor tests on a 3-phase induction motor; extract equivalent-circuit parameters; predict efficiency at full load; verify against measured values.	CO3
M4	**Synchronous Machines — Generators & Motors** Construction — salient-pole (low-speed hydro / diesel generators) vs cylindrical / non-salient-pole (high-speed thermal / nuclear / gas-turbine generators); the canonical N_s = 120f/P relationship.; EMF generation — the canonical phasor diagram approach; armature winding factors (distribution factor, pitch factor).; Armature reaction — the canonical demagnetising and cross-magnetising effects under leading vs lagging vs unity power-factor loads.; Equivalent circuit — synchronous reactance X_s, the canonical voltage-vector approach.; Voltage regulation — synchronous-impedance method (EMF method), MMF method, ZPF method; the canonical "why three methods" trade-off.; Parallel operation of synchronous generators — the canonical synchronisation conditions (voltage match, frequency match, phase sequence, phase angle); the canonical synchroscope and dark / bright-lamp methods; load-sharing under varying excitation and prime-mover input.; V-curves and inverted-V-curves — the canonical reactive-power-control characteristic; applications in synchronous condenser for power-factor correction at substations.; Synchronous motor — canonical constant-speed-drive and power-factor-correction applications; starting methods (damper winding, pony motor, slip-frequency starting).; Lab — synchronise a small lab synchronous generator to the lab grid; observe synchronisation conditions; operate at unity, leading and lagging power factor; sketch V-curve.	CO4
M5	**Special Electric Machines** BLDC motor (Brushless DC) — permanent-magnet rotor with electronic commutation; the canonical EV-and-appliance traction motor; Hall-sensor-based 6-step commutation.; PMSM (Permanent-Magnet Synchronous Motor) — sinusoidal back-EMF; the canonical high-performance EV-drive choice (Tesla Model 3, Ola Electric S1 Pro, Ather 450X, TVS iQube).; Switched-reluctance motor (SRM) — no permanent magnets, no rotor winding; canonical applications in industrial drives and rare-earth-free EV concepts.; Stepper motor — open-loop position control; the canonical 3D-printer / CNC-machine / robotics-actuator use cases; full-step / half-step / micro-step.; Universal motor — operates on both AC and DC; the canonical kitchen-appliance and power-tool motor.; Linear motors — linear-induction and linear-synchronous; canonical Maglev train and high-speed industrial-conveyor applications; the Shanghai Maglev and SR Magneglide / IIT-M LIM research benchmarks.; Modern motor-design context — the rare-earth supply-chain concern (China-dominated NdFeB market); India's response — rare-earth-mineral diversification, ferrite-magnet PMSM design, SRM for rare-earth-free EVs.; Lab — characterise a small BLDC motor; measure back-EMF waveform; implement Hall-sensor-based 6-step commutation on an Arduino / STM32; observe step response.	CO5

	Title & Topics	CO
M6	**Modern Motor-Drive Systems & EV Traction** EV traction-motor system architecture — the canonical PMSM / BLDC + battery + inverter + motor-control unit (MCU); canonical Tesla / Ola Electric / Tata EV / Mahindra Last Mile Mobility design choices.; Field-Oriented Control (FOC) / Vector Control — the canonical d-q decoupling enabling DC-motor-like control of PMSM and induction motors; canonical industrial-VFD and EV-MCU embedded firmware.; Regenerative braking — the canonical bidirectional inverter operation; regen-vs-friction-brake blending logic; canonical Indian-EV implementation context.; Sensorless motor control — eliminating Hall sensors / encoders via back-EMF observation (Luenberger observer, sliding-mode observer, MRAS); the canonical cost-reduction and reliability-improvement technique.; Modern simulation tools — MATLAB Simscape Electrical (Simulink-native), PSIM (specialised for power electronics + motor drives), PLECS (precision drives); the canonical industrial-tooling map.; Microcontroller and DSP platforms for motor control — STM32 G-series / F-series, TI C2000 (Piccolo / Delfino), NXP S32K; the canonical EV-grade automotive MCU choice.; AI-augmented motor control — neural-network-based speed estimation, ML-tuned PI gains, model-predictive control (MPC); the canonical 2025+ research direction; LLM coding assistants (Claude / GPT-4) for motor-control firmware boilerplate.; Indian motor-and-drives industry context — Crompton Greaves, Bharat Bijlee, Kirloskar Electric, Siemens-Crompton JV (legacy industrial); Ola Electric, Ather, Tata Motors EV, Mahindra (EV); the canonical 2025-2030 EV-driven manufacturing capacity expansion.; Final-week lab — implement V/f open-loop control of an induction motor on STM32 / TI C2000 motor-control kit; demonstrate speed-control with safety-aware ramp-up and protection logic.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Electric Machinery**	A. E. Fitzgerald, Charles Kingsley Jr. & Stephen D. Umans	McGraw-Hill	2014 (7th Ed.)
2	**Electric Machinery Fundamentals**	Stephen J. Chapman	McGraw-Hill Education	2020 (6th Ed.)
3	**Electric Drives**	Ned Mohan	Wiley	2014 (3rd Ed.)
4	**Electric Motor Drives — Modeling, Analysis, and Control**	R. Krishnan	Prentice Hall	2001
5	**Permanent Magnet Synchronous and Brushless DC Motor Drives**	R. Krishnan	CRC Press	2009
6	**Modern Power Electronics and AC Drives**	Bimal K. Bose	Prentice Hall	2001
7	**Electrical Machinery**	P. S. Bimbhra	Khanna Publishers	2018 (7th Ed.)

Electric Power Generation Transmission And Distribution

25EE2203

L-T-P-S	3-1-0-0	**Credits**	4	**Contact Hrs/Wk**	4
Semester	S04	**Mode**	Regular	**Category**	PCC
Pre-requisite	ECES				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply generation-technology fundamentals — thermal, hydro, nuclear, solar PV, wind, pumped-storage and battery storage — to compare the operating principle, cost structure and emission profile of each, with specific reference to the canonical Indian generation mix and the 2030 / 2050 net-zero pathway.	BTL3 · Apply
CO2	Apply power-plant economics — load curves, load-duration curve, load factor, plant capacity factor, plant utilisation factor, base-load vs peaking-plant economics, two- / three-part tariffs, LCOE — to compute and benchmark generation cost in the Indian DISCOM / UDAY context.	BTL3 · Apply
CO3	Apply transmission-line analysis — short, medium and long line models, π-model and T-model, ABCD parameters, voltage regulation, line efficiency — to predict the steady-state performance of an Indian-grid transmission corridor.	BTL3 · Apply
CO4	Apply transmission-system engineering — overhead-line conductor selection (ACSR, AAAC, ACCC, HTLS), tower types, insulator design, sag and tension calculation, corona, underground cables, HVDC, GIS — to design problems on Indian 765 / 400 / 220 / 132 kV systems.	BTL3 · Apply
CO5	Analyse distribution-system design — radial, ring-main and mesh topologies, distribution-transformer sizing, voltage regulation, on-load tap-changing, power-factor improvement, AT&C losses, AMI — in the Indian-DISCOM context (UDAY, RDSS, Saubhagya, Smart Meter National Programme).	BTL4 · Analyze
CO6	Analyse modern smart-grid systems — SCADA, IEC 61850 substation automation, WAMS / PMU, microgrids, grid-connected renewables, AI-augmented power-system operation — in the Indian National Smart Grid Mission (NSGM) rollout context.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO7	PO11	PSO1
CO1	**H**	**H**	**H**	–	–	**H**	–	**H**
CO2	**H**	**H**	M	–	–	–	M	**H**
CO3	**H**	**H**	**H**	–	M	–	–	**H**
CO4	**H**	**H**	**H**	M	–	–	–	**H**
CO5	**H**	**H**	**H**	–	–	–	M	**H**
CO6	**H**	**H**	**H**	–	**H**	**H**	–	**H**

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Generation Technologies & the Indian Mix** Thermal power plants — coal-fired (the dominant Indian generation, ~50% of capacity in 2026), gas-fired, oil-fired; the canonical Rankine cycle; supercritical and ultra-supercritical units.; Indian coal context — Coal India Limited (CIL), the canonical lignite / bituminous / sub-bituminous mix; the canonical 2050 net-zero pathway and CCUS (carbon capture, utilisation and storage) intuition.; Hydroelectric power plants — high-head (Pelton, 250+ m), medium-head (Francis, 30-300 m), low-head (Kaplan, < 30 m); canonical Indian hydro context (NHPC, Bhakra, Tehri, Sardar Sarovar).; Nuclear power plants — the canonical PHWR (Pressurised Heavy-Water Reactor) used by NPCIL (Tarapur, Kalpakkam, Kakrapar); India's three-stage program (PHWR → FBR → Thorium); the upcoming Bhavini PFBR.; Solar power plants — PV (utility-scale crystalline-silicon, rooftop) vs CSP (concentrated solar power); canonical Indian grid-scale projects (Bhadla Solar Park 2.25 GW, Pavagada 2.05 GW, Kurnool 1 GW); inverter, MPPT and string-vs-central architecture.; Wind power plants — onshore (Tamil Nadu, Karnataka, Gujarat clusters) vs offshore (early-stage Gujarat / Tamil Nadu MNRE pilots); canonical Suzlon, Vestas, Inox Wind, ReGen Powertech market context.; Storage technologies — pumped hydro (Tehri, Srisailam, Nagarjunasagar reversible units), battery storage (Li-ion grid-scale BESS), hydrogen at intuition; the canonical 2026 grid-scale storage role under renewable-integration push.; Lab — compare 2026 LCOE of solar PV vs coal vs gas vs nuclear in the Indian context using current MNRE / CERC / IEA data; reflect on India's 500 GW non-fossil 2030 target.	CO1
M2	**Power-Plant Economics** Load curves — daily, monthly, annual; the canonical analysis of demand patterns; the canonical Indian-summer-evening peak phenomenon.; Load-duration curve — hourly demand sorted in descending order; the canonical visualisation for capacity planning and energy-mix optimisation.; Load factor, plant capacity factor, plant utilisation factor — definitions, typical Indian-utility values, the canonical performance-metric set.; Base-load vs intermediate vs peaking plants — the canonical economic-dispatch logic; merit order; canonical Indian base-load (NTPC coal, NPCIL nuclear) and peak-load (gas, pumped-hydro) generation mix.; Cost of generation — fixed costs (capital, O&M, insurance, taxes), variable costs (fuel, ash handling, water); the canonical Levelised Cost of Energy (LCOE) formula.; Tariff structures — two-part tariff, three-part tariff, time-of-day (ToD) tariff; the canonical Indian electricity-tariff design across DISCOMs; cross-subsidisation in the Indian context.; India's power-sector financial context — DISCOM aggregate technical and commercial (AT&C) losses (~15-20% in 2026), UDAY scheme, Late Payment Surcharge Rules 2022, Revamped Distribution Sector Scheme (RDSS) 2021-2026.; First lab — given a 24-hour load profile for an Indian DISCOM area, compute load factor and design the optimal generation mix (base-load + intermediate + peaking) for minimum cost; verify with the canonical Indian-grid context.	CO2
M3	**Transmission-Line Modelling** Transmission-line motivation — connecting generators to loads at high voltage to reduce I^2R losses; the canonical Indian-grid hierarchy 765 / 400 / 220 / 132 / 66 / 33 / 11 kV / 415 V.; Line parameters — resistance R, inductance L, capacitance C per unit length; how they're computed for typical conductor configurations (single, bundled, double-circuit, three-phase).; Short transmission line (< 80 km) — only series impedance; the canonical equivalent circuit; voltage regulation by phasor approximation.; Medium transmission line (80-250 km) — nominal π-model and T-model with lumped shunt admittance; the canonical analysis approach.; Long transmission line (> 250 km) — distributed-parameter model; hyperbolic-function (sinh, cosh) ABCD parameters; characteristic impedance Z_c and propagation constant γ.; Voltage regulation and transmission efficiency — the canonical figures-of-merit; the canonical "sending-end vs receiving-end" power-circle diagrams.; Modern HVDC at intuition — high-voltage DC for very long distances and asynchronous interconnections; canonical Indian HVDC links (Champa-Kurukshetra ±800 kV, NER-NR HVDC, Vindhyachal back-to-back).; Lab — in MATLAB / PSCAD, simulate a 220 kV transmission line of 200 km using the π-model; sweep load conditions; observe voltage regulation; compare to actual Indian-grid line parameters.	CO3
M4	**Transmission-System Engineering** Overhead transmission lines — conductor types (ACSR, AAAC, ACCC, HTLS), tower types (lattice / monopole / guyed), insulator types (porcelain, glass, polymer composite); canonical PowerGrid Corporation / state-transco design conventions.; Sag and tension calculation — the canonical parabolic and catenary approximations; the canonical effect of temperature, wind, ice loading per IS 802.; Corona effect — disruptive critical voltage, visual critical voltage; canonical mitigation via bundle conductors (twin / triple / quad bundle); 765 kV and 1100 kV experimental UHV-AC context.; Underground cables — the canonical XLPE construction; oil-filled cables; canonical urban high-voltage applications in Mumbai, Delhi, Bengaluru CBD.; High-voltage AC transmission hierarchy — 765 / 400 / 220 kV inter-state corridors; canonical National Grid backbone; Inter-Regional Energy Transfer Capacity (IRETC) metrics.; HVDC transmission — long-distance bulk-power transfer, asynchronous-grid interconnection; canonical Indian HVDC links (Champa-Kurukshetra 6000 MW, Vindhyachal back-to-back, Rihand-Delhi); LCC vs VSC HVDC.; Gas-Insulated Switchgear (GIS) at intuition — for urban substations and brownfield expansion; the canonical compact-substation technology; canonical applications in Delhi Metro substations and high-density CBD areas.; Lab — design a 220 kV transmission line for a given corridor — select conductor, compute sag for different conditions, estimate corona losses; deliver a design report.	CO4

	Title & Topics	CO
M5	**Distribution-System Design** Distribution voltage levels — primary distribution (33 / 11 kV), secondary (415 / 230 V); the canonical Indian-DISCOM hierarchy; rural vs urban.; Distribution topologies — radial (cheap, low-reliability), ring-main (medium-reliability), interconnected / mesh (high-reliability, expensive); the canonical urban-vs-rural choices.; Distribution-transformer sizing — load forecasting, diversity factor, coincidence factor; the canonical sizing methodology and IS 1180 / IS 2026 transformer-loading norms.; Voltage regulation in distribution — line voltage drop, on-load tap-changer (OLTC) action, off-circuit tap changer; the canonical voltage-band requirement per IS standards.; Power-factor improvement — capacitor banks at substation / feeder / consumer end; the canonical industrial-customer requirement (kVAh billing vs kWh billing); SVCs and STATCOMs at intuition.; Aggregate Technical and Commercial (AT&C) losses — the canonical Indian-DISCOM concern; technical (line and transformer losses) vs commercial (theft, billing inefficiency, non-payment) breakdown; canonical 2026 reduction targets per RDSS.; Smart distribution — Advanced Metering Infrastructure (AMI), Distribution Management Systems (DMS), Outage Management Systems (OMS); the canonical Smart Meter National Programme rollout and RDSS Phase-I / II milestones.; Lab — design an LT distribution network for a residential colony of 100 houses; size the distribution transformer; select conductor; verify voltage-drop compliance; deliver design report.	CO5
M6	**Modern Smart-Grid Systems** SCADA architecture — Remote Terminal Units (RTUs) at substations, master stations at Load Despatch Centres; the canonical Indian POSOCO / NLDC / RLDC / SLDC hierarchy.; IEC 61850 substation automation — the canonical modern protocol; GOOSE (Generic Object Oriented Substation Event) messaging; Sampled Values (SV); MMS (Manufacturing Message Specification) reporting.; Wide-Area Measurement System (WAMS) — Phasor Measurement Units (PMUs) with GPS-time-stamped phasor data; canonical PMU deployment in Indian grid since 2015 URTDSM project; SynchroPhasor Data Concentrators.; Microgrids — small grids with local generation (rooftop PV, diesel, biomass) and storage that can island and re-synchronise; canonical campus / industrial-park applications (IIT-Madras, IIT-Bombay microgrid pilots).; Grid-connected renewables challenges — variability, ramp rates, frequency response, low-inertia issues; the canonical 2024+ engineering problems with high RE penetration.; AI in power-system operation — load forecasting (ML-based), RE generation forecasting, anomaly detection on PMU data, congestion-management RL agents; the canonical 2025+ research direction at IISc, IIT-M, IIT-B.; India's smart-grid roadmap — National Smart Grid Mission (NSGM), Smart-Grid pilots (Mysuru, Naroda, Puducherry, Pune), AMI rollout under RDSS; the canonical 2025-2030 outlook.; Final-week lab — simulate a microgrid in MATLAB Simulink with solar PV + battery storage + diesel backup; demonstrate islanded and grid-connected operation; reflect on the canonical engineering challenges.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Power System Analysis and Design**	J. Duncan Glover, Thomas J. Overbye & Mulukutla S. Sarma	Cengage Learning	2017 (6th Ed.)
2	**Electrical Power Systems**	C. L. Wadhwa	New Age International	2009 (6th Ed.)
3	**Power System Engineering**	D. P. Kothari & I. J. Nagrath	McGraw-Hill Education India	2019 (3rd Ed.)
4	**A Course in Power Systems**	J. B. Gupta	S. K. Kataria & Sons	2013
5	**Power System Protection and Switchgear**	B. Ravindranath & N. Chander	New Age International	2011 (2nd Ed.)
6	**Smart Grid — Technology and Applications**	Janaka Ekanayake, Kithsiri Liyanage, Jianzhong Wu, Akihiko Yokoyama & Nick Jenkins	Wiley	2012
7	**Renewable Energy Integration — Practical Management of Variability, Uncertainty, and Flexibility in Power Grids**	Lawrence E. Jones	Academic Press	2017 (2nd Ed.)

Electric Vehicle Systems & Battery Technologies

25EE4109

L-T-P-S	3-0-0-0	Credits	3	Contact Hrs/Wk	3
Semester	S07	Mode	Regular	Category	PCC
Pre-requisite	PEWC				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze electric-vehicle architectures (BEV, HEV, PHEV, FCEV) and the powertrain components and their interactions.	BTL4 · Analyze
CO2	Analyze vehicle dynamics and the traction power/energy requirement — the drive cycle, force balance and motor sizing — for an EV.	BTL4 · Analyze
CO3	Analyze battery cell chemistries, characteristics (energy/power density, cycle life, safety) and pack architecture for EV applications.	BTL4 · Analyze
CO4	Apply Battery Management System functions — state-of-charge/health estimation, cell balancing, thermal and safety management — to a battery pack.	BTL3 · Apply
CO5	Apply traction-drive and regenerative-braking control concepts (using PEWC converters and CSEA control as building blocks) to the EV powertrain.	BTL3 · Apply
CO6	Analyze EV charging infrastructure (AC/DC, standards, fast charging) and vehicle-to-grid integration, including the Indian standards and policy context.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO6	PO7	PSO1
CO1	H	H	–	–	–	H
CO2	H	H	–	–	–	H
CO3	H	H	–	–	H	H
CO4	H	–	H	–	–	H
CO5	H	–	H	–	–	H
CO6	H	–	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**EV Architectures and Powertrain** The EV transition: drivers, India's e-mobility mission and FAME/PLI policy.; Architectures: BEV, HEV (series/parallel), PHEV, FCEV.; Powertrain components: battery, converter, inverter, motor, transmission.; Energy flow in the EV; efficiency at each stage.; Range, performance and cost trade-offs.; 48 V mild hybrids and two-/three-wheeler EVs (the India volume segment).; Comparison with IC vehicles.; Framing the EV design case for the project.	CO1
M2	**Vehicle Dynamics and Traction Requirements** Forces on a vehicle: rolling resistance, aerodynamic drag, gradient, acceleration.; The tractive-effort and power equation.; Drive cycles (e.g. MIDC/WLTP) and energy consumption.; Motor sizing from the drive cycle.; Choice of traction motor: PMSM, induction, BLDC, SRM (construction owned by EMDE).; Gear ratio and single-speed transmission.; Range estimation from battery capacity and consumption.; Sizing the powertrain for the project vehicle.	CO2
M3	**Battery Cells, Chemistries and Packs** Cell fundamentals: voltage, capacity, C-rate, energy vs power cells.; Chemistries: Li-ion (NMC, LFP), and emerging (sodium-ion, solid-state) at concept level.; Key metrics: energy density, power density, cycle life, calendar life, cost.; Cell formats: cylindrical, prismatic, pouch.; Pack architecture: series/parallel configuration, modules, the pack.; Battery modelling: equivalent-circuit models.; Degradation and ageing mechanisms.; Designing a pack to meet the project voltage/energy target.	CO3

	Title & Topics	CO
M4	**Battery Management Systems (BMS)** Why a BMS: safety, performance, longevity.; State-of-Charge (SoC) estimation: coulomb counting, OCV, Kalman-filter methods.; State-of-Health (SoH) estimation.; Cell balancing: passive vs active.; Thermal management: air, liquid, and battery thermal runaway prevention.; Protection: over/under-voltage, over-current, over-temperature.; BMS architecture and communication (CAN).; Specifying/simulating the BMS for the project pack.	CO4
M5	**Traction Drives and Regenerative Braking** The traction inverter (built from PEWC converter knowledge).; Motor control strategies: scalar vs vector (FOC) control (using CSEA control).; Torque and speed control for traction.; Regenerative braking: recovering energy, the limits.; Wide-bandgap (SiC/GaN) in EV powertrains (devices owned by PEWC).; Powertrain efficiency mapping.; Functional safety (ISO 26262) concept for the powertrain.; Modelling the traction-drive behaviour over a drive cycle.	CO5
M6	**Charging Infrastructure and V2G** AC charging: on-board charger, levels.; DC fast charging: off-board, the high-power path.; Charging standards: CCS, CHAdeMO, and India's connectors/AIS standards.; Charging-station architecture and the grid impact.; Vehicle-to-Grid (V2G) and bidirectional charging.; Smart charging and load management (links to MPSG/REGI).; Battery swapping (the India two/three-wheeler model).; A charging + V2G plan for the project vehicle for the demo.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Electric and Hybrid Vehicles: Design Fundamentals**	Iqbal Husain	CRC Press	2021 (3rd Ed.)
2	**Modern Electric, Hybrid Electric, and Fuel Cell Vehicles**	Mehrdad Ehsani et al.	CRC Press	2018 (3rd Ed.)
3	**Battery Management Systems, Volume I & II**	Gregory L. Plett	Artech House	2015
4	**Electric Powertrain**	John G. Hayes & G. Abas Goodarzi	Wiley	2018
5	**Lithium-Ion Batteries: Fundamentals and Applications**	Yuping Wu	CRC Press	2015

Embedded Ai And Tiny Ml

25CS3113

L-T-P-S	2-0-2-0	**Credits**	3	**Contact Hrs/Wk**	4
Semester	S05	**Mode**	Regular	**Category**	PCC
Pre-requisite	EBSD				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the embedded-AI motivation and constraints — the canonical cloud-vs-edge trade-off (latency, bandwidth, privacy, energy), the canonical TinyML resource envelope (< 1 MB Flash, < 256 KB RAM, < 100 mW power), and the application landscape.	BTL2 · Understand
CO2	Understand modern embedded-platform architectures — ARM Cortex-M0/M4/M7/M33/M55/M85 (with Helium MVE), ESP32 family (S3, C3, P4), STM32, RISC-V (BL808, ESP32-C6), Nordic nRF52840 / nRF54L — and select an appropriate MCU for an embedded-AI workload.	BTL2 · Understand
CO3	Apply neural-network compression techniques — post-training quantization (INT8 / INT4), quantization-aware training, structured / unstructured pruning, knowledge distillation, neural architecture search (MCUNet, MobileNetV3) — to fit large models into MCU memory.	BTL3 · Apply
CO4	Apply embedded-inference frameworks — TensorFlow Lite Micro, ExecuTorch (Meta), ONNX Runtime, Edge Impulse, microTVM, CMSIS-NN (ARM), TensorRT (Jetson) — to deploy a trained model to an MCU or single-board computer.	BTL3 · Apply
CO5	Apply sensor-integration techniques — IMU (accelerometer / gyro / magnetometer), MEMS microphone (PDM / I2S), camera modules (OV2640, OV5640, HiMax HM01B0), environmental sensors (BME280, SHT41, MQ-series) — with on-device pre-processing (windowing, MFCC, FFT, filtering) for ML pipelines.	BTL3 · Apply
CO6	Analyze low-power design, OTA updates, security and end-to-end TinyMLOps for production deployment — deep-sleep / wake-on-event, energy harvesting (solar, RF, vibration), secure boot, signed firmware, model versioning, fleet management — in Indian-context applications (smart agriculture, urban air quality, monsoon flood monitoring).	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PSO1
CO1	**H**	M	–	–	M	–	–	**H**
CO2	**H**	**H**	**H**	–	M	–	–	**H**
CO3	**H**	**H**	**H**	M	**H**	–	–	**H**
CO4	**H**	**H**	**H**	–	**H**	–	–	**H**
CO5	**H**	**H**	**H**	–	**H**	–	–	**H**
CO6	**H**	**H**	**H**	–	**H**	**H**	**H**	**H**

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Edge-AI Motivation & The TinyML Envelope** Cloud vs edge vs near-edge vs fog computing — the canonical latency / bandwidth / privacy / energy trade-off matrix; canonical Indian examples (Aadhaar Auth on-device biometrics, smart-meter rollout under National Smart Grid Mission, e-Sanjeevani telehealth).; TinyML definition (TinyML Foundation 2019; Reddi / Warden, Harvard CS249r) — the canonical <1 MB Flash, <256 KB RAM, <100 mW resource envelope; canonical "always-on, always-listening, always-watching" model.; The canonical TinyML application taxonomy — keyword spotting (KWS, Hey Siri / OK Google paradigm), visual wake words, anomaly detection, gesture recognition, predictive maintenance, sensor-fusion classification.; Edge-AI on richer platforms — Raspberry Pi (Cortex-A), NVIDIA Jetson Nano / Orin Nano (CUDA edge), Google Coral (Edge TPU), Hailo-8, AMD-Xilinx Kria; the canonical "edge AI" continuum from MCU to edge-server.; Industry context — the canonical Apple Neural Engine, Google Pixel Tensor / Tensor G4, Qualcomm Hexagon NPU, MediaTek APU, Intel Movidius / NPU; Indian-context: Ola Krutrim, Sarvam-1 edge variant.; Energy budget — the canonical microwatt vs milliwatt vs watt regime; coin-cell battery (~1000 mWh) vs LiPo (~5000 mWh) vs solar harvesting (canonical Indian sun-hours: 5-7 per day across most states).; Hardware-software co-design — the canonical MCUNet (Han et al. 2020) approach: search the network jointly with the inference scheduler; canonical 5-10x improvement vs unoptimized deployment.; Hands-on lab — Power-profile a sample TinyML application (Person Detection, MobileNetV2-quantized) on an ESP32-S3 + Joulescope IV meter; measure energy per inference (mJ), idle current ((mu)A), and project battery life on a 2000-mAh LiPo cell; document the energy-vs-accuracy Pareto frontier.	CO1
M2	**Embedded Platforms — MCU Architectures & Boards** ARM Cortex-M family — M0+/M3/M4F (canonical mid-range with FPU), M7 (high-performance with cache), M33 (TrustZone-M security), M55 / M85 (Helium MVE for ML acceleration); the canonical CMSIS-NN library.; ESP32 family (Espressif) — ESP32 (classic, dual-core Xtensa LX6), ESP32-S3 (with vector instructions for AI), ESP32-C3 / C6 (RISC-V), ESP32-P4 (RV64); the canonical Wi-Fi + BLE + low-cost-MCU sweet spot for Indian IoT projects.; Raspberry Pi Pico family — RP2040 (dual Cortex-M0+, no FPU), RP2350 (Cortex-M33 + RISC-V Hazard3, 2024); the canonical sub-$5 board for student projects.; STM32 family — STM32H7 (Cortex-M7 480 MHz), STM32U5 (ultra-low-power Cortex-M33), STM32N6 (with Neural-ART NPU, 2024); canonical STMicroelectronics X-CUBE-AI toolchain.; Nordic nRF families — nRF52840 (Cortex-M4F + BLE), nRF54L15 / nRF54H20 (multi-core, 2024); the canonical low-power BLE wearable / sensor-node platform.; RISC-V on the edge — BL808 (Bouffalo Lab, multi-core RV with AI accelerator), CH32V (WCH, ultra-cheap), SiFive Performance P670; the canonical 2024+ open-ISA momentum, India-context: Shakti (IIT-M), DIR-V mission.; Single-board computers (SBCs) — Raspberry Pi 5 (BCM2712 Cortex-A76), Orange Pi 5 (RK3588), NVIDIA Jetson Orin Nano (Ampere GPU + 8 GB LPDDR5), Google Coral Dev Board Mini (Edge TPU); when an MCU isn't enough.; Hands-on lab — Set up the dev environment for 3 boards: Arduino Nano 33 BLE Sense (Cortex-M4F), ESP32-S3 DevKit, Raspberry Pi Pico W; compile and flash a "hello sensors" sketch on each that reads IMU + temperature + humidity; benchmark MCU performance using CoreMark and the TinyML perf benchmark (MLPerf Tiny).	CO2
M3	**Model Compression for the Edge** Quantization fundamentals — the canonical FP32 → INT8 mapping (4x size reduction, 2-4x speedup on CPU, near-free on NPU); symmetric vs asymmetric quantization; per-tensor vs per-channel scales.; Post-Training Quantization (PTQ) — the canonical "calibration set" approach; representative dataset, percentile-clipping; canonical TFLite Converter + ONNX Runtime quantization APIs.; Quantization-Aware Training (QAT) — the canonical "fake-quantize during training" approach; canonical 1-2% accuracy recovery vs PTQ; PyTorch QAT and TF Model Optimization Toolkit.; Sub-INT8 quantization — INT4 (Bitsandbytes, GPTQ), INT2 (canonical 2024+ for LLMs), 1.58-bit (Microsoft BitNet, 2024); the canonical accuracy-vs-size trade-off; canonical limitations for vision vs language.; Pruning — unstructured (canonical magnitude-based, Han et al. 2015), structured (channel-pruning, filter-pruning), N:M sparsity (NVIDIA Ampere 2:4 hardware-accelerated); the canonical "prune-and-finetune" recipe.; Knowledge Distillation (Hinton et al. 2015) — the canonical teacher-student approach; soft targets, temperature; canonical 2024+ "distillation for LLMs" (DistilBERT, TinyLLaMA, Phi-3-mini); MobileNet families.; Neural Architecture Search for MCUs — MCUNet (Lin et al. NeurIPS 2020), MCUNetV2 (Lin et al. 2021), MicroNets (Banbury et al. MLSys 2021); the canonical "co-search architecture + scheduler" approach.; Hands-on lab — Take a trained Hindi keyword-spotting model (~1 MB FP32, 100 K parameters) and compress it for ESP32-S3: apply PTQ INT8 → measure accuracy drop; apply QAT → recover accuracy; apply 50% structured pruning + retrain; compare final-size / accuracy / inference-latency vs original on a held-out test set.	CO3

	Title & Topics	CO
M4	**Edge Inference Frameworks** TensorFlow Lite Micro (TFLM) — the canonical CMSIS-NN-backed inference framework for MCUs; the canonical Pete Warden + Daniel Situnayake approach; canonical "convert TF → TFLite → TFLM" pipeline.; ExecuTorch (Meta, 2024 GA) — the canonical PyTorch-native edge runtime; canonical advantages over TFLM (TorchScript-free, modular backends); canonical mobile + MCU support.; ONNX Runtime — the canonical interop framework; ORT-Mobile, ORT-Web; canonical advantages for cross-framework portability.; Edge Impulse — the canonical end-to-end no-code TinyML platform (data ingestion → DSP block → NN block → deployment); canonical advantages for rapid prototyping; canonical limitations vs custom pipelines.; microTVM (Apache TVM) — the canonical compiler-based edge deployment; auto-tuning for target MCU; canonical 2-5x speedup over hand-optimized kernels.; Vendor-specific frameworks — STMicroelectronics X-CUBE-AI, NXP eIQ, Renesas e-AI Translator, Silicon Labs Machine Learning Toolkit; the canonical "vendor SDK gives best perf on their MCU" pragmatism.; Larger-edge frameworks — TensorRT (NVIDIA Jetson), OpenVINO (Intel), Edge TPU compiler (Google Coral), Hailo Dataflow Compiler; for "edge AI" beyond MCUs.; Hands-on lab — Deploy a person-detection model (MobileNetV2 SSDLite) two ways to an Arduino Nano 33 BLE Sense + OV7670 camera: (a) via TFLM with CMSIS-NN, (b) via Edge Impulse; benchmark frames/sec, RAM usage, accuracy on a custom Indian-pedestrian dataset; profile cycle-counts per layer.	CO4
M5	**Sensors, Signal Processing & On-Device Features** IMU sensors — the canonical MPU-6050 / MPU-9250 / LSM6DSOX / ICM-42670 (accelerometer + gyroscope, optional magnetometer); I2C / SPI interfaces; sampling rate and filtering (low-pass, high-pass, complementary, Kalman).; MEMS microphones — PDM (Pulse Density Modulation) vs I2S (Inter-IC Sound); canonical examples (Knowles SPH0645, ST MP34DT06J, InvenSense ICS-43434); PDM-to-PCM decimation.; Audio feature extraction — framing + windowing (Hamming, Hann), STFT, MFCC (Mel-Frequency Cepstral Coefficients), log-Mel spectrograms; the canonical KWS pipeline (Hello Edge, Zhang et al. 2017).; Camera modules — OV2640 (UXGA, JPEG), OV5640 (5MP, MIPI), HiMax HM01B0 (ultra-low-power, monochrome); image-signal-processing pipeline: demosaicing, white-balance, gamma; the canonical "resize to 96×96 for MCU" pre-processing.; Environmental sensors — BME280 / BME680 (temp + humidity + pressure + VOC), SHT41 (high-accuracy temp/humidity), MQ-series gas sensors, PM2.5 (Plantower, Sensirion SPS30); canonical applications in India air-quality (CPCB, IndiaSpend / Climate Trends data).; Signal processing on the MCU — ARM CMSIS-DSP (FFT, FIR / IIR filters, matrix ops), ESP32-DSP (Espressif library); canonical optimizations for Cortex-M4F / M7 FPU + SIMD.; Sensor fusion — the canonical complementary / Kalman / Madgwick / Mahony filters for orientation; canonical applications (drone IMU, wearable activity recognition).; Hands-on lab — Build a Hindi keyword-spotter on an Arduino Nano 33 BLE Sense: collect Hindi audio for 4 commands ("chalo", "ruko", "daayen", "baayen") + background, extract MFCC features on-device, train a small CNN on a laptop, deploy via TFLM with INT8 quantization; demonstrate live inference and accuracy.	CO5
M6	**Low Power, Security, OTA & TinyMLOps** Low-power design — the canonical sleep / deep-sleep / hibernation modes; wake-on-IMU-tap (LSM6DSOX), wake-on-sound-event (PDM mic threshold), wake-on-image-difference (HM01B0); RTC-driven scheduling.; Energy harvesting — solar (canonical Indian-sun model: 5-7 kWh/m²/day), thermoelectric (Seebeck effect), vibration (piezo), RF (LoRa-WAN, ambient Wi-Fi); canonical PMIC ICs (BQ25570, AEM10941).; Wireless connectivity — BLE 5.x (sub-mW), LoRa / LoRaWAN (10+ km range, mW-class), NB-IoT / LTE-M (telco network), Zigbee 3.0, Thread / Matter (canonical 2023+ smart-home standard); Indian-context: BSNL LoRa, Reliance Jio NB-IoT.; Secure boot — the canonical chain-of-trust from boot ROM → bootloader → firmware → application; ESP32 secure-boot + flash-encryption; STM32 TrustZone; ARM PSA Certified.; OTA (Over-The-Air) updates — the canonical A/B partition scheme (Espressif ESP-IDF OTA, MCUBoot bootloader); delta updates (Bsdiff, courgette); signed firmware (Ed25519, ECDSA); rollback protection.; TinyMLOps — the canonical "MLOps for the edge" emerging discipline; model versioning, fleet management (Edge Impulse, Arduino Cloud, NXP Edge Insights, AWS IoT Greengrass + SageMaker Edge); the canonical "deploy to 1000 devices, monitor drift, push v2" lifecycle.; Indian-context production deployments — the canonical case studies: SatSure (satellite + edge crop analytics), Kissht (rural fintech KYC), CropIn (precision agriculture in 92 countries), Stellapps (dairy farm IoT), Detect Technologies (industrial predictive maintenance).; Final hands-on lab — Build a complete end-to-end TinyML deployment: a solar-powered ESP32-S3 + BME280 + microphone + SX1262 LoRa node that detects "unusual" agricultural-field acoustic events (engine, gunshot, livestock distress) using an INT8 CNN on MFCC features; runs from a 200-mWh solar harvester; reports via LoRaWAN to a Helium / ChirpStack gateway; supports signed OTA updates; produce an end-of-semester field deployment report with energy + accuracy + connectivity measurements over 7 days.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**TinyML — Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers**	Pete Warden & Daniel Situnayake	O'Reilly Media	2019

#	Title	Authors	Publisher	Year
2	**TinyML Cookbook — Combine artificial intelligence and ultra-low-power embedded devices to make the world smarter**	Gian Marco Iodice	Packt Publishing	2023 (2nd Ed.)
3	**AI at the Edge — Solving Real-World Problems with Embedded Machine Learning**	Daniel Situnayake & Jenny Plunkett	O'Reilly Media	2023
4	**Deep Learning for Vision Systems**	Mohamed Elgendy	Manning Publications	2020
5	**Machine Learning Systems — Principles and Practices of Engineering Artificially Intelligent Systems**	Vijay Janapa Reddi (Editor)	Harvard / CS249r open textbook	2024 (continuously updated)
6	**Practical Deep Learning for Cloud, Mobile, and Edge**	Anirudh Koul, Siddha Ganju & Meher Kasam	O'Reilly Media	2019
7	**Embedded Machine Learning for Cyber-Physical, IoT, and Edge Computing**	Sudeep Pasricha & Muhammad Shafique (Eds.)	Springer	2024

Embedded Software Development

25EC2308

L-T-P-S	2-0-0-4	Credits	3	Contact Hrs/Wk	6
Semester	S04	Mode	Regular	Category	PCC
Pre-requisite	EBSD				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply embedded software fundamentals — memory-mapped I/O, register access, interrupt handling, the discipline of working with peripheral datasheets — to write firmware that controls hardware reliably.	BTL3 · Apply
CO2	Apply communication peripherals — UART, SPI, I2C, CAN at intuition — to interface with sensors, actuators, and other MCUs.	BTL3 · Apply
CO3	Apply RTOS fundamentals — tasks, scheduling, semaphores, queues, FreeRTOS / Zephyr APIs — to build multi-tasking firmware.	BTL3 · Apply
CO4	Analyze real-time scheduling — rate-monotonic and earliest-deadline-first at intuition, priority inversion, deadline analysis — to reason about timing-critical systems.	BTL4 · Analyze
CO5	Apply driver development patterns — HAL APIs vs direct register, interrupt-driven I/O, DMA at intuition, the abstraction layers in production firmware.	BTL3 · Apply
CO6	Analyze contemporary embedded development — Rust on MCUs at intuition, edge AI deployment via TensorFlow Lite Micro, OTA updates, secure-boot fundamentals, the AI-augmented embedded workflow.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO6	PSO1
CO1	H	–	H	–	–	H
CO2	H	–	H	–	–	H
CO3	H	–	H	–	–	H
CO4	H	H	H	–	–	H
CO5	H	–	H	–	–	H
CO6	H	–	H	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Embedded Software Foundations** The embedded mental model — firmware running on a small MCU, no OS, direct hardware access; the contrast with application programming.; Memory-mapped I/O — peripheral registers as memory addresses, the volatile keyword, the canonical bug from forgetting it.; Cortex-M architecture at intuition — Thumb-2, NVIC interrupt controller, SysTick timer; the canonical reference platform.; Toolchain fundamentals — gcc-arm-none-eabi, the linker script, the startup file, the canonical build flow.; Modern IDE landscape — STM32CubeIDE, PlatformIO (the modern alternative), Zephyr west; the productivity gain.; Debugging fundamentals — GDB over OpenOCD, the SWD protocol; the canonical debugging workflow with breakpoints and watchpoints.; A first lab: blink an LED on STM32 / ESP32 / NXP via direct register write; observe via oscilloscope; switch to HAL API; compare code clarity.	CO1
M2	**Communication Peripherals** UART — async serial communication, baud rate, framing, the canonical commissioning protocol.; SPI — master-slave synchronous, MOSI/MISO/SCLK/CS, modes 0-3; the canonical fast-and-simple bus.; I2C — multi-master serial, START/STOP/ACK protocol, 7-bit addressing; the canonical sensor bus.; CAN at intuition — automotive-canonical multi-master broadcast bus; depth in automotive-track courses downstream.; USB at intuition — the host-vs-device distinction, USB classes (CDC, HID, MSC); the canonical "USB-to-serial" CDC class.; Modern wireless — BLE via NimBLE / softdevice, WiFi via ESP-IDF; the bridge to FOIS-deeper IoT.; A lab: interface DHT22 via I2C, BMP280 via SPI, ESP32 via UART (between two boards); demonstrate three-protocol simultaneous communication on STM32 / ESP32.	CO2

	Title & Topics	CO
M3	**RTOS Fundamentals** Why RTOS — multi-tasking, deterministic scheduling, ease of writing concurrent firmware; the contrast with bare-metal super-loop.; Tasks and the scheduler — priorities, ready/blocked/running states; the canonical context-switch mechanism.; Synchronization primitives — semaphores (counting, binary, mutex), queues, event groups.; FreeRTOS API — xTaskCreate, vTaskDelay, xQueueSend / xQueueReceive, the canonical patterns.; Zephyr RTOS at intuition — the Linux Foundation backing, the kconfig + DTS approach, why Zephyr is the modern industrial choice.; Stack management — per-task stacks, stack overflow detection, the canonical bug source in embedded multi-tasking.; A lab: in FreeRTOS on STM32 / ESP32, build a 3-task system — sensor task reads data at 10Hz, processing task computes moving average, display task updates a 16×2 LCD; use a queue for inter-task communication.	CO3
M4	**Real-Time Scheduling** Real-time vs general-purpose computing — the deadline-meeting guarantee.; Hard, firm, and soft real-time — the canonical distinction (anti-lock brakes vs MP3 decode vs UI animation).; Rate-monotonic scheduling at intuition — fixed priorities by period; Liu and Layland's schedulability test.; Earliest-deadline-first at intuition — dynamic-priority alternative; the trade-offs.; Priority inversion — the canonical Mars Pathfinder bug; the priority-inheritance and priority-ceiling protocols.; Worst-case execution time analysis at intuition — measurement-based vs static-analysis approaches.; Schedulability analysis — given a set of tasks with periods and worst-case execution times, can they all meet deadlines?; A lab: model a 5-task system in software (no real hardware), apply RMS analysis, verify schedulability; introduce a priority-inversion scenario; observe the deadline miss; fix with priority inheritance.	CO4
M5	**Driver Development Patterns** Driver abstractions — HAL APIs vs LL (low-level) APIs vs direct register; the trade-offs (portability vs performance vs clarity).; Interrupt-driven I/O — UART RX in interrupt mode, SPI completion interrupt, the canonical patterns.; Ring buffers for I/O — the canonical pattern for decoupling interrupt handlers from main code.; DMA at intuition — peripheral-to-memory and memory-to-peripheral, double-buffering, the canonical performance win for high-rate ADC sampling.; The driver-application boundary — a clean API design discipline that pays back during testing and reuse.; Testing embedded code — host-side unit tests on driver logic, hardware-in-the-loop for full validation; the modern testing practices.; A lab: write a clean UART driver with a ring-buffered interrupt-driven RX; expose a putc/getc API to the application; build a small command-line interpreter on top.	CO5
M6	**Modern Embedded — Rust, Edge AI, Security, AI-Augmented Workflow** Rust on MCUs at intuition — the embedded-rust ecosystem; cortex-m / cortex-m-rt; why Rust is the safer alternative for new safety-critical projects.; TensorFlow Lite Micro / TinyML at intuition — running a 50KB neural network on a Cortex-M4 at 80 MHz; the canonical keyword-spotting demo.; OTA (over-the-air) updates — the canonical bootloader + dual-bank pattern; the implications for production IoT deployments.; Secure boot — chain of trust from ROM bootloader to application; the canonical TLS / X.509 use cases for IoT.; Modern observability for firmware — RTT (Real-Time Transfer), Memfault, Tracealyzer; the productivity gains for production debugging.; AI coding assistants for embedded — Claude / Copilot / Cursor for HAL boilerplate, peripheral configuration; the discipline of always testing on hardware (the AI assistant cannot see the board).; When AI assistants fail in embedded — register-bit miscalculation, missed interrupt-clear, missing volatile, compiler-version-specific bugs; the eye students must develop.; A final-week lab: deploy a TFLite Micro keyword-spotting model on ESP32 / STM32 with an I2S MEMS microphone; verify it can detect "yes/no/up/down" with >85% accuracy; measure inference time; submit to GitHub.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Mastering STM32**	Carmine Noviello	Leanpub	2024
2	**Real-Time Embedded Multithreading Using ThreadX**	Edward L. Lamie	CRC Press	2020 (4th Ed.)
3	**Mastering FreeRTOS — The FreeRTOS Real Time Kernel**	Richard Barry	FreeRTOS (free PDF)	2021
4	**Making Embedded Systems — Design Patterns for Great Software**	Elecia White	O'Reilly	2024 (2nd Ed.)
5	**Discovering the STM32 Microcontroller**	Geoffrey Brown	Free PDF (Indiana University)	2016
6	**TinyML — Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers**	Pete Warden & Daniel Situnayake	O'Reilly	2019
7	**The Rust on Embedded Devices Book**	Embedded Working Group	Online (free)	2024

Embedded System Design And Iot

25EC2206

L-T-P-S	2-0-0-4	Credits	3	Contact Hrs/Wk	6
Semester	S03	Mode	Regular	Category	PCC
Pre-requisite	DISD				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply ARM Cortex-M architectural concepts (core pipeline, memory model, NVIC, system tick) to characterise an STM32H7-class microcontroller and its programming model.	BTL3 · Apply
CO2	Apply STM32 peripheral subsystems — GPIO, ADC, timers, PWM, USART, SPI, I²C — through STM32 HAL to interface with sensors and actuators in small embedded programs.	BTL3 · Apply
CO3	Apply interrupt-driven and timer-driven programming patterns to build periodic and event-driven embedded behaviour without an RTOS.	BTL3 · Apply
CO4	Apply discrete-time PID controller design to a small position or velocity control loop, accounting for sampling rate and saturation at an introductory level.	BTL3 · Apply
CO5	Apply PWM-based motor drive concepts and current sensing to characterise a basic brushless or stepper motor drive scheme.	BTL3 · Apply
CO6	Apply CAN bus fundamentals — frame format, arbitration, transceivers — to design a simple node-to-node telemetry protocol on STM32H7 FDCAN.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	M	–	–	–
CO2	H	–	M	M	M
CO3	H	–	M	–	–
CO4	H	M	M	–	–
CO5	H	–	M	–	–
CO6	H	–	M	M	–

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**STM32 Microcontroller Architecture** Microcontroller vs microprocessor; on-chip integration of memory, peripherals, and connectivity.; ARM Cortex-M family overview and the Cortex-M7 in the STM32H7 series.; STM32H7 memory map: flash, ITCM, DTCM, AXI SRAM, peripheral region.; Clock tree at the user level: HSE, HSI, PLL, system clock, peripheral prescalers.; NVIC interrupt model: priority levels, the system tick, vector table.; Boot sequence: from power-on to main(); the role of the startup file.; STM32CubeMX as the configuration entry point; STM32CubeIDE as the development environment.; A first program: blink an LED on a Nucleo-H743ZI board.	CO1
M2	**GPIO, ADC, Timers, PWM** GPIO at the HAL level: input mode, output mode, alternate function, pull-up/pull-down.; Reading buttons and driving LEDs through the HAL.; STM32 timer subsystems: basic, general-purpose, advanced-control timers.; Timer modes: counting, capture, compare, PWM generation.; PWM configuration through STM32CubeMX: frequency, duty cycle, polarity.; STM32H7 ADC: SAR architecture, channels, resolution, sampling time.; Reading an analog sensor (potentiometer, temperature) at a fixed sample rate.; Common GPIO/ADC/PWM pitfalls and how to debug them.	CO2

	Title & Topics	CO
M3	**Interrupts, Timers, and Periodic Tasks** Polling vs interrupts: when each is appropriate.; NVIC priority and pre-emption.; Writing interrupt service routines: do's and don'ts; keeping ISRs short.; Periodic tasks via timer interrupt: building a 1 kHz tick for a control loop.; External interrupts (EXTI): wiring a button or limit switch to a callback.; DMA fundamentals at the introductory level: offloading data movement from the CPU.; Bare-metal periodic-task pattern: state machine driven by a tick interrupt.; When you would want an RTOS — and why this course stays bare-metal (RTOS depth comes in ESDT).	CO3
M4	**Discrete-Time PID Control** Continuous PID at intuition level: P, I, D — what each term contributes.; Discrete-time PID — the implementable form on a microcontroller.; Sampling rate selection: rule of thumb relative to the closed-loop bandwidth.; Output saturation and anti-windup at introductory level.; Tuning by manual adjustment using step responses.; Implementing a PID loop on a 1 kHz timer tick.; Measuring the closed-loop response and verifying stability.; Cascaded control loops at preview level (position outside, velocity inside).	CO4
M5	**Motor Drive Basics** PWM-based motor drive: duty cycle as voltage proxy.; DC motor drive with an H-bridge; direction and speed control.; Brushless DC motor basics: three-phase windings, six-step commutation.; Stepper motor basics: full-step and half-step modes.; Motor driver ICs: integrated H-bridges and three-phase drivers.; Current sensing through a shunt resistor and op-amp.; Safe operating limits: current, temperature, dead-time.; Pairing the PID loop from M4 with a small motor drive.	CO5
M6	**CAN Bus Fundamentals** Why CAN: dominance in automotive and industrial automation; differential signalling and noise immunity.; CAN physical layer: CAN-H/CAN-L, twisted pair, 120 Ω termination, common bit rates.; CAN frame format: identifier, DLC, data field, CRC, ACK.; Priority by identifier and bus arbitration.; CAN transceivers as the interface between the MCU and the bus.; STM32H7 FDCAN peripheral at the HAL level: configuration, transmit, receive.; A simple node-to-node telemetry exchange: periodic state frames between two STM32 boards.; Higher-level protocols (CANopen, J1939) at awareness level — those are application-domain courses, not part of EBSD.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Mastering STM32**	Carmine Noviello	Leanpub	2023 (revised)
2	**Programming with STM32 — Getting Started with the Nucleo Board and C/C++**	Donald Norris	McGraw-Hill	2018
3	**The Definitive Guide to ARM Cortex-M3 and Cortex-M4 Processors**	Joseph Yiu	Newnes	2014 (3rd Ed.)
4	**Embedded Systems — Real-Time Operating Systems for ARM Cortex-M Microcontrollers**	Jonathan Valvano	CreateSpace	2017
5	**Controller Area Network Prototyping with Arduino**	Wilfried Voss	Copperhill Technologies	2014
6	**Electric Motors and Drives — Fundamentals, Types and Applications**	Austin Hughes & Bill Drury	Newnes	2019 (5th Ed.)

Foreign Language

FL

L-T-P-S	3-0-0-0	**Credits**	3	**Contact Hrs/Wk**	3
Semester	S06	**Mode**	Regular	**Category**	HSS
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the script, sounds, and certification format of the chosen language.	BTL2 · Understand
CO2	Apply foundational grammar and politeness levels to form correct sentences.	BTL3 · Apply
CO3	Apply everyday and workplace vocabulary across common situations.	BTL3 · Apply
CO4	Apply listening and speaking for daily and business interactions.	BTL3 · Apply
CO5	Apply business reading and writing for emails, forms, and certification tasks.	BTL3 · Apply
CO6	Analyze the country's culture and business etiquette, and pass a basic-cert mock.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO6	PO9	PO10	PO11	PSO1	PSO2
CO1	–	–	**H**	**H**	M	–
CO2	–	–	**H**	**H**	M	–
CO3	–	–	**H**	M	M	–
CO4	–	**H**	**H**	–	M	–
CO5	–	–	**H**	**H**	M	–
CO6	**H**	–	**H**	–	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Script, Sounds & First Contact** Writing system — German alphabet / Japanese hiragana & katakana / Korean Hangul.; Sounds, pronunciation, and reading aloud.; Greetings, farewells, and politeness formulas.; Introducing yourself and others.; Personal details — asking and giving.; Numbers, dates, and time.; Classroom and survival language.; Basic-certification format overview (JLPT N5 / Goethe A1 / TOPIK I). **Labs:** Script + pronunciation lab with recorded self-assessment.; Self-introduction role-play in a first-day workplace setting. **Case Study:** Case Study — a graduate's first day at a German/Japanese/Korean company: analyse the greetings and self-introduction they must handle in-country.	CO1
M2	**Core Grammar & Politeness Foundations** Nouns, articles/particles, and word order.; Personal pronouns and addressing people.; Present-tense verbs and conjugation.; Essential irregular verbs (to be / to have / to do).; Negation and yes/no & wh-questions.; Politeness levels — Sie/du, -masu/desu, jondaemal.; Plurals/counters and basic particles.; Common beginner errors for Indian learners. **Labs:** Grammar-in-use lab building polite simple sentences about yourself.; Register lab: convert casual sentences to workplace-appropriate forms. **Case Study:** Case Study — formal vs informal register at work: analyse a situation where the wrong politeness level caused offence, and the correct form.	CO2

	Title & Topics	CO
M3	**Everyday & Workplace Vocabulary** Family, people, and relationships.; Food, dining, and shopping.; Time, routine, and schedules.; Directions, transport, and places.; Office, departments, and roles.; Meetings, email, and telephone basics.; Numbers, prices, and money.; Certification core vocabulary lists. **Labs:** Themed-vocabulary lab (everyday + office) with spaced repetition.; Role-play lab: ordering, directions, and a workplace introduction. **Case Study:** Case Study — settling in abroad: a relocated employee handles shopping, transport, and an office introduction — identify the vocabulary each needs.	CO3
M4	**Listening & Speaking** Understanding slow, clear speech.; Everyday conversations and small talk.; Self-introduction in a business setting.; Exchanging contact details and business cards.; Telephone and simple meeting expressions.; Making and responding to requests.; Expressing likes, needs, and preferences.; Certification listening-section practice. **Labs:** Listening lab on basic-cert dialogues with comprehension tasks.; Paired speaking lab: business self-introduction + card exchange. **Case Study:** Case Study — a first client phone/intro call: analyse a simple business-contact exchange in-country and the listening cues that matter.	CO4
M5	**Business Reading & Writing** Reading short texts, signs, and notices.; Reading simple emails and messages.; Writing short professional messages and emails.; Filling forms and profiles.; Tone and politeness in writing.; Connecting words and simple sentences.; Certification reading-section practice.; Certification writing-section practice (where applicable). **Labs:** Reading-comprehension lab on basic-cert texts.; Write-and-revise lab: a short professional email, then critique with/beyond an AI tool. **Case Study:** Case Study — a relocation form / first work email: decode a real beginner-level document and draft an appropriate short reply.	CO5
M6	**Culture, Etiquette & Certification Readiness** Cultural values and communication style of the country.; Business etiquette — greetings, bowing/handshake, business cards.; Hierarchy, seniority, and addressing colleagues.; Punctuality, meetings, and workplace norms.; Dining and after-work culture (Feierabend / nomikai / hoesik).; Do's and don'ts and avoiding offence.; Basic-certification full mock test and strategy.; Continued-learning routine with AI tools. **Labs:** Business-etiquette simulation (greetings, cards, seating) for the chosen country.; Full basic-certification mock test with scoring and review. **Case Study:** Case Study — an etiquette misstep by a foreign employee (mishandling a Japanese business card / over-familiarity in Germany / ignoring Korean seniority): analyse what went wrong and the correct etiquette.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Genki I: An Integrated Course in Elementary Japanese (Textbook), 3rd Ed.**	Eri Banno, Yoko Ikeda, Yutaka Ohno, Chikako Shinagawa & Kyoko Tokashiki	The Japan Times	2020
2	**TRY! JLPT N5: Japanese Language Proficiency Test**	ASK Publishing	ASK Corporation	2014
3	**Netzwerk neu A1: Deutsch als Fremdsprache (Kurs- und Übungsbuch)**	Stefanie Dengler, Tanja Mayr-Sieber, Paul Rusch & Helen Schmitz	Klett	2019
4	**Fit fürs Goethe-Zertifikat A1: Start Deutsch 1**	Evelyn Frey	Hueber Verlag	2019
5	**Integrated Korean: Beginning 1, 3rd Ed.**	Young-mee Cho, Hyo Sang Lee, Carol Schulz, Ho-min Sohn & Sung-Ock Sohn	University of Hawai'i Press	2019
6	**Korean Grammar in Use: Beginning**	Ahn Jean-myung, Lee Kyung-ah & Han Hoo-young	Darakwon	2010
7	**The Culture Map: Breaking Through the Invisible Boundaries of Global Business**	Erin Meyer	PublicAffairs	2014

Full Stack Web Development

25SC2008

L-T-P-S	0-0-2-4	Credits	2	Contact Hrs/Wk	6
Semester	S03	Mode	Regular	Category	ESC
Pre-requisite	DAST				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the full-stack-web architecture — client-server model, HTTP request-response cycle, the canonical layers (browser → frontend framework → API → application server → database) — and trace one user action end-to-end.	BTL2 · Understand
CO2	Apply HTML/CSS/JavaScript foundations — semantic HTML5, modern CSS (Flexbox + Grid + responsive design), vanilla JavaScript (ES2020+), the DOM API, fetch — to build static and dynamic web pages.	BTL3 · Apply
CO3	Apply React fundamentals — functional components with hooks (useState, useEffect), props, conditional rendering, lists with keys, controlled forms — to build interactive single-page-application frontends.	BTL3 · Apply
CO4	Apply server-side Node.js + Express — routing, middleware, REST API design, request validation, error handling, environment configuration — to build production-grade backend services.	BTL3 · Apply
CO5	Apply data persistence — MongoDB via Mongoose (document-style), PostgreSQL via Prisma (relational with type safety), basic CRUD endpoints, schema design discipline — to build full-stack apps with real data.	BTL3 · Apply
CO6	Analyze authentication (JWT, sessions, OAuth at intuition), deployment (Vercel for frontend, Render/Railway for backend, MongoDB Atlas / Supabase for managed databases), the AI-augmented developer workflow (Copilot/Claude/Cursor) — and ship a complete full-stack application to production.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO6	PO9	PSO1
CO1	H	H	–	–	–	–	H
CO2	H	–	H	–	–	–	H
CO3	H	–	H	–	–	–	H
CO4	H	–	H	–	–	–	H
CO5	H	–	H	H	–	–	H
CO6	H	–	H	H	M	M	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The Web Stack and HTML/CSS Foundations** The full-stack architecture: browser (HTML/CSS/JS) → HTTP/HTTPS → backend server → database → response back. The mental model that every full-stack engineer uses.; HTTP at the level a full-stack engineer needs: methods (GET/POST/PUT/DELETE), status codes (2xx/3xx/4xx/5xx), headers (Content-Type, Authorization, Cookie), request/response bodies.; Browser rendering revisited: HTML parsed into DOM, CSS into CSSOM, render tree, layout, paint; why CSS in <head> and JS at end-of-body.; Semantic HTML5: <header>/<nav>/<main>/<aside>/<footer> landmarks, <h1>-<h6> hierarchy, <form> with proper <label>/<input> association, accessibility implications of semantic markup.; Modern CSS layout: the box model, Flexbox (1D layout) with justify-content/align-items, CSS Grid (2D layout) with grid-template-columns/areas.; Responsive design: mobile-first methodology, media queries, fluid typography with clamp(), viewport-relative units (vw, vh, dvh).; CSS custom properties (variables): theming, design tokens, the modern alternative to Sass for most use cases.; A first lab: build a static 3-page responsive site (Home / About / Contact) using semantic HTML + Flexbox + Grid + CSS variables; verify Lighthouse 90+ on Accessibility.	CO1

	Title & Topics	CO
M2	**JavaScript and the DOM** Modern JavaScript essentials (ES2020+): let/const, arrow functions, template literals, destructuring, spread/rest, modules (import/export), optional chaining, nullish coalescing.; Functions revisited: first-class functions, callbacks, higher-order functions (map/filter/reduce); the functional patterns that React relies on.; Promises and async/await: the modern asynchronous model; why callback hell ended; error handling via try/catch with async/await.; Fetch API: the modern HTTP client; GET / POST / PUT / DELETE patterns; the response.ok and response.json() flow.; DOM manipulation in vanilla JS: querySelector / querySelectorAll, element.classList, addEventListener, dataset.; Event handling: the event object, event delegation, removing listeners; the canonical patterns.; Local storage and session storage at intuition level: client-side persistence; the security and size limits.; A lab: build a small to-do app in vanilla JavaScript — fetch from a JSONPlaceholder API, render to DOM, add filter buttons, persist to localStorage; deploy to GitHub Pages.	CO2
M3	**React Component Basics** Why React: declarative ("describe what UI should look like") vs imperative DOM manipulation; render(state) → UI as the central abstraction.; JSX: syntactic sugar over React.createElement; what the bundler does before the browser sees it; the rules (one root element, className not class).; Functional components: function that returns JSX; modern default since hooks (2019); class components mentioned only as legacy.; Props: passing data into a component; props as the component's API; the read-only discipline.; useState hook: local component state; the setState updater pattern; the reducer-style updater function.; useEffect hook: side effects after render; the dependency array; common pitfalls (missing deps, infinite loops).; Conditional rendering and lists: ternaries vs && short-circuit, the .map() pattern, the importance of keys and the canonical bug from missing them.; Controlled forms in React: state owns the input value; onChange updates state; the discipline of controlled inputs.; Project setup with Vite: npm create vite@latest, the dev server with HMR, the production build; the modern alternative to Create React App.; A lab: rebuild the M2 to-do app as a React + Vite project; observe the productivity gain and the cost (no internals visibility); deploy to Vercel.	CO3
M4	**Server-Side Node and Express** Node.js as the JS runtime outside the browser: the event loop, npm as the package manager, package.json as the project descriptor.; Express.js as the canonical web framework: app.use middleware, app.get/post/put/delete routes, req/res objects.; Middleware deeply: the (req, res, next) signature; built-in (express.json), third-party (cors, morgan), custom (auth, logging).; REST API design: resource-oriented URLs (/users, /users/:id), HTTP methods mapped to CRUD operations, response status conventions.; Request validation: zod or joi as the modern schema-first approach; the discipline of validating inputs at the API boundary.; Error handling: the express-async-errors pattern, centralized error middleware, the discipline of returning structured error responses.; Environment configuration: dotenv for .env files, the dev/test/staging/prod separation, secrets management discipline.; tRPC at intuition level: when both frontend and backend are TypeScript, type-safe RPC eliminates an entire class of "the API broke the frontend" bugs; the modern alternative to REST in full-stack TypeScript projects.; A lab: build a Node + Express REST API with 5 endpoints (CRUD on a notes resource), zod validation, centralized error handling; test with curl and Thunder Client / Postman.	CO4
M5	**Data Persistence — MongoDB and Postgres** Why a database is the right answer: persistence across server restarts, query support, integrity guarantees, multi-process safety.; MongoDB at survey level: the document model (collections of JSON-like documents), the BSON storage format, the operators ($eq, $gt, $in).; Mongoose as the canonical Node ODM: schema definition, model creation, CRUD methods, populate for references; when to use it vs raw driver.; PostgreSQL at survey level: the relational model, schema with tables and constraints, SQL queries, JSON columns for flexibility (depth in DBMS).; Prisma as the modern Postgres ORM: schema-first, TypeScript-first, type-safe queries, migrations as code; the 2026-current contrast with classic Sequelize.; Schema design discipline: when to embed vs reference (Mongo), normalisation vs denormalisation (SQL); the patterns that survive scale.; CRUD endpoints in practice: the canonical "GET /resources, POST /resources, GET /resources/:id, PUT /resources/:id, DELETE /resources/:id" mapping with database operations.; A lab: extend the M4 notes API to persist to either MongoDB Atlas (Mongoose) or Supabase Postgres (Prisma); pick one stack per student preference; verify CRUD via Postman; demonstrate data survives server restart.	CO5

	Title & Topics	CO
M6	**Authentication, Deployment, the AI-Augmented Workflow** Authentication fundamentals: identifying who the user is. The canonical schemes — session-based (cookie + server-side store) vs token-based (JWT in Authorization header).; Password handling discipline: NEVER store plaintext, ALWAYS use bcrypt/argon2, the salt-and-hash pattern; the canonical security failures.; JWT (JSON Web Tokens) at depth needed: the three-part structure (header.payload.signature), how the server verifies, the access-token + refresh-token pattern.; OAuth 2.0 at intuition level: "Sign in with Google/GitHub", the authorization-code flow; using libraries (Passport, Auth.js, Clerk) rather than rolling your own.; CORS: the same-origin policy, when CORS is needed, the Express CORS middleware; the canonical deployment-time gotcha.; Frontend deployment: Vercel as the canonical zero-config option for React apps; environment variables, preview deployments per PR.; Backend deployment: Render / Railway / Fly.io for Node + Express; the "git push to deploy" model; the contrast with classic VPS-with-PM2.; Database hosting: MongoDB Atlas free tier, Supabase free tier (Postgres + auth + storage), Neon for serverless Postgres; the managed-DB landscape.; AI coding assistants for full-stack: Copilot / Claude / Cursor for component scaffolding, Express route boilerplate, Prisma schema generation, test writing; the discipline of treating AI output as a first draft requiring review.; When AI assistants fail in full-stack work: subtle auth bugs (missed token validation), missing input sanitization (XSS / SQL injection holes), CORS misconfigs, exposed secrets in committed code; the security review eye that students must develop.; A lab: ship a complete full-stack mini-app — React frontend on Vercel, Node/Express backend on Render, MongoDB Atlas database, JWT auth, deployed to a real public URL; a 5-minute peer-review session before final submission.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Eloquent JavaScript — A Modern Introduction to Programming**	Marijn Haverbeke	No Starch Press / Free PDF	2024 (4th Ed.)
2	**Learning React — Modern Patterns for Developing React Apps**	Alex Banks & Eve Porcello	O'Reilly	2020 (2nd Ed.)
3	**Node.js Design Patterns — Design and implement production-grade Node.js applications**	Mario Casciaro & Luciano Mammino	Packt	2020 (3rd Ed.)
4	**Express in Action — Writing, building, and testing Node.js applications**	Evan M. Hahn	Manning	2016
5	**MongoDB — The Definitive Guide — Powerful and Scalable Data Storage**	Shannon Bradshaw, Eoin Brazil & Kristina Chodorow	O'Reilly	2019 (3rd Ed.)
6	**Full Stack React — A Practical Guide to Building Real-World React Applications**	Anthony Accomazzo, Ari Lerner, Nate Murray, Clay Allsopp, David Guttman & Tyler McGinnis	Fullstack.io	2017
7	**Designing Web APIs — Building APIs That Developers Love**	Brenda Jin, Saurabh Sahni & Amir Shevat	O'Reilly	2018

Fundamentals Of Iot And Sensors

25EC1203

L-T-P-S	2-0-2-0	Credits	3	Contact Hrs/Wk	4
Semester	S02	Mode	Regular	Category	ESC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand IoT system architecture — the four-layer model (perception/sensing → network → middleware/cloud → application) — and place real IoT systems in this canonical structure.	BTL2 · Understand
CO2	Apply sensor and actuator selection — categories (analog/digital, contact/non-contact), specifications (range, accuracy, sampling rate), interfacing (ADC, GPIO, I2C, SPI) — to choose the right component for a given application.	BTL3 · Apply
CO3	Apply microcontroller basics on a contemporary platform (Arduino / ESP32 / Raspberry Pi Pico) — GPIO, ADC, PWM, timers, basic embedded-C / MicroPython programming — to wire and read sensors.	BTL3 · Apply
CO4	Apply IoT communication protocols — Wi-Fi, Bluetooth/BLE, LoRa, Zigbee, MQTT, HTTP/REST — to send sensor data from device to cloud.	BTL3 · Apply
CO5	Apply cloud-IoT integration — AWS IoT Core / Azure IoT Hub / ThingsBoard at intuition, sensor-data dashboards, basic device management — to build complete IoT pipelines.	BTL3 · Apply
CO6	Analyze IoT security and emerging trends — common attack categories (device hijacking, MITM, supply chain), TinyML at intuition, edge AI, the convergence with 5G — and reason about the practical engineer's defense and design choices.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO6	PO8	PSO1
CO1	H	H	–	–	–	–	H
CO2	H	–	H	–	–	–	H
CO3	H	–	H	–	–	–	H
CO4	H	–	H	–	–	–	H
CO5	H	–	H	H	–	–	H
CO6	H	–	H	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**IoT System Architecture** What IoT is — the canonical definition: physical objects connected to the internet, sensing/acting on the world, exchanging data.; The four-layer reference architecture — perception (sensors), network (connectivity), middleware (cloud platform), application (user-facing).; IoT vs traditional embedded — the connectivity dimension that changes everything; security and update implications.; IoT scale at 2026 — the canonical projection numbers (50 billion+ devices); the implications for engineering.; IoT verticals — consumer (smart home), industrial (Industry 4.0), agricultural, healthcare, smart-city; the canonical use cases per vertical.; A first lab: tour 3 real IoT products on campus or at home (Wi-Fi router, smart speaker, fitness tracker); identify which components handle which layer.	CO1
M2	**Sensors and Actuators** Sensor categories — temperature, humidity, pressure, light, motion (accelerometer/gyro), gas, sound, image; the canonical parts per category.; Sensor specifications — range, accuracy, resolution, sampling rate, response time, drift; reading datasheets.; Analog vs digital sensors — when each type fits; the role of ADC for analog sensors.; Actuators — LEDs, motors (DC, servo, stepper), relays, buzzers; the categories and selection criteria.; Sensor interfacing — voltage levels, pull-up/pull-down resistors, signal conditioning at intuition.; Communication interfaces — analog (ADC), digital GPIO, I2C, SPI, UART; the canonical use cases for each.; A lab: wire a DHT22 temperature/humidity sensor and a servo motor to a microcontroller; read sensor and drive actuator based on threshold.	CO2

	Title & Topics	CO
M3	**Microcontroller Basics** Microcontroller vs microprocessor — the system-on-chip distinction; the canonical architectures (8-bit AVR Arduino, 32-bit ARM Cortex-M).; The development board landscape — Arduino UNO/Nano (educational standard), ESP32 (Wi-Fi+BLE), Raspberry Pi Pico (RP2040), STM32 Blue Pill (depth in EBSD).; GPIO — digital input/output, configuration, reading button states, driving LEDs.; ADC — reading analog sensors (potentiometer, LDR, thermistor); resolution and reference voltage.; PWM — pulse-width modulation for variable LED brightness, motor speed control.; Timers and interrupts at intuition — the difference from busy-wait; the canonical low-power IoT pattern.; Programming approaches — Arduino C/C++ (the dominant teaching language), MicroPython (the modern alternative on ESP32/RP2040).; A lab: program ESP32 (or equivalent) to read 4 sensors via different interfaces (analog, digital, I2C, SPI) and log to serial monitor; demonstrate timer-based sampling at 10Hz.	CO3
M4	**IoT Communication Protocols** Wireless personal-area networks — Bluetooth Classic vs BLE; ranges and use cases.; Wireless local-area networks — Wi-Fi standards (802.11n/ac/ax) at intuition; security modes.; Wireless wide-area networks — LoRa/LoRaWAN (low-power long-range), NB-IoT (cellular IoT), Sigfox; when each wins.; Mesh networking at intuition — Zigbee, Thread (Matter); the canonical smart-home stack.; Application-layer protocols — HTTP/REST (familiar but heavy), MQTT (the canonical pub-sub IoT protocol), CoAP at intuition.; MQTT mechanics — broker/client architecture, topics, QoS levels; why MQTT dominated IoT.; Modern IoT-protocol context — Matter as the unified smart-home protocol from 2024+; the implications for cross-vendor compatibility.; A lab: program ESP32 to publish sensor readings to a free public MQTT broker (test.mosquitto.org or HiveMQ); subscribe via mobile app or laptop; observe pub-sub mechanics.	CO4
M5	**Cloud-IoT Integration** Why cloud — long-term storage, analytics, dashboards, multi-device coordination.; AWS IoT Core at intuition — devices register, X.509 certs for auth, MQTT-based ingestion, IoT Rules for routing data downstream.; Alternatives — Azure IoT Hub, Google Cloud IoT Core (deprecated 2023, important to know); ThingsBoard as open-source alternative.; Time-series databases — InfluxDB as the canonical OSS option; AWS Timestream, Azure Time Series Insights as managed alternatives.; Dashboards — Grafana with InfluxDB (the canonical OSS combo); ThingsBoard built-in; AWS QuickSight.; Device management — over-the-air updates, device provisioning, remote configuration; the production-grade concerns.; A lab: deploy a free-tier ThingsBoard instance; provision an ESP32 device; ingest live sensor data; build a dashboard with current readings + 24-hour history; share publicly.	CO5
M6	**IoT Security and Emerging Trends** IoT security threat landscape — Mirai botnet as the canonical wakeup call; the Indian IoT-security context.; Common attack categories — default-password device hijacking, unencrypted communications (MITM), supply-chain compromises, denial-of-service.; Defense fundamentals — TLS for transport encryption, X.509 certs for device identity, secure boot, signed firmware updates.; TinyML at intuition — running ML models on MCU-class devices; sample sizes (kilobytes), use cases (keyword spotting, anomaly detection); depth in EATM.; Edge AI vs cloud AI — when to compute where; latency, privacy, bandwidth trade-offs.; 5G + IoT — network slicing for IoT, ultra-reliable-low-latency for industrial; depth in WT5G.; AI-augmented IoT development — Claude / ChatGPT for circuit/code design assistance; the discipline of always testing on hardware.; A final-week lab: secure the M5 ThingsBoard pipeline — enable TLS, rotate device credentials, demonstrate a man-in-the-middle attack succeeds without TLS and fails with TLS enabled; write a 1-page security review of your own deployment.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Internet of Things — A Hands-On Approach**	Arshdeep Bahga & Vijay Madisetti	Universities Press	2015
2	**Designing the Internet of Things**	Adrian McEwen & Hakim Cassimally	Wiley	2014
3	**Internet of Things — Architectures, Protocols and Standards**	Simone Cirani, Gianluigi Ferrari, Marco Picone & Luca Veltri	Wiley	2018
4	**Hands-On Internet of Things with MQTT — Build Connected IoT Devices Using MQTT, Raspberry Pi, and Arduino**	Tim Pulver	Packt	2018
5	**Practical Internet of Things Security — A Practical Guide to Securing the IoT Era**	Brian Russell & Drew Van Duren	Packt	2018 (2nd Ed.)
6	**TinyML — Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers**	Pete Warden & Daniel Situnayake	O'Reilly	2019

#	Title	Authors	Publisher	Year
7	**Industrial Internet of Things — Cybermanufacturing Systems**	Sabina Jeschke, Christian Brecher, Houbing Song & Danda B. Rawat	Springer	2017

Generative Ai And Ai Platform Engineering

25CS2207

L-T-P-S	2-0-0-4	Credits	3	Contact Hrs/Wk	6
Semester	S05	Mode	Regular	Category	PCC
Pre-requisite	AIML				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the GenAI landscape — foundation models, the transformer architecture at intuition level, the API tier (GPT/Claude/Gemini) vs self-hosted tier (Llama/Qwen/Mistral) — and place modern GenAI in the broader AI lineage from AIML.	BTL2 · Understand
CO2	Apply prompting techniques — zero-shot, few-shot, chain-of-thought, structured output via JSON schemas / function calling, system prompts, the prompt-as-code discipline — to ship GenAI features at production fluency.	BTL3 · Apply
CO3	Apply embeddings and Retrieval-Augmented Generation — sentence-transformers, vector databases (pgvector / Pinecone / Weaviate / Qdrant), chunking strategies, hybrid search (dense + BM25), reranking — to ground LLM outputs in trusted knowledge.	BTL3 · Apply
CO4	Apply API-level fine-tuning — when fine-tuning beats prompting, OpenAI / Anthropic fine-tuning APIs, Parameter-Efficient Fine-Tuning (PEFT) at intuition level (LoRA, QLoRA), dataset preparation discipline — to specialize models for narrow tasks.	BTL3 · Apply
CO5	Apply agentic patterns at survey level — function calling, ReAct loops, tool use, the canonical multi-agent frameworks (LangGraph, AutoGen) at intuition level — and reason about when an agent is the right architecture vs a simple LLM call.	BTL3 · Apply
CO6	Analyze evaluation, safety, cost, and the AI-engineer role — capability evals, harm evals, prompt injection and the canonical attack categories, observability with LangSmith / Helicone, the API cost-discipline, the API-vs-self-hosted trade-offs — to ship GenAI responsibly in production.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO6	PO8	PSO1
CO1	H	H	–	–	–	–	H
CO2	H	–	H	–	–	–	H
CO3	H	–	H	H	–	–	H
CO4	H	–	H	–	–	–	H
CO5	H	–	H	–	–	–	H
CO6	H	H	H	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The GenAI Landscape and Foundation Models** Defining GenAI: models that generate new content (text, code, images, audio, video) rather than predict labels; the contrast with classical ML from AIML.; Foundation models: massive pre-trained models (GPT, Claude, Gemini, Llama, Qwen, Mistral) usable for many downstream tasks; the "foundation" terminology from the 2021 Stanford paper.; Brief transformer history: Attention Is All You Need (2017), GPT-1 (2018), BERT (2018), GPT-3 (2020), ChatGPT (Nov 2022), the post-2022 explosion.; The transformer architecture at intuition level: tokenisation → embedding → attention layers → unembedding → next-token prediction; depth lives in NLPL.; Capability tiers in 2026: small (1-3B params, runs on consumer hardware), mid (7-70B params, runs on H100s), large (300B+ params, API-only); the cost-vs-capability frontier.; The API tier landscape: OpenAI (GPT-4 family), Anthropic (Claude family), Google (Gemini family), Cohere, Mistral; rough capability/cost positioning as of 2026.; The self-hosted tier landscape: Llama (Meta), Qwen (Alibaba), Mistral (open weights), Phi (Microsoft, small); when self-hosted wins (privacy, cost at scale, customization).; A first lab: call three different LLMs (GPT, Claude, Gemini) on the same prompts via their APIs; compare quality, cost, latency; document the differences.	CO1

	Title & Topics	CO
M2	**Prompting and Structured Generation** Prompting as engineering, not magic: the prompt-is-code discipline; version control, testing, regression detection.; Zero-shot prompting: the simplest pattern; instruction + input → output; works for tasks the model already knows.; Few-shot prompting: instruction + 2-5 examples + input; the "in-context learning" capability; when few-shot beats zero-shot.; Chain-of-thought prompting: "let's think step by step"; the reasoning-trace technique; documented to improve accuracy on math and logic tasks.; System prompts: the role/persona/constraints layer; how providers separate it from user input; best-practice patterns.; Structured output: JSON schemas, function calling (OpenAI), tool use (Anthropic); the production-grade alternative to "parse the response with a regex".; Prompt patterns: ReAct (reason + act), Reflexion (self-critique), Tree-of-Thoughts (branch and search); when each helps.; Prompt-as-code workflow: prompts in version-controlled files, evals run on every change, the cultural practice that distinguishes shipping AI features from prototyping them.; A lab: write a prompt that classifies Indian-language tweets into 5 emotion categories; iterate from zero-shot to few-shot to chain-of-thought; build a 50-example eval set; document the accuracy improvement at each stage.	CO2
M3	**Embeddings and Retrieval-Augmented Generation** The motivation for RAG: LLMs hallucinate; LLMs have a knowledge cutoff; LLMs can't access your private data. RAG fixes all three by grounding responses in retrieved trusted context.; Embeddings as the foundation: vector representations of text; semantically-similar texts have nearby vectors; the cosine-similarity metric.; Embedding model landscape in 2026: OpenAI text-embedding-3 family, Anthropic's embedding models, open-source sentence-transformers (all-MiniLM, BGE, E5); the cost-vs-quality frontier.; Vector databases: pgvector (PostgreSQL extension), Pinecone (managed), Weaviate (open-source), Qdrant (open-source), Chroma (developer-friendly); when each fits.; Chunking strategies: fixed-size, sentence-boundary, recursive-character, semantic; the trade-offs in retrieval precision.; The RAG retrieval-step pipeline: embed query → ANN search in vector DB → top-K retrieval → optional reranking → augment prompt → generate.; Hybrid search: dense (embeddings) + sparse (BM25) → reciprocal-rank fusion; why hybrid usually beats dense-alone in production.; Reranking: a smaller cross-encoder model rescores top-50 results to find the true top-5; the latency-quality trade-off.; A lab: build a RAG over 50 PDFs (Indian government schemes / RBI circulars / KLU academic regulations); use sentence-transformers + Qdrant + GPT-4 / Claude; evaluate on 20 questions with known answers; report retrieval precision and answer quality.	CO3
M4	**Fine-Tuning at the API Level** When fine-tuning beats prompting: narrow domain with consistent format, large training set available, prompt-engineering has plateaued, latency / cost matters at scale.; When fine-tuning is the wrong tool: the task changes frequently, you don't have labelled data, the base model already does it well; the canonical anti-patterns.; OpenAI's fine-tuning API: dataset format (JSONL with prompt-completion pairs), the upload + training + deployment flow, the cost model.; Anthropic and Google fine-tuning at preview level: the platform-specific differences and similarities.; Parameter-Efficient Fine-Tuning (PEFT) at intuition level: instead of updating all 7B parameters, update a small adapter (LoRA) of ~10M parameters; depth lives in NLPL.; LoRA and QLoRA at intuition level: low-rank decomposition + quantised base model; how an enthusiast fine-tunes a 7B model on a single consumer GPU.; Dataset preparation discipline: the 80/15/5 split, deduplication, format consistency, the canonical bugs (label leakage, train/test contamination).; Evaluation of fine-tuned models: domain-specific eval set, comparison against the base model, the regression-tracking discipline.; A lab: fine-tune GPT-3.5-class via OpenAI's API on a small Indian-context dataset (e.g., Indian-language sentiment, legal-clause classification, code-comment translation); compare against the un-fine-tuned base on a held-out test set.	CO4
M5	**Agentic Patterns and Tool Use** What an agent is: an LLM that decides what to do next based on observations, takes actions via tools, observes results, repeats until done.; Function calling / tool use as the foundational primitive: schemas describe what each tool does, the LLM decides when to call which tool with what arguments.; The ReAct loop: Reason → Act → Observe → repeat; the canonical agent control flow.; Single-agent vs multi-agent: when one agent with many tools is enough vs when separating roles (planner / executor / critic) helps.; Multi-agent frameworks at survey level: LangGraph (state-machine-based), AutoGen (conversation-based), CrewAI (role-based); the dominant 2026 options. Depth in AAAE.; When agents are the right architecture: open-ended tasks, tool composition required, dynamic decision-making.; When agents are the wrong tool: deterministic workflows (just write code), simple QA (just use RAG), latency-critical paths (every step costs an LLM call).; Agent observability: tracing every LLM call, tool call, decision; LangSmith / Helicone / Langfuse as the canonical tools.; A lab: build a small ReAct agent that answers questions about a small Indian dataset (e.g., RBI policy rates) by composing 3 tools — search, calculator, table lookup; trace every step in LangSmith.	CO5

	Title & Topics	CO
M6	**Evaluation, Safety, Cost, and the AI Engineer Role** Why evaluation is hard: GenAI outputs are not ground-truthable like classification accuracy; the evaluation tooling lags the model capability.; Capability evaluation: domain-specific eval sets, LLM-as-judge with caveats (positional bias, length bias, self-preference), human-in-the-loop spot checks.; Harm evaluation: bias (BBQ, CrowS-Pairs at preview), toxicity, the canonical jailbreak categories (DAN-style, role-play, encoding tricks); responsible-deployment discipline.; Prompt injection at depth: direct injection ("ignore previous instructions"), indirect injection (instructions hidden in retrieved documents); the defence-in-depth approach.; Cost discipline: every API call costs; the cost-per-feature analysis; caching identical prompts, prompt caching (Anthropic), batch processing (50% discount on OpenAI).; Latency optimisation: streaming responses for perceived speed, parallelisation when possible, model selection (cheap-and-fast for simple tasks, expensive-and-good for hard ones).; Observability in production: per-feature cost dashboard, latency distributions, error rates, hallucination signals, drift detection; the AI-engineer's SRE practice.; API vs self-hosted decision: the four axes (cost, latency, privacy, customization); the matrix that decides; the realistic Indian-startup reality (start with API, move to self-hosted only when scale or compliance forces it).; The AI-engineer role in 2026: the shift from "build model" to "compose and operate AI features"; the emerging title "AI engineer" distinct from "ML engineer"; what this means for placement positioning.; A final-week lab: take the RAG system from M3, add evals (10 questions × 3 quality dimensions × LLM-as-judge), instrument with LangSmith, build a small cost-and-latency dashboard, ship it to a real public URL via Vercel.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**AI Engineering — Building Applications with Foundation Models**	Chip Huyen	O'Reilly	2024
2	**Hands-On Large Language Models — Language Understanding and Generation**	Jay Alammar & Maarten Grootendorst	O'Reilly	2024
3	**Building LLMs for Production — Enhancing LLM Abilities and Reliability with Prompting, Fine-Tuning, and RAG**	Louis-François Bouchard & Louie Peters	Towards AI	2024
4	**Generative Deep Learning — Teaching Machines to Paint, Write, Compose, and Play**	David Foster	O'Reilly	2023 (2nd Ed.)
5	**Designing Machine Learning Systems — An Iterative Process for Production-Ready Applications**	Chip Huyen	O'Reilly	2022
6	**Natural Language Processing with Transformers — Building Language Applications with Hugging Face**	Lewis Tunstall, Leandro von Werra & Thomas Wolf	O'Reilly	2022 (Revised Ed.)
7	**LangChain in Action**	Ben Auffarth	Manning	2024

Human Values, Gender Equality & Professional Ethics

25UC0026

L-T-P-S	2-0-0-0	Credits	2	Contact Hrs/Wk	2
Semester	S07	Mode	Regular	Category	HSS
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand human values and their role in technology and society.	BTL2 · Understand
CO2	Apply ethical frameworks to technology design and professional practice.	BTL3 · Apply
CO3	Analyze gender and diversity inclusion in technology and the workplace.	BTL4 · Analyze
CO4	Analyze technology-and-society impacts — access, equity, and harm.	BTL4 · Analyze
CO5	Analyze AI ethics, algorithmic fairness, and data rights.	BTL4 · Analyze
CO6	Apply inclusive, value-sensitive design to real technology problems.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO5	PO6	PO7	PO8	PSO1	PSO2
CO1	–	M	–	**H**	M	–
CO2	–	**H**	–	**H**	M	–
CO3	–	**H**	–	**H**	–	M
CO4	–	**H**	M	–	–	M
CO5	M	–	–	**H**	–	M
CO6	–	**H**	–	**H**	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Human Values in a Technological World** Why values matter for technologists.; Self-exploration and natural acceptance.; Human aspirations and well-being.; Values in relationships and teams.; Harmony in society and nature.; Technology and the good life.; Responsibility and conscience.; From values to professional conduct. **Labs:** Values-reflection journal linking personal values to tech decisions.; Group exercise: define a value charter for a technology team. **Case Study:** Case Study — a values-driven technologist's choice: analyse a real case where an engineer's values shaped a technology decision and its societal impact.	CO1
M2	**Ethics for Technologists** Moral theories applied to technology.; Professional codes (IEEE, ACM, IEI).; Responsibility, accountability, and risk.; Honesty, integrity, and conflicts of interest.; Whistleblowing in tech.; Intellectual property and fair use.; Ethical decision-making frameworks.; Moral courage in organisations. **Labs:** Apply a professional code to a real engineering-ethics scenario.; Ethical-dilemma debate using competing moral frameworks. **Case Study:** Case Study — a tech-industry ethics case (e.g., a whistleblower or a product-safety cover-up): analyse the ethical failures and the codes that applied.	CO2

	Title & Topics	CO
M3	**Gender & Diversity Inclusion in Technology** Gender gaps in tech and STEM.; Stereotypes, bias, and the 'pipeline'.; Inclusive hiring and retention.; The PoSH Act and safe workplaces.; Intersectionality and diverse identities.; Allyship and inclusive culture.; Accessibility and disability inclusion.; Measuring and improving inclusion. **Labs:** Audit a workplace/product scenario for inclusion gaps and remedies.; Design an inclusive-team or inclusive-hiring policy. **Case Study:** Case Study — diversity in a tech company (a real DEI success or failure): analyse what drove or blocked inclusion and the business and ethical stakes.	CO3
M4	**Technology, Society & Equity** Technology's social impact.; The digital divide and access.; Technology and inequality.; Surveillance and civil liberties.; Misinformation and platform harms.; Technology for social good.; Frugal and inclusive innovation.; Engineer's social responsibility. **Labs:** Map the equity/access impacts of a technology on different groups.; Design a tech-for-good intervention for an underserved community. **Case Study:** Case Study — a digital-divide / tech-for-good case in India (e.g., access gaps in a digital service): analyse who is excluded and how inclusive design could help.	CO4
M5	**AI Ethics, Fairness & Data Rights** Ethics of AI and automation.; Algorithmic bias and fairness.; Transparency and explainability.; Data privacy and the DPDP Act 2023.; Consent, surveillance, and autonomy.; Accountability for AI harms.; Responsible-AI principles.; Governance and regulation. **Labs:** Conduct a fairness/bias + privacy audit of a real AI use case.; Draft responsible-AI guidelines for a technology project. **Case Study:** Case Study — a biased AI system (e.g., facial recognition or automated decisioning): analyse the harm, the fairness failure, and the principles that should have governed it.	CO5
M6	**Inclusive & Value-Sensitive Design** Value-sensitive design.; Inclusive and universal design.; Designing for accessibility.; Participatory and co-design.; Ethics-by-design and privacy-by-design.; Stakeholder values in design.; Trade-offs and responsible choices.; Embedding values in the product lifecycle. **Labs:** Redesign a digital product for inclusion and accessibility.; Run a value-sensitive-design workshop on a real technology brief. **Case Study:** Case Study — inclusive design done well (a product redesigned for accessibility/inclusion): analyse how value-sensitive design changed who the product served.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**A Foundation Course in Human Values and Professional Ethics, 2nd Revised Ed.**	R. R. Gaur, R. Asthana & G. P. Bagaria	Excel Books	2019
2	**Ethics in Engineering, 4th Ed.**	Mike W. Martin & Roland Schinzinger	McGraw-Hill	2005
3	**Engineering Ethics: Concepts and Cases, 5th Ed.**	Charles E. Harris, Michael S. Pritchard & Michael J. Rabins	Cengage Learning	2013
4	**Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy**	Cathy O'Neil	Crown	2016
5	**Design Justice: Community-Led Practices to Build the Worlds We Need**	Sasha Costanza-Chock	MIT Press	2020
6	**Invisible Women: Data Bias in a World Designed for Men**	Caroline Criado Perez	Abrams Press	2019
7	**Seeing Like a Feminist**	Nivedita Menon	Penguin/Zubaan	2012

Indian Ancient Knowledge Systems - Vedic Mathematics

25UC0017

L-T-P-S	2-0-0-0	**Credits**	0	**Contact Hrs/Wk**	2
Semester	S01	**Mode**	Regular	**Category**	AUC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the scope of Indian Knowledge Systems and their contributions.	BTL2 · Understand
CO2	Understand the foundations and philosophy of Vedic Mathematics and its sutras.	BTL2 · Understand
CO3	Apply Vedic arithmetic techniques for fast, accurate computation.	BTL3 · Apply
CO4	Apply Vedic methods to algebra and problem-solving.	BTL3 · Apply
CO5	Understand India's broader contributions to mathematics and science.	BTL2 · Understand
CO6	Analyze the relevance of IKS and Vedic Mathematics in the modern AI era.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO5	PO6	PO11	PSO1	PSO2
CO1	–	–	–	**H**	**H**	M	–
CO2	**H**	–	–	–	M	M	–
CO3	**H**	M	–	–	–	M	–
CO4	**H**	**H**	–	–	–	M	–
CO5	–	–	–	**H**	**H**	M	–
CO6	–	–	M	–	**H**	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Introduction to Indian Knowledge Systems** What Indian Knowledge Systems are.; Scope and classification of IKS.; Oral and written knowledge traditions.; The Vedic corpus and Vedangas.; Knowledge domains — science, math, arts, medicine.; Gurukula and learning traditions.; Why IKS matters today.; IKS in the national curriculum (NEP 2020). **Labs:** Survey and map domains of IKS with examples for each.; Research-and-present lab on one Indian knowledge contribution. **Case Study:** Case Study — a rediscovered Indian contribution (e.g., the concept of zero / ancient metallurgy of the Iron Pillar): analyse the contribution and how it shaped the world.	CO1
M2	**Vedic Mathematics Foundations** Origin — Jagadguru Bharati Krishna Tirthaji.; The 16 Sutras and 13 sub-sutras.; Philosophy and coherence of the system.; Place value and number sense.; Complements and the 'all from 9, last from 10'.; Mental-math principles.; Advantages — speed and accuracy.; When Vedic methods help most. **Labs:** Drill lab: apply 'all from 9, last from 10' to subtraction problems.; Speed-vs-accuracy lab comparing conventional and Vedic methods. **Case Study:** Case Study — mental-math champions / competitive-exam aspirants: analyse how Vedic techniques give a real edge in speed and accuracy under exam conditions.	CO2

	Title & Topics	CO
M3	**Vedic Arithmetic Techniques** Multiplication by Nikhilam.; Multiplication by Urdhva-Tiryagbhyam (vertically and crosswise).; Multiplication special cases.; Division techniques.; Squares and square roots.; Cubes and cube roots.; Divisibility tests.; Digit sums and checking answers. **Labs:** Multiplication lab using Nikhilam and Urdhva-Tiryagbhyam.; Squares/cubes and roots lab with self-timed problems. **Case Study:** Case Study — large-number multiplication under time pressure: students solve a set conventionally and via sutras, and analyse the time/accuracy difference.	CO3
M4	**Vedic Algebra & Applications** Vedic approach to algebra.; Linear equations via sutras.; Simultaneous equations.; Factorisation techniques.; Quadratic insights.; Applications in aptitude and competitive exams.; Cross-checking with digit sums.; Building a personal technique toolkit. **Labs:** Equation-solving lab applying Vedic algebra sutras.; Aptitude lab: solve a timed competitive-style set using Vedic methods. **Case Study:** Case Study — a competitive-exam quantitative section (CAT/GATE/bank): analyse how a Vedic toolkit can shave time on real problem types.	CO4
M5	**India's Contributions to Mathematics & Science** The decimal system and zero.; Aryabhata, Brahmagupta, Bhaskara.; The Kerala school and infinite series.; Sulba Sutras and geometry.; Indian astronomy.; Ancient metallurgy and chemistry.; Ayurveda and life sciences (overview).; Linguistics and Panini's grammar. **Labs:** Timeline lab mapping Indian mathematical/scientific milestones.; Deep-dive presentation on one Indian mathematician/scientist. **Case Study:** Case Study — the Kerala school's pre-Newtonian series: analyse how Indian mathematicians anticipated calculus concepts centuries earlier.	CO5
M6	**IKS & Vedic Mathematics in the AI Era** Algorithmic thinking in the sutras.; Vedic methods and computing/digital logic.; IKS and modern STEM education.; Preservation and digitisation of texts.; AI for ancient-text analysis and translation.; Heritage, ethics, and authenticity.; Evidence-based vs exaggerated claims.; Carrying IKS forward responsibly. **Labs:** Express a Vedic sutra as an algorithm/flowchart and discuss its logic.; Critically evaluate a popular IKS claim against evidence. **Case Study:** Case Study — AI applied to ancient Indian texts (digitisation/translation of manuscripts): analyse how modern AI helps preserve and study IKS, and the pitfalls.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Vedic Mathematics**	Jagadguru Swami Sri Bharati Krishna Tirthaji Maharaja	Motilal Banarsidass	2015
2	**Introduction to Indian Knowledge System: Concepts and Applications**	B. Mahadevan, Vinayak Rajat Bhat & Nagendra Pavana R. N.	PHI Learning	2022
3	**Vedic Mathematics Made Easy**	Dhaval Bathia	Jaico Publishing House	2005
4	**The Power of Vedic Maths, 2nd Ed.**	Atul Gupta	Jaico Publishing House	2004
5	**Mathematics in India: 500 BCE–1800 CE**	Kim Plofker	Princeton University Press	2009
6	**Pride of India: A Glimpse into India's Scientific Heritage**	Samskrita Bharati	Samskrita Bharati	2006
7	**A Concise History of Science in India, 2nd Ed.**	D. M. Bose, S. N. Sen & B. V. Subbarayappa (Eds.)	Universities Press	2009

Indian Constitution

25UC0008

L-T-P-S	2-0-0-0	**Credits**	0	**Contact Hrs/Wk**	2
Semester	S08	**Mode**	Regular	**Category**	AUC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the Constitution as the practical foundation of the laws governing professional life.	BTL2 · Understand
CO2	Apply fundamental rights to real personal, civic, and workplace situations.	BTL3 · Apply
CO3	Analyze equality, anti-discrimination, and anti-harassment protections at work.	BTL4 · Analyze
CO4	Analyze employee rights and obligations under constitutionally-grounded labour and contract law.	BTL4 · Analyze
CO5	Analyze digital-age rights — privacy, data protection, IP, and speech — for a working engineer.	BTL4 · Analyze
CO6	Apply mechanisms to access justice and use the law for personal and collective benefit.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO6	PO8	PO10	PO11	PSO2
CO1	**H**	M	–	–	M
CO2	**H**	**H**	–	–	M
CO3	**H**	**H**	–	–	M
CO4	**H**	–	–	M	M
CO5	**H**	**H**	–	–	M
CO6	**H**	–	M	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The Constitution as Your Foundation** What the Constitution is and why it matters to you personally.; Preamble values — justice, liberty, equality, fraternity — in everyday and work life.; The Constitution as a living, enforceable document (not trivia).; Rule of law and constitutional supremacy.; How constitutional law flows down into the laws that govern your job.; Fundamental Rights, Directive Principles, and Duties — the practical map.; How to find, read, and make sense of a law in plain language.; Your identity and responsibilities as a citizen-professional. **Labs:** Map Preamble values to real personal and workplace decisions.; 'Find the law' lab: locate and plain-language summarise the law behind a given right. **Case Study:** Case Study — an ordinary person who invoked a constitutional right to fix a real problem (e.g., a worker securing dignity or livelihood): trace how a Preamble value became an enforceable outcome.	CO1
M2	**Your Fundamental Rights in Practice** Right to equality (Art 14–18) — fairness and non-arbitrariness in daily life.; Right to freedom (Art 19) — speech, profession, movement — and reasonable restrictions.; Right to life and personal liberty (Art 21) — dignity, livelihood, health, privacy.; Right against exploitation (Art 23–24) — forced and child labour.; Freedom of religion and cultural/educational rights (in brief).; Rights against the State vs against private employers.; Right to constitutional remedies (Art 32) — the writs, in practice.; How and where to actually seek a remedy. **Labs:** Scenario lab: match real situations to the fundamental right that applies.; Outline a writ/grievance for a rights-violation scenario. **Case Study:** Case Study — the expansion of Art 21 (e.g., right to livelihood / privacy / dignity): analyse how a broad reading of one right now protects working people.	CO2

	Title & Topics	CO
M3	**Equality, Anti-Discrimination & Anti-Harassment at Work** Equality applied to hiring, pay, and promotion.; Anti-discrimination — gender, caste, religion, disability.; Equal pay for equal work.; Sexual harassment and the PoSH Act 2013 (from Vishakha to statute).; The Internal Committee — your rights and the complaint process.; Rights of persons with disabilities (RPwD Act 2016).; Maternity and related benefits and their constitutional basis.; Demanding and building an inclusive workplace. **Labs:** PoSH complaint walkthrough: draft and route a complaint correctly.; Review a workplace anti-discrimination/inclusion policy for gaps. **Case Study:** Case Study — a real workplace harassment/discrimination matter and its resolution: analyse the rights invoked, the PoSH/IC process, and the outcome.	CO3
M4	**Your Rights & Duties as an Employee** Constitutional basis of labour welfare (DPSP — Art 38, 39, 41, 42, 43).; The new Labour Codes — Wages, Industrial Relations, Social Security, OSH.; Employment contracts — what to check before you sign.; Wages, hours, leave, PF, ESI, and gratuity.; NDAs, non-compete, and employment bonds — what is enforceable vs not (Art 19 freedom of profession).; Termination, notice, and recourse against wrongful dismissal.; Whistleblower protection.; Trade unions and collective rights. **Labs:** Review a sample employment contract and flag the red-flag clauses.; Draft a professional grievance/representation letter for a workplace issue. **Case Study:** Case Study — an employment-bond / non-compete dispute (common for fresh engineers): analyse what the law actually enforces and how an employee protected themselves.	CO4
M5	**The Engineer-Citizen in the Digital Age** Right to privacy (Puttaswamy 2017) — what it means for you and your data.; The DPDP Act 2023 — your rights as a data principal and duties as a professional handling data.; Intellectual property at work — who owns what you create (work-for-hire), patents, copyright.; Freedom of speech online — limits, defamation, and employer social-media policies.; IT Act essentials — cyber offences, your protection and your liability.; Consumer rights (Consumer Protection Act 2019) as professional and consumer.; Contract law essentials (Indian Contract Act) for everyday agreements.; Where professional ethics meets legal duty. **Labs:** Draft a data-rights (DPDP) request and a basic privacy-notice check.; Analyse an IP-ownership / NDA clause and explain its real effect. **Case Study:** Case Study — a data-privacy or IP-ownership dispute relevant to a tech employee: analyse who held the rights, what the law said, and how it was resolved.	CO5
M6	**Accessing Justice & Using the Law for Your Benefit** The court and tribunal hierarchy — which forum for which dispute.; Consumer, labour, civil, and writ forums in practice.; Alternative dispute resolution — arbitration, mediation, Lok Adalat.; Right to Information (RTI) — how to actually use it.; Legal aid and your right to it.; Filing complaints — police, and commissions (NHRC, women's, SC/ST, consumer).; Public Interest Litigation for collective benefit.; Practical legal literacy — notices, FIRs, affidavits, and staying law-aware. **Labs:** File a mock RTI application for a real information need.; Choose-the-right-forum simulation: route given disputes to the correct remedy. **Case Study:** Case Study — someone who used RTI / a consumer forum / a tribunal to secure their rights: analyse the path they took and why it worked.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Introduction to the Constitution of India, 25th Ed.**	Durga Das Basu	LexisNexis	2018
2	**The Indian Constitution (Oxford India Short Introductions)**	Madhav Khosla	Oxford University Press	2012
3	**Labour and Industrial Laws**	S. N. Misra	Central Law Publications	2021
4	**Information Technology Law and Practice, 5th Ed.**	Vakul Sharma	LexisNexis	2021
5	**Intellectual Property Rights, 2nd Ed.**	V. K. Ahuja	LexisNexis	2017
6	**Law of Consumer Protection: Principles and Practice**	Avtar Singh	Eastern Book Company	2020
7	**Privacy 3.0: Unlocking Our Data-Driven Future**	Rahul Matthan	HarperCollins India	2018

Industrial Automation, Plc & Scada Systems

25EE3207

L-T-P-S	3-0-2-0	Credits	4	Contact Hrs/Wk	5
Semester	S06	Mode	Regular	Category	PCC
Pre-requisite	CSEA				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze the industrial-automation pyramid — field, control, supervisory and enterprise levels — and the role of each in a modern automated plant.	BTL4 · Analyze
CO2	Apply sensor and actuator interfacing — discrete and analog field devices, signal conditioning, and standard 4–20 mA / industrial signalling — to acquire and act on plant data.	BTL3 · Apply
CO3	Apply PLC programming using IEC 61131-3 languages (ladder logic, function block, structured text) to implement discrete and sequential control.	BTL3 · Apply
CO4	Apply HMI and SCADA design — operator screens, alarms, trends and data logging — to supervise and operate an automated process.	BTL3 · Apply
CO5	Analyze industrial communication networks and protocols (Modbus, PROFIBUS/PROFINET, EtherNet/IP, OPC UA) and select appropriate connectivity for an automation system.	BTL4 · Analyze
CO6	Analyze Industry 4.0 and OT/IT convergence — IIoT, digital twins, MES, and OT cybersecurity — and their application to smart manufacturing.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO7	PSO1
CO1	H	H	–	–	–	H
CO2	H	–	H	–	–	H
CO3	H	–	H	–	–	H
CO4	–	–	H	H	–	H
CO5	H	–	–	H	–	H
CO6	–	H	–	–	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The Automation Pyramid** What industrial automation is and why it matters (productivity, safety, quality).; The automation pyramid: field, control (PLC/DCS), supervisory (SCADA), MES, ERP.; PLC vs DCS vs PAC — when each is used.; Discrete vs process vs batch automation.; A tour of a real automated process (conveyor/bottling/water-treatment example).; The role of the control engineer / system integrator.; Standards landscape: IEC 61131, ISA-88/95, IEC 62443.; Framing the course project process to automate.	CO1
M2	**Sensors, Actuators and Field Interfacing** Discrete sensors: proximity, limit, photoelectric.; Analog sensors: temperature, pressure, flow, level.; The 4–20 mA current loop and industrial signal standards.; Actuators: solenoids, relays, contactors, motor starters, valves.; Variable-frequency drives as automation actuators (drive internals owned by EMDE).; Signal conditioning and isolation.; Wiring, grounding and noise in the plant environment.; Interfacing field devices to PLC I/O.	CO2
M3	**PLC Programming (IEC 61131-3)** PLC architecture: CPU, I/O, scan cycle, memory.; Ladder logic: contacts, coils, latches, the scan-based execution model.; Timers and counters.; Function block diagram and structured text.; Sequential function charts for sequential processes.; PID function blocks in the PLC (using CSEA tuning knowledge).; Program structure, modularity and good practice.; Programming the project process control logic.	CO3

	Title & Topics	CO
M4	**HMI and SCADA** The HMI: operator screens, graphics, navigation.; SCADA architecture: RTUs, master station, historian.; Alarms and events; alarm management best practice.; Trends, data logging and the historian.; Recipes and batch control (ISA-88 concept).; Redundancy and reliability in SCADA.; Security of HMI/SCADA (intro; deep OT security in M6).; Building the HMI/SCADA for the project process.	CO4
M5	**Industrial Communication Networks** Why fieldbuses: deterministic, robust plant communication.; Serial fieldbuses: Modbus RTU, PROFIBUS.; Industrial Ethernet: PROFINET, EtherNet/IP, Modbus TCP.; OPC and OPC UA — the interoperability and IT/OT bridge standard.; Time-sensitive networking (concept).; Network topology, determinism and redundancy.; Wireless in industrial settings (concept).; Networking the project PLC, HMI and field devices.	CO5
M6	**Industry 4.0 and OT/IT Convergence** The Industry 4.0 vision and India's SAMARTH Udyog initiative.; Industrial IoT (IIoT): edge gateways, cloud connectivity.; Digital twins of processes and machines.; Manufacturing Execution Systems (MES) and ISA-95.; Predictive maintenance and analytics on plant data.; OT cybersecurity: IEC 62443, the Purdue model, segmentation.; The smart factory and the role of AI.; Adding an IIoT/analytics layer to the project for the demo.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Programmable Logic Controllers**	Frank D. Petruzella	McGraw-Hill	2016 (5th Ed.)
2	**Programmable Logic Controllers**	W. Bolton	Newnes	2015 (6th Ed.)
3	**Industrial Automation: Hands-On**	Frank Lamb	McGraw-Hill	2013
4	**SCADA: Supervisory Control and Data Acquisition**	Stuart A. Boyer	ISA	2009 (4th Ed.)
5	**Automating Manufacturing Systems with PLCs**	Hugh Jack	Open-source (Jack)	2010

Insights To Sustainable Development Goals

25UC0032

L-T-P-S	2-0-0-0	Credits	0	Contact Hrs/Wk	2
Semester	1S	Mode	Regular	Category	AUC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand sustainable development — concept, pillars, and the road to the SDGs.	BTL2 · Understand
CO2	Understand the 17 SDGs, their targets, and their interlinkages.	BTL2 · Understand
CO3	Analyze the people-and-prosperity SDGs in the Indian and global context.	BTL4 · Analyze
CO4	Analyze the planet SDGs — climate, energy, water, and ecosystems.	BTL4 · Analyze
CO5	Apply SDG measurement, governance, and localisation.	BTL3 · Apply
CO6	Analyze how technology, AI, and innovation can accelerate the SDGs.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO5	PO6	PO7	PO11	PSO1	PSO2
CO1	–	M	**H**	–	M	–
CO2	–	**H**	**H**	–	M	–
CO3	–	**H**	M	–	–	M
CO4	–	M	**H**	–	–	M
CO5	–	–	**H**	M	–	M
CO6	**H**	–	**H**	–	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Foundations of Sustainable Development** What sustainable development means.; The Brundtland definition.; The three pillars — economic, social, environmental.; Planetary boundaries.; From MDGs to SDGs.; The 2030 Agenda and 'leave no one behind'.; Sustainability vs growth debates.; India's sustainability context. **Labs:** Map a real project against the three pillars of sustainability.; Footprint lab: estimate and reduce a sustainability footprint. **Case Study:** Case Study — a development-vs-environment trade-off (e.g., a dam/infrastructure project): analyse how the three pillars of sustainability were balanced or violated.	CO1
M2	**The 17 SDGs Overview** Overview of all 17 goals.; Targets and indicators.; The SDG 'wedding cake' model.; Interlinkages and trade-offs among goals.; Synergies across goals.; 'Leave no one behind'.; The global indicator framework.; Tracking progress. **Labs:** Build an SDG interlinkage map for a chosen development problem.; Indicator lab: interpret SDG progress data for a country/state. **Case Study:** Case Study — SDG interlinkages in practice (e.g., clean energy enabling health and education): analyse how progress on one goal advances others.	CO2

	Title & Topics	CO
M3	**People & Prosperity SDGs** No poverty and zero hunger.; Good health and well-being.; Quality education.; Gender equality.; Decent work and economic growth.; Reduced inequalities.; Industry, innovation, and infrastructure.; India's progress on people/prosperity goals. **Labs:** Analyse a region's people/prosperity indicators and gaps.; Design an intervention targeting a people/prosperity SDG. **Case Study:** Case Study — an Indian social initiative (e.g., a health/education/livelihood program): analyse its SDG contribution and outcomes.	CO3
M4	**Planet SDGs** Clean water and sanitation.; Affordable and clean energy.; Climate action.; Life below water.; Life on land.; Sustainable cities and communities.; Responsible consumption and production.; Circular economy. **Labs:** Audit a setting for responsible consumption and propose circular fixes.; Design a planet-SDG (energy/water/waste) solution for a community. **Case Study:** Case Study — a clean-energy or water initiative in India (e.g., solar mission / river rejuvenation): analyse its planet-SDG impact and scalability.	CO4
M5	**Partnership, Governance & Measurement** Peace, justice, and strong institutions (SDG16).; Partnerships for the goals (SDG17).; Financing the SDGs.; Indicators and the SDG India Index.; Localising the SDGs.; Voluntary National Reviews.; Role of governments, business, and civil society.; Accountability and reporting. **Labs:** Build an SDG scorecard for an organisation/community.; Design a multi-stakeholder partnership for an SDG target. **Case Study:** Case Study — the SDG India Index / a state's SDG localisation: analyse how SDGs are measured and localised, and what drives improvement.	CO5
M6	**Technology, AI & Innovation for the SDGs** Technology as an SDG accelerator.; AI for good — use cases across goals.; Social entrepreneurship and impact ventures.; Frugal and inclusive innovation.; Digital public infrastructure for development.; Measuring impact (theory of change).; The engineer's and youth's role.; Designing an SDG-aligned solution. **Labs:** Design a technology/AI-enabled solution mapped to an SDG with a theory of change.; Pitch a social-impact venture targeting one or more SDGs. **Case Study:** Case Study — AI/technology for an SDG (e.g., AI in healthcare access or precision agriculture): analyse how technology accelerates a specific goal, with its risks.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**The Age of Sustainable Development**	Jeffrey D. Sachs	Columbia University Press	2015
2	**Doughnut Economics: Seven Ways to Think Like a 21st-Century Economist**	Kate Raworth	Random House Business	2017
3	**An Introduction to Sustainable Development, 4th Ed.**	Jennifer A. Elliott	Routledge	2012
4	**Sustainability: Principles and Practice, 2nd Ed.**	Margaret Robertson	Routledge	2017
5	**Prosperity without Growth: Foundations for the Economy of Tomorrow, 2nd Ed.**	Tim Jackson	Routledge	2017
6	**Our Common Future**	World Commission on Environment and Development (Brundtland Commission)	Oxford University Press	1987
7	**Factfulness: Ten Reasons We're Wrong About the World**	Hans Rosling with Ola Rosling & Anna Rosling Rönnlund	Flatiron Books	2018

Iot Protocols And Cloud Integration

25EC2309

L-T-P-S	2-0-2-0	Credits	3	Contact Hrs/Wk	4
Semester	S04	Mode	Regular	Category	PCC
Pre-requisite	EBSD				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand IoT-protocol stacks — physical/MAC layers (LoRa, BLE, Wi-Fi, Zigbee), network and transport (IPv6, 6LoWPAN, UDP), application (MQTT, CoAP, HTTP) — and place modern IoT systems in this canonical structure.	BTL2 · Understand
CO2	Apply LPWAN protocols — LoRaWAN at depth, NB-IoT and LTE-M at intuition, Sigfox at preview — to long-range low-power IoT applications.	BTL3 · Apply
CO3	Apply MQTT at production depth — broker / client architecture, topics, QoS levels, retained messages, last-will-and-testament, security via TLS + X.509.	BTL3 · Apply
CO4	Apply cloud-IoT integration — AWS IoT Core / Azure IoT Hub / ThingsBoard, device provisioning, fleet management, OTA updates, time-series ingestion (TimescaleDB / InfluxDB) — to ship complete pipelines.	BTL3 · Apply
CO5	Analyze IoT security — device identity (X.509), TLS, secure provisioning, OTA security, the Mirai-class threat models, the modern zero-trust patterns.	BTL4 · Analyze
CO6	Analyze edge AI integration, 5G + IoT convergence, digital twins at intuition, the AI-augmented IoT design workflow — to architect 2026-current IoT systems.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO6	PO8	PSO1
CO1	H	H	–	–	–	–	H
CO2	H	–	H	–	–	–	H
CO3	H	–	H	–	–	–	H
CO4	H	–	H	H	–	–	H
CO5	H	–	H	–	H	H	H
CO6	H	–	H	H	–	–	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**IoT Protocol Stacks** The IoT-protocol stack — analogous to OSI but with the IoT-specific layer choices; the canonical reference architecture.; PHY/MAC layers — LoRa (long-range), BLE (short-range), Wi-Fi (medium), Zigbee/Thread (mesh); range and power trade-offs.; 6LoWPAN — IPv6 over Low-power Wireless PAN; how IPv6 addresses fit in IEEE 802.15.4 frames.; Transport — UDP (default for IoT), TCP (when reliability matters), QUIC at preview.; Application protocols — MQTT (the dominant IoT pub-sub), CoAP (RESTful for constrained devices), HTTP (familiar but heavy).; Comparing protocol choices — the canonical decision matrix (range, power, throughput, latency, cost).; A first lab: in Wireshark, capture and identify MQTT, CoAP, and HTTP packets from a sample IoT device; observe overhead differences.	CO1
M2	**LPWAN — LoRaWAN at Depth** LPWAN motivation — battery-powered devices, 5-15 km range, no Wi-Fi assumption; the use cases (agricultural, asset tracking, smart meters).; LoRa modulation at intuition — chirp spread spectrum; the canonical SF7-SF12 spreading factors and the range-vs-data-rate trade-off.; LoRaWAN architecture — devices → gateway → network server → application server; the asymmetric topology.; LoRaWAN device classes — Class A (asynchronous, lowest power), Class B (scheduled), Class C (always-listening); when each fits.; LoRaWAN security — AppSKey + NwkSKey, OTAA vs ABP activation; the canonical X.509-style key management.; NB-IoT and LTE-M at intuition — cellular alternatives via existing networks; Indian operators (Jio, Airtel) deployments.; A lab: deploy two ESP32 LoRa boards in a private LoRaWAN network via ChirpStack on a Raspberry Pi; demonstrate uplink data at SF7 (close range) and SF12 (long range, slow).	CO2

	Title & Topics	CO
M3	**MQTT at Production Depth** MQTT recap — broker / client pub-sub architecture, the canonical Mosquitto broker.; Topics — hierarchical naming (sensors/floor1/temp), wildcards (+, #) at intuition, the canonical naming-convention discipline.; QoS levels — 0 (at-most-once), 1 (at-least-once), 2 (exactly-once); the overhead trade-offs.; Retained messages — last-known-good values for new subscribers; the canonical "device status" pattern.; Last-will-and-testament — broker publishes message on client unexpected disconnect; the canonical "device offline" detection.; MQTT security — TLS for transport, X.509 client certificates for identity, ACL for topic-level authorization.; Production MQTT brokers — HiveMQ, EMQX, AWS IoT Core (MQTT-compatible); the multi-tenant scale concerns.; A lab: deploy Mosquitto with TLS + client certs on a free-tier cloud VM; provision 3 ESP32 clients with their own X.509 identities; demonstrate publish-subscribe with QoS 1; demonstrate that without the cert, connection is rejected.	CO3
M4	**Cloud-IoT Integration** AWS IoT Core at depth — device registry, X.509-based auth, IoT Rules engine for routing data, IoT Jobs for fleet management.; Azure IoT Hub at intuition — the Microsoft alternative; device twin pattern.; GCP IoT Core deprecated 2023 — the canonical "lock-in risk" cautionary tale; the migration to ClearBlade.; ThingsBoard at depth — the open-source alternative; deployable on free-tier cloud VM; canonical for Indian-context deployments where data residency matters.; Time-series databases — TimescaleDB (Postgres extension), InfluxDB (the canonical OSS option), AWS Timestream as managed alternative.; Dashboards — Grafana with InfluxDB or TimescaleDB; the canonical OSS combination.; Device fleet management — provisioning, OTA updates, remote configuration, health monitoring.; A lab: provision 5 ESP32 devices on a free-tier ThingsBoard instance; ingest live sensor data; build dashboard with current readings + 24-hour history; demonstrate OTA firmware update of one device.	CO4
M5	**IoT Security in Practice** IoT threat landscape — Mirai botnet (2016) as the canonical wakeup; the Indian IoT security context (mass-deployed cheap devices, weak defaults).; Common attack categories — default-password device hijacking, unencrypted communications (MITM), supply-chain compromises, denial-of-service amplification.; Device identity — X.509 client certificates as the modern standard; secure-element chips (ATECC608A, NXP SE050) for hardware-rooted identity.; Secure boot and trusted firmware — chain of trust from ROM bootloader to application; the canonical implementation patterns.; OTA security — signed firmware images, anti-rollback, the canonical "what if the OTA channel is compromised?" attack scenarios.; TLS for IoT at depth — TLS 1.3, the implementation footprint on Cortex-M4 (mbedTLS, wolfSSL), the cipher-suite choices for constrained devices.; Zero-trust IoT — never trust the network, always verify the device; the post-perimeter security model.; A lab: secure the M4 ThingsBoard pipeline — enable TLS, rotate device credentials, demonstrate a man-in-the-middle attack succeeds without TLS and fails with TLS enabled; write a 1-page security review.	CO5
M6	**Edge AI, 5G+IoT, Digital Twins** Edge AI motivation — latency, privacy, bandwidth, offline operation; the trade-offs vs cloud AI.; TFLite Micro at depth — the canonical 50KB-class model for keyword spotting, anomaly detection, basic image classification on Cortex-M4.; TinyML deployment workflow — train in Python with TensorFlow, convert to TFLite, quantize, deploy to MCU.; 5G + IoT — network slicing for IoT, ultra-reliable-low-latency for industrial; mMTC (massive machine-type) slice.; Digital twins at intuition — the canonical mirror-physical-asset-in-cloud pattern; Azure Digital Twins, AWS IoT TwinMaker.; AIoT (AI + IoT) — the convergence; sensor-data → ML inference → actuation as the canonical industrial pattern.; AI-augmented IoT design — Claude / ChatGPT for protocol-decision brainstorming, schema design, dashboard SQL; the discipline of always testing on real devices.; A final-week lab: deploy a TFLite Micro anomaly-detection model on ESP32 reading vibration data from accelerometer; transmit via LoRaWAN to ThingsBoard; observe normal-vs-anomaly classification on dashboard; document end-to-end latency.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Internet of Things — A Hands-On Approach**	Arshdeep Bahga & Vijay Madisetti	Universities Press	2015
2	**Hands-On Internet of Things with MQTT**	Tim Pulver	Packt	2018
3	**LoRa and LoRaWAN — A Technical Overview**	Andreas Spiess	LoRa Alliance / Online (free)	2024
4	**Practical Internet of Things Security**	Brian Russell & Drew Van Duren	Packt	2018 (2nd Ed.)
5	**TinyML — Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers**	Pete Warden & Daniel Situnayake	O'Reilly	2019
6	**Industrial Internet of Things — Cybermanufacturing Systems**	Sabina Jeschke et al.	Springer	2017

#	Title	Authors	Publisher	Year
7	**Architecting the Industrial Internet — The Architect's Guide to Designing Industrial Internet Solutions**	Shyam Nath & Robert Stackowiak	Packt	2017

Language Skills For Engineers

25UC1102

L-T-P-S	0-0-4-0	Credits	2	Contact Hrs/Wk	4
Semester	S01	Mode	Regular	Category	HSS
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand and apply core English grammar and sentence construction accurately.	BTL2 · Understand
CO2	Apply vocabulary-building strategies to use precise technical and academic language.	BTL3 · Apply
CO3	Apply reading strategies to comprehend and critically engage with technical texts.	BTL3 · Apply
CO4	Apply writing skills to produce clear, coherent paragraphs, emails, and descriptions.	BTL3 · Apply
CO5	Apply listening and speaking skills for fluent, confident communication.	BTL3 · Apply
CO6	Analyze and use AI-era language tools critically, ethically, and effectively.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	–	–	**H**	**H**	M	–
CO2	–	–	**H**	**H**	M	–
CO3	–	–	**H**	**H**	M	–
CO4	–	M	**H**	–	M	–
CO5	–	**H**	**H**	–	M	–
CO6	**H**	–	**H**	–	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Foundations of English for Engineers** Parts of speech and sentence elements.; Sentence types and construction.; Tenses and their use.; Articles, prepositions, and determiners.; Subject-verb agreement.; Active and passive voice.; Common grammatical errors of Indian learners.; Punctuation and mechanics. **Labs:** Diagnostic test + targeted error-correction worksheet on your own writing.; Rewrite a flawed technical paragraph for grammatical accuracy and clarity. **Case Study:** Case Study — a real engineering email gone wrong: analyse an actual workplace email whose grammar/ambiguity caused a costly misunderstanding, and rewrite it correctly.	CO1
M2	**Vocabulary & Word Power** Word formation — roots, prefixes, suffixes.; Technical and academic vocabulary.; Synonyms, antonyms, and shades of meaning.; Collocations and word partnerships.; Idioms and phrasal verbs in professional use.; Context clues for unknown words.; Homophones and confusing pairs.; Building a personal vocabulary log. **Labs:** Build and present a domain-specific vocabulary bank with usage examples.; Word-formation lab: derive families of words from engineering roots. **Case Study:** Case Study — vocabulary in technical specifications: examine how an imprecise word choice in a product spec/datasheet changed meaning, and the cost of imprecision.	CO2
M3	**Reading Skills** Skimming for gist.; Scanning for specific information.; Intensive vs extensive reading.; Reading comprehension strategies.; Making inferences and predictions.; Reading technical and research texts.; Critical reading and evaluating arguments.; Note-making from reading. **Labs:** Timed skimming/scanning lab on a technical article with comprehension Q&A.; Critical-reading lab: evaluate the argument and evidence in an op-ed. **Case Study:** Case Study — reading a real research abstract/technical manual: students extract purpose, method, and findings under time pressure, then compare interpretations.	CO3

	Title & Topics	CO
M4	**Writing Skills** The writing process — plan, draft, revise.; Paragraph structure and topic sentences.; Coherence and cohesion (linkers).; Technical descriptions and definitions.; Emails and formal correspondence.; Summarising and paraphrasing.; Editing and proofreading.; Avoiding plagiarism. **Labs:** Write, peer-review, and revise a 200-word technical description.; Draft a formal email for a real scenario and edit it to professional standard. **Case Study:** Case Study — a strong vs weak technical description: compare two descriptions of the same device/process and identify what makes one clear and the other confusing.	CO4
M5	**Listening & Speaking** Active and critical listening.; Listening for note-making.; English sounds and pronunciation.; Word and sentence stress, intonation.; Fluency and accuracy balance.; Everyday and academic conversation.; Introduction to presentations.; Overcoming hesitation and L1 interference. **Labs:** Pronunciation + intonation practice lab with recorded self-assessment.; Pair conversation/role-play lab on an engineering topic with peer feedback. **Case Study:** Case Study — a recorded technical talk/podcast: students listen, note key points, and discuss how delivery affected understanding.	CO5
M6	**Language in the AI Era** Digital and multimodal literacy.; Clear writing for global, remote teams.; AI writing and grammar assistants — strengths and limits.; Prompt clarity as a language skill.; Critical evaluation of AI-generated text.; Ethics, attribution, and academic integrity with AI.; Avoiding over-reliance and skill loss.; Language for the international workplace. **Labs:** AI-tool lab: draft, critique, and ethically revise a document using and then beyond an AI assistant.; Compare human vs AI revision of the same text and report the differences. **Case Study:** Case Study — AI-assisted writing in practice: analyse an AI-drafted email/report, find its errors and tone problems, and improve it — debating responsible use.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Communication Skills**	Sanjay Kumar & Pushp Lata	Oxford University Press	2015
2	**English Grammar in Use, 5th Ed.**	Raymond Murphy	Cambridge University Press	2019
3	**Technical Communication: Principles and Practice, 3rd Ed.**	Meenakshi Raman & Sangeeta Sharma	Oxford University Press	2015
4	**Practical English Usage, 4th Ed.**	Michael Swan	Oxford University Press	2016
5	**Word Power Made Easy**	Norman Lewis	Goyal Publishers	2011
6	**On Writing Well: The Classic Guide to Writing Nonfiction**	William Zinsser	Harper Perennial	2006
7	**The Elements of Style, 4th Ed.**	William Strunk Jr. & E. B. White	Pearson/Longman	1999

Linear Algebra And Calculus For Engineers

25MT1001

L-T-P-S	2-2-0-0	Credits	4	Contact Hrs/Wk	4
Semester	S02	Mode	Regular	Category	BSC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply matrix operations and linear systems — multiplication, transposition, inverse, rank, Gauss-Jordan elimination — to solve engineering systems of linear equations.	BTL3 · Apply
CO2	Analyze vector spaces, eigenvalues, and eigenvectors — basis, dimension, characteristic equation, diagonalization — to characterize linear transformations.	BTL4 · Analyze
CO3	Apply single-variable calculus — limits, derivatives, integrals, the fundamental theorem, applications (rates, optimization) — to engineering problems.	BTL3 · Apply
CO4	Apply multivariable calculus — partial derivatives, gradients, multiple integrals, line integrals, Green's/Stokes'/Divergence theorems — to engineering scalar and vector fields.	BTL3 · Apply
CO5	Apply ordinary differential equations — first-order, second-order linear, undetermined coefficients, Laplace transforms — to model engineering dynamic systems.	BTL3 · Apply
CO6	Apply numerical methods — Newton-Raphson, Simpson's rule, Runge-Kutta, modern computational tools (NumPy/SciPy/MATLAB) — to engineering problems where analytical solutions are intractable.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	H	–	–	H
CO2	H	H	–	–	H
CO3	H	H	–	–	H
CO4	H	H	–	–	H
CO5	H	H	H	–	H
CO6	H	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Matrices and Linear Systems** Matrices and matrix arithmetic — addition, subtraction, scalar multiplication, matrix multiplication; the row-by-column rule.; Matrix transpose and properties; symmetric, skew-symmetric, orthogonal matrices.; Determinants — definition for 2×2 and 3×3, properties, the Laplace cofactor expansion at intuition.; Matrix inverse — when it exists, computation via adjoint or Gauss-Jordan, the canonical engineering use cases.; Linear systems Ax = b — solving via Gauss-Jordan elimination, the existence-and-uniqueness conditions.; Rank of a matrix — definition, computation via row reduction, the connection to solvability.; A first lab: in MATLAB or Python (NumPy), set up and solve 5×5 engineering linear systems (canonical examples — truss analysis, electrical circuit nodal analysis); verify by hand for small cases.	CO1
M2	**Vector Spaces, Eigenvalues, Eigenvectors** Vector spaces — definition, examples (R^n, function spaces); subspaces; basis and dimension.; Linear independence and span — the canonical tests; how to find a basis from a spanning set.; Linear transformations — matrix representation; null space, range; rank-nullity theorem.; Eigenvalues and eigenvectors — definition, geometric interpretation, characteristic equation.; Diagonalization — when a matrix is diagonalizable; the canonical computation.; Symmetric matrices and the spectral theorem — orthogonal diagonalization; the engineering applications (vibration modes, principal stress).; A lab: compute eigendecomposition of mass-stiffness matrix for a 3-DOF spring-mass system in MATLAB; identify natural frequencies and mode shapes; visualize.	CO2

	Title & Topics	CO
M3	**Single-Variable Calculus** Limits and continuity — formal definition (epsilon-delta), the canonical limit laws.; Derivatives — definition, rules (product, quotient, chain), implicit differentiation.; Applications of derivatives — rates of change, optimization (max/min), Taylor series, L'Hopital's rule.; Integrals — definite and indefinite, the fundamental theorem of calculus, integration techniques (substitution, by parts, partial fractions).; Improper integrals at intuition — when convergence matters in engineering applications.; Sequences and series at intuition — convergence tests, power series, Taylor and Maclaurin series for engineering approximation.; A lab: in Python or MATLAB, find the maximum of a real engineering function (e.g., transmitter-power vs distance trade-off) via Newton's method; verify by symbolic differentiation.	CO3
M4	**Multivariable Calculus** Functions of several variables — partial derivatives, gradient, directional derivatives.; Total differential and applications — error propagation in measurements; the canonical engineering uncertainty-analysis use.; Chain rule for multivariable functions — the canonical applications in physical systems.; Multiple integrals — double and triple integrals, change of variables (polar, cylindrical, spherical), Jacobians.; Line integrals and conservative vector fields; Green's theorem in the plane.; Surface integrals; Stokes' theorem and the Divergence theorem at intuition; the unifying view.; A lab: compute moment of inertia and center of mass of an irregular plate via double integration in Python (sympy or NumPy with quadrature).	CO4
M5	**Ordinary Differential Equations** First-order ODEs — separable, exact, linear, integrating factor; the canonical engineering examples (RC circuit, cooling Newton's law).; Second-order linear ODEs with constant coefficients — homogeneous and non-homogeneous; characteristic equation method.; Method of undetermined coefficients — for non-homogeneous ODEs with standard forcing.; Variation of parameters at intuition — when undetermined coefficients fails.; Laplace transforms — definition, table of common transforms, inverse Laplace, applications to ODEs with discontinuous forcing.; Engineering applications — RLC circuit transient analysis, mass-spring-damper system, control system step response.; A lab: solve mass-spring-damper system response to step input via Laplace transform method; verify against numerical ODE solver in MATLAB or Python.	CO5
M6	**Numerical Methods and Modern Tools** Why numerical methods — the reality that most engineering problems lack closed-form solutions.; Root finding — bisection, Newton-Raphson, secant method; convergence rates and limitations.; Numerical integration — trapezoidal rule, Simpson's rule, Gaussian quadrature at intuition.; Numerical solution of ODEs — Euler's method, improved Euler, Runge-Kutta methods (RK4 as the canonical workhorse).; Numerical linear algebra — LU decomposition, iterative methods (Jacobi, Gauss-Seidel) at intuition.; Modern tools — NumPy/SciPy as the engineering Python stack; MATLAB as the historical alternative; Julia as the emerging high-performance contender.; AI-assisted numerical work — Claude / Copilot for generating numerical scripts; the discipline of verifying generated code against analytical solutions or convergence diagnostics.; A final-week lab: solve a real engineering problem (chosen by branch) using all four canonical methods (root finding + integration + ODE + linear system); compare numerical vs analytical solutions where both exist.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Advanced Engineering Mathematics**	Erwin Kreyszig	Wiley	2022 (10th Ed.)
2	**Linear Algebra and Its Applications**	Gilbert Strang	Cengage Learning	2016 (5th Ed.)
3	**Calculus — Early Transcendentals**	James Stewart, Daniel K. Clegg & Saleem Watson	Cengage Learning	2020 (9th Ed.)
4	**Higher Engineering Mathematics**	B. S. Grewal	Khanna Publishers	2017 (44th Ed.)
5	**Numerical Methods for Engineers**	Steven C. Chapra & Raymond P. Canale	McGraw-Hill	2020 (8th Ed.)
6	**Differential Equations and Linear Algebra**	Gilbert Strang	Wellesley-Cambridge Press	2014
7	**Introduction to Linear Algebra**	Gilbert Strang	Wellesley-Cambridge Press	2023 (6th Ed.)

Mathematics For Electrical Engineering

25MT2022

L-T-P-S	2-2-0-0	Credits	4	Contact Hrs/Wk	4
Semester	S03	Mode	Regular	Category	BSC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply complex-variable theory — analytic functions, Cauchy-Riemann equations, contour integration at intuition, residue theorem applications — to AC circuit analysis and signal processing.	BTL3 · Apply
CO2	Apply Laplace transforms — definitions, properties, inverse via partial fractions, applications to ODEs with initial conditions, transfer functions — to circuit and control analysis.	BTL3 · Apply
CO3	Apply Z-transforms and discrete-time analysis — Z-transform pairs, system function, stability via pole locations — to digital control and DSP-on-power-systems applications.	BTL3 · Apply
CO4	Apply vector calculus and electromagnetics math — gradient, divergence, curl, Maxwell's equations in integral and differential form — to electric and magnetic field analysis.	BTL3 · Apply
CO5	Apply Fourier methods — Fourier series, Fourier transform, FFT for spectrum analysis — to power-quality and harmonic analysis.	BTL3 · Apply
CO6	Apply numerical methods and modern computational tools — MATLAB / Octave / Python (NumPy + SciPy + Control Systems library), AI-assisted symbolic computation — to EEE engineering problems.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	H	–	–	H
CO2	H	H	H	–	H
CO3	H	H	H	–	H
CO4	H	H	–	–	H
CO5	H	H	H	–	H
CO6	H	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Complex Analysis for Electrical Engineering** Complex numbers — algebraic and polar forms; the canonical phasor representation in AC.; Functions of complex variable — analyticity; the canonical Cauchy-Riemann equations.; Elementary complex functions — exp, sin, cos, log; the canonical Euler formula bridge.; Contour integration at intuition — line integrals in the complex plane; the canonical Cauchy integral theorem.; Residue theorem at intuition — for evaluating real integrals via complex analysis; the canonical applications.; Conformal mapping at intuition — the canonical Smith chart use in RF/transmission-line design.; Applications to AC circuit analysis — phasor algebra recap; impedance as complex number.; A first lab: in MATLAB or Python, compute admittance and impedance of complex-RLC circuits using phasor analysis; verify against simulation.	CO1
M2	**Laplace Transforms** Laplace transform definition — F(s) = ∫f(t)e^(-st)dt; the canonical "frequency-domain" view.; Common transform pairs — δ(t), u(t), e^(-at), sin/cos at; the canonical reference table.; Properties — linearity, time-shift, frequency-shift, differentiation, integration; the canonical algebraic shortcuts.; Inverse Laplace via partial fractions — the canonical decomposition method.; Solving ODEs with Laplace — the canonical "transform → algebra → inverse-transform" workflow.; Transfer functions — the canonical input-output view of LTI systems; depth in control courses downstream.; Applications — RLC transient analysis, control-system step response, the canonical engineering problems.; A lab: solve the step response of a 2nd-order RLC circuit via Laplace transform; verify against numerical ODE solution; observe damping behavior for over/under/critically-damped cases.	CO2

	Title & Topics	CO
M3	**Z-Transform and Discrete-Time Analysis** Discrete-time signals and systems — recap from MFCS at intuition.; Z-transform definition — X(z) = Σx[n]z^(-n); the canonical generalization for sampled signals.; Common Z-transform pairs — δ[n], u[n], a^n; the canonical reference table.; Region of convergence — definition; the canonical relationship to causality and stability.; Inverse Z-transform via partial fractions — the canonical computation.; Difference equations — discrete analog of ODEs; the canonical solution via Z-transform.; System stability — pole locations inside/outside unit circle; the canonical "all poles inside unit circle = stable" criterion.; Applications — digital filters for power-system measurement; digital controllers for power electronics.; A lab: in Python (scipy.signal) or MATLAB, design a digital low-pass filter via Z-transform; verify frequency response; apply to a real power-system measurement signal.	CO3
M4	**Vector Calculus and Electromagnetic Math** Vector calculus recap (from LACE) — gradient, divergence, curl; the canonical operators.; Important vector identities — div curl = 0, curl grad = 0; the canonical engineering shortcuts.; Line, surface, and volume integrals — recap with EM applications.; Stokes' theorem and Divergence theorem — the canonical conversions between integral types.; Maxwell's equations in integral form — Gauss's law (electric, magnetic), Faraday's law, Ampere's law (with Maxwell correction); the canonical foundational EE physics.; Maxwell's equations in differential form — the canonical compact representation.; Wave equation derivation — from Maxwell's equations; the canonical $c^2 \nabla^2 E = \partial^2 E/\partial t^2$ result.; A lab: solve a canonical EM problem (e.g., field around an infinite line charge, around an infinite straight current) using both integral and differential forms of Maxwell's equations; verify equivalence.	CO4
M5	**Fourier Methods for Power-Quality Analysis** Fourier series recap (from MFCS) — periodic-signal decomposition into sinusoids.; Fourier series for non-sinusoidal periodic waveforms — square waves, triangle waves, rectified sine; the canonical examples in power electronics.; Total Harmonic Distortion (THD) — the canonical power-quality metric; the canonical formula.; Fourier transform for aperiodic signals — for transient analysis.; FFT for spectrum analysis — recap from MFCS; applications to power-system harmonic measurement.; Power-quality issues — voltage sags, swells, transients, harmonics; the canonical IEEE 519 standard.; Modern power-quality monitoring — smart meters, IEDs (Intelligent Electronic Devices); the canonical IoT-on-power-grid pattern.; A lab: in Python or MATLAB, compute FFT of a typical SMPS-injected current waveform; identify dominant harmonics; calculate THD; reflect on regulatory implications.	CO5
M6	**Numerical Methods and Computational Tools** Numerical solution of linear systems — recap from LACE; iterative methods for power-flow analysis.; Newton-Raphson method — for nonlinear power-flow equations; the canonical iterative solver.; Numerical integration — Trapezoidal, Simpson's, Runge-Kutta; for transient simulation.; MATLAB / Octave — the canonical EE simulation platform; Simulink for system modeling.; Python alternative — NumPy + SciPy + python-control; the open-source modern alternative; canonical for the new generation.; Modern EE-specific simulation tools — PSCAD/EMTDC for transients, ETAP for power-system analysis, PSpice / LTspice for circuits, the canonical industrial tooling.; AI-augmented mathematical computation — Claude / GPT-4 + sympy for symbolic computation, Wolfram Alpha for verification; the discipline of always cross-checking with hand-derivation or simulation.; A final-week lab: solve a real EEE problem using all the math from M1-M5 (e.g., transient analysis of a 3-phase rectifier feeding an inductive load); use MATLAB or Python; produce a complete report.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Advanced Engineering Mathematics**	Erwin Kreyszig	Wiley	2022 (10th Ed.)
2	**Complex Variables and Applications**	James Ward Brown & Ruel V. Churchill	McGraw-Hill	2013 (9th Ed.)
3	**Signals and Systems**	Alan V. Oppenheim, Alan S. Willsky & S. Hamid Nawab	Pearson	1996 (2nd Ed.)
4	**Engineering Electromagnetics**	William H. Hayt Jr. & John A. Buck	McGraw-Hill	2018 (9th Ed.)
5	**Mathematical Methods for Physicists**	George B. Arfken, Hans J. Weber & Frank E. Harris	Academic Press	2012 (7th Ed.)
6	**Higher Engineering Mathematics**	B. S. Grewal	Khanna Publishers	2017 (44th Ed.)
7	**Numerical Methods for Engineers**	Steven C. Chapra & Raymond P. Canale	McGraw-Hill	2020 (8th Ed.)

Modern Power Systems & Smart Grid Technologies

25EE3106

L-T-P-S	2-0-0-4	Credits	3	Contact Hrs/Wk	6
Semester	S05	Mode	Regular	Category	PCC
Pre-requisite	EGTD				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze power-system representation — the per-unit system, single-line diagrams and the bus admittance/impedance matrices — to model an interconnected network.	BTL4 · Analyze
CO2	Apply load-flow methods — Gauss-Seidel, Newton-Raphson and fast-decoupled — to compute the operating state of a transmission and distribution network.	BTL3 · Apply
CO3	Analyze symmetrical and unsymmetrical faults and rotor-angle stability (the swing equation and the equal-area criterion) for secure grid operation.	BTL4 · Analyze
CO4	Apply smart-grid sensing and communication — PMUs, WAMS, SCADA/EMS and grid communication standards — to build observable, controllable grids.	BTL3 · Apply
CO5	Analyze the integration of distributed energy resources, microgrids and energy storage into a modern distribution system.	BTL4 · Analyze
CO6	Apply grid operation and analytics — economic dispatch, demand-side management and AI-assisted forecasting — to a smart-grid case study.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO7	PSO1
CO1	H	H	–	–	–	–	H
CO2	H	H	–	–	M	–	H
CO3	H	H	–	M	–	–	H
CO4	H	–	M	–	H	–	H
CO5	H	H	–	–	–	H	H
CO6	H	–	H	–	H	M	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Power-System Representation & Per-Unit System** Recap of the EGTD end-to-end system; what “modern” adds; The per-unit system — base quantities, changing base and its advantages; Single-line and impedance diagrams — symbols for generator, transformer, bus and line; The bus admittance matrix (Y-bus) — formation by inspection and by singular transformation; The bus impedance matrix (Z-bus) and network reduction; Modelling generators, transformers, lines and loads for system studies; Symmetrical components — introduction (full fault use in Module 3); Setting up a small test-network model	CO1
M2	**Load Flow / Power Flow Analysis** The power-flow problem — bus types (slack, PV, PQ); Power-flow equations and their nonlinearity; Gauss-Seidel method — acceleration and convergence; Newton-Raphson method and the Jacobian; Fast-decoupled load flow (B′ and B″ matrices); Interpreting results — voltage profile, line flows, losses and reactive margins; Contingency analysis (N-1); ill-conditioned systems and flat start; Load-flow software (PowerWorld, MATPOWER) on the test network	CO2
M3	**Fault Analysis & System Stability** Symmetrical three-phase fault — pre-fault conditions, Thevenin equivalent and the Z-bus method; Symmetrical components — positive, negative and zero sequence; the operator a; Sequence networks for generators, transformers and lines; Unsymmetrical faults — single-line-to-ground, line-to-line and double-line-to-ground; Fault levels, short-circuit MVA and circuit-breaker rating (ANSI/IEEE C37); The swing equation and the inertia constant H; Steady-state and transient stability — the equal-area criterion and critical clearing time; Methods to improve stability — fast fault clearing, excitation control and governor action	CO3

	Title & Topics	CO
M4	**Smart-Grid Sensing & Communication** The smart-grid vision — a two-way, digital, observable grid; Phasor Measurement Units (PMUs) and synchrophasors; Wide-Area Monitoring Systems (WAMS); SCADA and Energy Management Systems (EMS); Grid communication standards — IEC 61850 and DNP3; Cyber-physical security of the grid (overview); State estimation and bad-data detection; Designing a monitoring scheme for the test feeder	CO4
M5	**Distributed Energy Resources, Microgrids & Storage** Distributed generation — rooftop solar and small wind and their grid impact; Energy storage in the grid — batteries and the duck curve; Microgrids — grid-connected and islanded operation; Distribution challenges with high DER penetration — voltage rise and reverse flow; Hosting capacity and interconnection standards (IEEE 1547); Electric vehicles as grid loads and resources (V2G — depth in EVBT); India's 500-GW-by-2030 renewable context; Analysing DER impact on the test distribution system	CO5
M6	**Grid Operation, Analytics & Demand-Side Management** Economic dispatch and unit-commitment basics; Frequency control — governor action, load-frequency control and AGC; Voltage and reactive-power control; FACTS at concept level; Smart metering and Advanced Metering Infrastructure (AMI); Demand response and demand-side management; time-of-use tariffs and prosumers; Load and renewable forecasting (AI-assisted; ML tools used as given); Outage management, self-healing grids and grid digital twins; A smart-grid analytics case study for the demo	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Power System Analysis and Design**	J. Duncan Glover, Thomas Overbye & Mulukutla Sarma	Cengage	2017 (6th Ed.)
2	**Modern Power System Analysis**	I. J. Nagrath & D. P. Kothari	McGraw-Hill	2011 (4th Ed.)
3	**Power System Analysis**	Hadi Saadat	McGraw-Hill	2010 (3rd Ed.)
4	**Power System Analysis**	John J. Grainger & William D. Stevenson	McGraw-Hill	1994
5	**Power System Stability and Control**	Prabha Kundur	McGraw-Hill	1994
6	**Smart Grid: Technology and Applications**	Janaka Ekanayake, Kithsiri Liyanage, Jianzhong Wu, Akihiko Yokoyama & Nick Jenkins	Wiley	2012

Power Electronics & Wide-Bandgap Converters

25EE2204

L-T-P-S	2-0-0-4	Credits	3	Contact Hrs/Wk	6
Semester	S04	Mode	Regular	Category	PCC
Pre-requisite	ECES				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply power-semiconductor-device fundamentals — power diode, BJT, MOSFET, IGBT, thyristor (SCR), GTO and IGCT — their V-I characteristics, switching dynamics, gate-drive requirements and canonical applications to select an appropriate device for a given converter.	BTL3 · Apply
CO2	Apply AC-DC converter analysis — single-phase and three-phase half-wave / full-wave / bridge rectifiers, controlled rectifiers, 12-pulse converters — to design rectifier-based DC supplies including power-factor and harmonic-mitigation requirements per IEEE 519 / IS 16636.	BTL3 · Apply
CO3	Apply DC-DC converter design — buck, boost, buck-boost, Cuk, SEPIC, and isolated topologies (flyback, forward, push-pull, half-bridge, full-bridge) — in CCM and DCM regimes for canonical SMPS, solar-MPPT, EV-DC-DC and telecom-power applications.	BTL3 · Apply
CO4	Apply DC-AC inverter design — single-phase and three-phase voltage-source inverters, square-wave and PWM modulation strategies (sine-PWM, space-vector PWM), multilevel inverters at intuition — to canonical EV-traction-inverter and grid-tied solar-inverter applications.	BTL3 · Apply
CO5	Analyse AC-AC converters (cycloconverters, matrix converters), snubber-circuit design, gate-driver requirements, EMI / EMC considerations per CISPR 22 / FCC Part 15, and PCB-layout discipline for high-frequency power-electronics designs.	BTL4 · Analyze
CO6	Analyse wide-bandgap devices (SiC, GaN), modern simulation tools (PSIM, LTspice, MATLAB Simulink), the AI-augmented power-electronics-design workflow and the Indian power-electronics ecosystem in the context of EV traction, solar-string inverter, and emerging high-frequency power-supply applications.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO7	PSO1
CO1	H	H	H	M	–	H
CO2	H	H	H	M	–	H
CO3	H	H	H	H	–	H
CO4	H	H	H	H	–	H
CO5	H	H	H	H	–	H
CO6	H	H	H	H	M	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Power Semiconductor Devices** Why power semiconductors — efficient energy conversion via switching; the canonical "switch on and off rapidly with low loss" principle; comparison with linear regulators.; Power diode — V-I characteristics, reverse recovery time t_rr, the canonical PIN-structure; soft-recovery and Schottky-diode variants.; Power BJT — saturated-switch operation, the canonical safe-operating-area (SOA), second breakdown; mostly legacy in 2026 (replaced by MOSFET / IGBT).; Power MOSFET — the canonical switching-power-supply workhorse below 200 V; R_DS(on), gate charge Q_g, body diode, Miller plateau; canonical Infineon CoolMOS, ON-Semi, ST Micro families.; IGBT (Insulated Gate Bipolar Transistor) — the canonical 600 V – 6500 V industrial / EV-traction switch; gate-drive simplicity meets bipolar conduction loss; canonical Mitsubishi, Infineon, ABB modules.; Thyristor (SCR) — the canonical line-commutated high-power switch; latched-on operation requiring natural or forced commutation; canonical HVDC / industrial-rectifier applications.; GTO and IGCT at intuition — high-power gate-turn-off devices; the canonical Indian-Railways traction and HVDC-valve applications (LCC-HVDC stations).; First lab — characterise a small MOSFET; measure R_DS(on), threshold voltage V_GS(th), switching waveforms with an oscilloscope; observe the canonical Miller plateau in the gate waveform.	CO1

	Title & Topics	CO
M2	**AC-DC Converters (Rectifiers)** Diode-based half-wave rectifier — the canonical simplest topology; the canonical 0.318 V_p average output; output ripple analysis.; Diode-based full-wave rectifier — centre-tapped and bridge configurations; the canonical 0.636 V_p average output.; Filtering — capacitive (most common), LC and π-filter; the canonical ripple-voltage equation V_r ≈ I_dc / (f C); design trade-offs.; Three-phase rectifiers — half-wave (3-pulse), full-wave (6-pulse bridge); the canonical voltage and current waveforms and ripple analysis.; 12-pulse rectifiers — the canonical high-power industrial / HVDC / large-motor-drive application; harmonic-cancellation benefit.; Controlled rectifiers (thyristor-based) — phase-control; the canonical α-control of average DC output (V_dc = V_dc0 cosα); single-phase and three-phase variants.; Power-factor and harmonics in rectifiers — the canonical IEEE 519 / IS 16636 limits; modern power-factor-correction (PFC) techniques (boost-PFC, bridgeless-PFC).; Lab — build and characterise a 1-phase bridge rectifier with capacitive filter; measure ripple, regulation, power factor; verify against textbook predictions.	CO2
M3	**DC-DC Converters** Buck converter — canonical step-down topology; the canonical V_o = D · V_in relationship in CCM; inductor and capacitor selection.; Boost converter — canonical step-up topology; the canonical V_o = V_in / (1 − D) relationship; the canonical solar-MPPT / EV-DC-DC-stage application.; Buck-boost converter — bidirectional voltage transformation; the canonical inverting non-isolated topology; V_o = − V_in · D / (1 − D).; Cuk and SEPIC converters at intuition — the canonical advanced non-isolated topologies with continuous input current.; Continuous Conduction Mode (CCM) vs Discontinuous Conduction Mode (DCM) — the canonical regimes; design criteria; transition load current.; Isolated DC-DC converters — flyback (low power), forward, push-pull, half-bridge, full-bridge (high power); the canonical "transformer-isolated SMPS" workhorses.; Modern SMPS controller ICs — TI UCC2802x family, ADI LT8000-series, Infineon ICE family; the canonical "controller-plus-MOSFET" design pattern.; Lab — design and build a small 12 V → 5 V buck converter at 1 A; measure efficiency at full load; observe switching waveforms; verify against design calculations.	CO3
M4	**DC-AC Inverters** Single-phase voltage-source inverter — half-bridge and full-bridge; the canonical four switching states and dead-time requirements.; Square-wave vs PWM modulation — the canonical motivation for PWM (lower harmonics, cleaner output); modulation index.; Sine-PWM (SPWM) — the canonical compare-with-sine-reference technique; carrier frequency, modulation index, over-modulation.; Three-phase voltage-source inverter — the canonical 6-switch topology; eight switching states (six active + two zero); 120° and 180° conduction modes.; Space-Vector PWM (SVPWM) at intuition — the canonical higher-modulation-index technique; the canonical use in modern motor drives and grid-tied inverters.; Multilevel inverters at intuition — neutral-point-clamped (NPC), flying-capacitor, modular multilevel (MMC); the canonical HVDC-VSC, traction and large-PV-inverter applications.; EV traction inverter — the canonical three-phase IGBT or SiC-MOSFET inverter with FOC control; canonical Tesla / Ola Electric / Tata EV / Mahindra Last Mile Mobility designs.; Lab — simulate a three-phase SPWM inverter feeding an RL load in MATLAB Simulink; observe line voltages, line currents, and FFT showing dominant harmonics; vary modulation index.	CO4
M5	**AC-AC Converters & Protection** AC voltage controllers — phase-controlled and integral-cycle control; the canonical light-dimmer / fan-speed-control / industrial-heater applications.; Cycloconverters — the canonical high-power low-frequency-output applications (cement-kiln drives, large rolling-mill drives); mostly legacy due to VFD prevalence.; Matrix converters at intuition — direct AC-AC conversion without DC-link; canonical research and specialty applications (aircraft, aerospace, sustainability research).; Snubber circuits — RC, RCD; the canonical switching-spike suppression and EMI control.; Gate drivers — the canonical isolated gate-drive ICs (TI UCC family, Infineon EiceDriver, Silicon Labs Si827x); Miller-clamp, desat-detection, fault-handling features.; EMI / EMC in power electronics — conducted and radiated emissions; the canonical CISPR 22 / FCC Part 15 / IS 13252 standards; mitigation via input / output LC filtering and PCB layout.; PCB layout for power electronics — the canonical loop-area minimisation, ground-plane discipline, thermal-pad practice; canonical industry-standard layout patterns.; Lab — characterise the EMI emissions of the M3 buck converter using a near-field probe and spectrum analyser; identify dominant frequencies; apply LC filtering; measure the improvement.	CO5

	Title & Topics	CO
M6	**Wide-Bandgap Devices, Modern Tooling & Indian Ecosystem** Silicon carbide (SiC) — the canonical wide-bandgap material for high-voltage (650 – 1700 V), high-frequency (50 kHz – 200 kHz) applications; canonical EV-traction-inverter and solar-string-inverter applications.; Gallium nitride (GaN) — the canonical wide-bandgap material for low-voltage (40 – 650 V), very-high-frequency (1 – 10 MHz) applications; canonical mobile-phone-charger (GaNFast) and compact-power-supply applications.; SiC vs Si trade-offs — switching-loss reduction, conduction-loss trade-off, thermal-management implications; the canonical efficiency-vs-cost engineering decision.; GaN vs Si trade-offs — size and weight reduction at high frequency; gate-drive challenges (GaN HEMT lower V_GS_max); canonical applications.; Modern simulation tools — LTspice (free, ubiquitous), PSIM (specialised for power electronics + control), MATLAB Simulink with Simscape Electrical, PLECS (precision-converter modelling); the canonical "verify before building" workflow.; AI-augmented power-electronics design — LLM-assisted component selection (Claude / GPT-4 for sanity-checking power-stage choices), control-loop auto-tuning starting points; the discipline of always validating with simulator and bench measurement.; India's power-electronics ecosystem — Tata Power Solar inverters (Bengaluru), Delta Electronics India (Gurgaon, Chennai), ABB India, Schneider Electric India, Bharat Electronics Limited (BEL) defence-electronics; the upcoming Tata-PSMC 28 nm fab implications for power-semiconductor manufacturing.; Final-week lab — redesign the M3 buck converter using a GaN-based controller (e.g., TI LMG3422) at 1 MHz switching frequency; compare efficiency and size against the silicon-based version; observe the canonical wide-bandgap advantages.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Power Electronics — Converters, Applications, and Design**	Ned Mohan, Tore M. Undeland & William P. Robbins	Wiley	2003 (3rd Ed.)
2	**Power Electronics — Circuits, Devices, and Applications**	Muhammad H. Rashid	Pearson Education	2017 (4th Ed.)
3	**Fundamentals of Power Electronics**	Robert W. Erickson & Dragan Maksimović	Springer	2020 (3rd Ed.)
4	**Power Electronics**	P. S. Bimbhra	Khanna Publishers	2017 (6th Ed.)
5	**Power Electronics — A First Course**	Ned Mohan	Wiley	2012
6	**Wide Bandgap Semiconductor Power Devices — Materials, Physics, Design, and Applications**	B. Jayant Baliga	Woodhead Publishing	2018
7	**GaN Transistors for Efficient Power Conversion**	Alex Lidow, Michael de Rooij, Johan Strydom, David Reusch & John Glaser	Wiley	2019 (3rd Ed.)

Power System Protection And Control

25EE3208

L-T-P-S	2-0-0-4	Credits	3	Contact Hrs/Wk	6
Semester	S06	Mode	Regular	Category	PCC
Pre-requisite	EGTD				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand power-system protection philosophy, relay types and protection-zone concepts.	BTL2 · Understand
CO2	Apply overcurrent protection — relay characteristics, coordination and grading — for radial feeders.	BTL3 · Apply
CO3	Apply differential protection for transformers, generators and busbars.	BTL3 · Apply
CO4	Analyze distance (impedance) protection for transmission lines — zones, characteristics, pilot schemes and power swings.	BTL4 · Analyze
CO5	Apply economic operation of power systems — economic dispatch and hydrothermal scheduling.	BTL3 · Apply
CO6	Analyze load-frequency control and voltage stability of a power system — the single-area LFC loop, automatic voltage regulation, and P-V/Q-V behaviour with reactive-power support.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	H	H	M	M	M	L	L	–	L	L	–	H	M
CO2	H	H	H	H	H	L	–	–	M	M	L	H	M
CO3	H	H	H	H	H	L	–	–	M	M	L	H	M
CO4	H	H	H	H	H	L	–	–	L	L	–	H	M
CO5	H	H	H	H	H	L	–	–	L	L	–	H	M
CO6	H	H	H	H	H	L	–	L	M	M	M	H	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Protection Philosophy and Relay Fundamentals** Protection zones — overlapping zones; primary and backup protection; speed vs selectivity; Instrument transformers — CT accuracy class, burden and saturation; PT accuracy class; Relay operating principles — electromagnetic attraction; induction disc; static; digital; numerical; Relay characteristics — definite time; inverse time; extremely inverse; Circuit breakers — SF6 and vacuum; arc interruption; operating time; auto-reclosing; SCADA interface; Protection requirements — reliability (dependability + security), sensitivity, selectivity, speed, simplicity	CO1
M2	**Overcurrent Protection** Overcurrent relay (OCR) — plug-setting multiplier (PSM) and time-multiplier setting (TMS); IDMT relay — t = TMS × 0.14/(PSM^0.02 − 1); standard, very and extremely inverse curves; Grading of overcurrent relays — time-current grading; radial-feeder coordination; grading margin (0.3–0.4 s); Directional overcurrent relay — operating vs polarizing quantity; forward/reverse; Earth-fault relay — residual current I_0 = (I_a + I_b + I_c)/3; high-impedance earthing; restricted earth fault; Ring-main and interconnected systems — directional relays; pilot-wire protection; check-zone coordination	CO2
M3	**Differential Protection** Differential principle — I_d = I_1 − I_2; operate if I_d > k·I_r; Transformer differential protection — CT-ratio matching; zero-sequence elimination; percentage-bias characteristic; Inrush blocking — second-harmonic restraint; dual-slope bias characteristic; Transformer protection zones — differential for internal faults; overcurrent for through faults; Buchholz relay; Generator differential protection — stator-winding short-circuit; split-phase protection; Busbar differential protection — high-impedance (Solkor); low-impedance (Merz-Price); check relay	CO3

	Title & Topics	CO
M4	**Distance Protection** Distance-relay principle — Z_app = V/I; apparent impedance on the R-X diagram; Zone settings — Zone 1 (80–85%, instantaneous); Zone 2 (120%, time-delayed); Zone 3 (backup); Relay characteristics — impedance, reactance, mho, quadrilateral; load encroachment; Power-swing blocking — distinguishing a fault from a stable/unstable power swing; Phase and ground distance — separate pickup and settings; the mutual-impedance effect; Pilot protection schemes — POTT, PUTT, blocking schemes and DUTT	CO4
M5	**Economic Operation and Control** Economic dispatch — minimise total generation cost subject to load balance; Lagrange multiplier λ; Incremental cost (λ) — λ = dC_i/dP_i; the equal-incremental-cost criterion; Generator cost curve — input-output, heat-rate and incremental-heat-rate curves; fuel cost; Loss formula (B-coefficients) — P_L = ΣΣ P_i B_ij P_j; penalty factor; optimal dispatch with losses; Unit commitment — priority-list method; dynamic programming; MIP formulation; Hydrothermal scheduling — water value; the coordination equation; weekly hydro-energy target	CO5
M6	**LFC and Voltage Stability** Load-frequency control (single area) — governor, turbine and generation-load model; primary and secondary loops; Automatic voltage regulator (AVR) — block diagram; Voltage stability — maintaining acceptable voltages under steady-state and disturbance conditions; P-V (nose) curve — maximum loadability; the voltage-collapse point; voltage-stability margin; Q-V curve — reactive demand vs voltage; the stability boundary; Q reserve; Reactive-power sources — capacitor banks, synchronous condenser, STATCOM, SVC and generator VAr capability	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Protective Relaying: Principles and Applications**	J. Lewis Blackburn & Thomas J. Domin	CRC Press	2014 (4th Ed.)
2	**Power System Protection**	A. G. Phadke & J. S. Thorp	Wiley-IEEE Press	2009 (2nd Ed.)
3	**Fundamentals of Power System Protection**	Y. G. Paithankar & S. R. Bhide	PHI	2011 (2nd Ed.)
4	**Digital Protection for Power Systems**	A. G. Phadke & J. S. Thorp	IET	2009
5	**Power System Analysis**	Hadi Saadat	McGraw-Hill	2010 (3rd Ed.)

Problem Solving Through C

25SC1101

L-T-P-S	3-0-2-4	Credits	5	Contact Hrs/Wk	9
Semester	S01	Mode	Regular	Category	ESC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply C language fundamentals — variables, primitive types, operators, control flow (if/else, while, for, switch) — to write small programs.	BTL3 · Apply
CO2	Apply functions and modular programming — function definition, parameters, return values, scope, recursion — to decompose problems into reusable pieces.	BTL3 · Apply
CO3	Apply arrays and strings — 1D and 2D arrays, array-of-strings idiom, the canonical string-manipulation patterns — to manage collections of values.	BTL3 · Apply
CO4	Analyze pointers — addresses, indirection, pointer arithmetic, pointer-to-pointer, the relationship to arrays — and use them correctly to write efficient code.	BTL4 · Analyze
CO5	Apply structures and dynamic memory — struct definition, nested structs, malloc/calloc/realloc/free, the discipline of resource management — to build complex data layouts.	BTL3 · Apply
CO6	Apply file I/O, modular compilation, and basic debugging — fopen/fprintf/fscanf/fclose, header files, multi-file projects, gdb fundamentals, AI coding-assistant productivity — to ship working C programs.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	–	–	M	H
CO2	H	–	H	–	H
CO3	H	–	H	–	H
CO4	H	H	H	–	H
CO5	H	–	H	–	H
CO6	H	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**C Fundamentals — Variables, Types, Control Flow** A first C program — Hello World; the compile-link-run cycle on Linux (gcc, ./a.out); how C's text-to-executable pipeline works.; Variables and primitive types — int, char, float, double; the size of each on typical architectures; signed vs unsigned.; Operators — arithmetic, relational, logical, bitwise, assignment, conditional (?:); operator precedence and associativity.; Input/output — printf, scanf, the format specifiers; the canonical bugs (forgetting the & in scanf, mismatched types).; Control flow — if/else, switch/case, while, do-while, for; the canonical idioms; break and continue.; Type conversions — implicit and explicit casts; the canonical surprises (integer division, narrowing conversions).; A first lab: write a program that reads an integer N and prints the first N prime numbers; demonstrate by running for N=1, 10, 100; commit to GitHub.	CO1
M2	**Functions and Modular Programming** Function definition and declaration — the prototype, the definition, the call; the parameter-passing model.; Function parameters — pass-by-value (the default in C); how to "pass by reference" using pointers (preview of M4).; Return values — single return, the canonical idioms; how to return multiple values via pointers or structs.; Scope and lifetime — local variables, global variables, static variables; the storage classes (auto, register, static, extern).; Recursion — the canonical examples (factorial, Fibonacci, towers of Hanoi); the call stack mental model.; Iteration vs recursion — when each fits; the trade-offs; the discipline of choosing intentionally.; A lab: implement gcd via Euclidean algorithm both iteratively and recursively; benchmark on 1M random pairs; observe practical performance.	CO2

	Title & Topics	CO
M3	**Arrays and Strings** Arrays — declaration, initialization, indexing; how arrays are stored in memory.; The array-name-decays-to-pointer rule — the canonical C subtlety; the implications for sizeof.; 2D arrays — row-major storage; the canonical matrix-multiplication pattern; preview of why this matters for cache efficiency.; Strings as char arrays — null termination, the standard string functions (strlen, strcpy, strcmp, strcat) from string.h.; String manipulation idioms — the canonical patterns (in-place reverse, palindrome check, substring search).; Array-of-strings — char *names[] vs char names[][N]; the mental model for both.; A lab: implement a small text-processing utility — read a file, count words/lines/chars, find longest word, identify palindromes; verify against wc and a Python reference.	CO3
M4	**Pointers** Pointers — declaration, the address-of operator (&), the indirection operator (*); the memory-address mental model.; Pointers and arrays — the equivalence (a[i] == *(a+i)); when to use array vs pointer notation.; Pointer arithmetic — adding/subtracting integers from pointers; the canonical idioms (iterate via ptr++).; Pointers to pointers — char **argv; the canonical use cases (commandline args, dynamic 2D arrays).; Function pointers — the syntax, the canonical use case (qsort comparison function); preview of how they enable polymorphism in C.; Common pointer bugs — null pointer dereference, dangling pointers, off-by-one in pointer iteration; debugging discipline.; A lab: implement a generic sort using qsort with custom comparator function; sort an array of structs by multiple fields; test on at least 3 different sort orders.	CO4
M5	**Structures and Dynamic Memory** Structs — declaration, initialization, accessing fields with .; nested structs.; Pointer-to-struct — the → operator; the canonical idiom for linked structures.; Typedef — the cleanup tool; the canonical typedef struct pattern.; Dynamic memory — malloc, calloc, realloc, free; the heap vs stack distinction.; Memory leaks and double-free bugs — the canonical errors; the discipline of "every malloc has a matching free".; Building a linked list from scratch — the canonical introduction to dynamic data structures (full coverage in DAST).; A lab: build a small student-records system — struct Student with id/name/marks, dynamic array of student pointers, malloc/free discipline, free everything at the end; verify no leaks via valgrind.	CO5
M6	**File I/O, Modular Compilation, Debugging** File I/O — fopen, fprintf, fscanf, fclose; the FILE * abstraction; binary vs text mode.; Multi-file C projects — header files (.h), source files (.c), include guards; the compile-link-run flow with multiple files.; Make basics — Makefile fundamentals; the canonical CSE-style Makefile for a small project.; gdb fundamentals — break, run, next, step, print, backtrace; debugging segfaults the disciplined way.; valgrind for memory checking — detecting leaks and invalid accesses; the canonical clean-output target.; AI coding assistants for C — Claude / Copilot / Cursor for boilerplate (struct definitions, getters, Makefile templates); the discipline of treating output as draft, especially for security-critical pointer code.; When AI assistants fail in C — subtle pointer bugs, off-by-one in arrays, missed free()s, format-string vulnerabilities; the eye for these only develops by reading and writing lots of careful C.; A final-week lab: build a small command-line phonebook — multi-file project with Makefile, file-backed persistence, dynamic memory for entries, gdb-tested for crash-free, valgrind-tested for leak-free; submit to GitHub.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**The C Programming Language**	Brian W. Kernighan & Dennis M. Ritchie	Prentice Hall	1988 (2nd Ed.)
2	**Programming in C**	Stephen G. Kochan	Addison-Wesley	2014 (4th Ed.)
3	**C Programming — A Modern Approach**	K. N. King	W. W. Norton	2008 (2nd Ed.)
4	**Expert C Programming — Deep C Secrets**	Peter van der Linden	Prentice Hall	1994
5	**Modern C**	Jens Gustedt	Manning	2024 (2nd Ed.)
6	**21st Century C — C Tips from the New School**	Ben Klemens	O'Reilly	2014 (2nd Ed.)
7	**Effective Debugging — 66 Specific Ways to Debug Software and Systems**	Diomidis Spinellis	Addison-Wesley	2016

Python Programming

25SC2007

L-T-P-S	2-0-2-0	Credits	3	Contact Hrs/Wk	4
Semester	S02	Mode	Regular	Category	ESC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply Python language fundamentals — variables, types, operators, control flow, functions, the PEP-8 style discipline — to write idiomatic Python.	BTL3 · Apply
CO2	Apply Python's built-in data structures — lists, tuples, sets, dicts, comprehensions, generators — to solve problems with the right structure.	BTL3 · Apply
CO3	Apply object-oriented programming in Python — classes, inheritance, special methods (dunder), context managers — to write reusable code.	BTL3 · Apply
CO4	Apply file I/O, exception handling, and modular programming — file reading/writing, JSON, csv, try/except, modules and packages — to build practical Python programs.	BTL3 · Apply
CO5	Apply the scientific Python stack — NumPy for arrays, Pandas for data frames, Matplotlib for visualization — to data-analysis tasks.	BTL3 · Apply
CO6	Apply modern Python tooling — virtual environments, package management (uv / pip), testing with pytest, type hints, AI coding assistants — to ship working Python projects.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO3	PO5	PSO1
CO1	H	–	H	H
CO2	H	H	–	H
CO3	H	H	–	H
CO4	H	H	–	H
CO5	H	H	H	H
CO6	H	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Python Fundamentals** Why Python — readability, dynamism, the scientific ecosystem; the contrast with C from PSTC.; Variables, dynamic typing, the canonical primitive types (int, float, str, bool, None).; Operators and expressions — arithmetic, comparison, logical, bitwise; the difference from C operator semantics.; Control flow — if/elif/else, while, for-in (over iterables, not by index); break and continue.; Functions — def, parameters with defaults, *args and **kwargs, lambda; the first-class-function principle.; PEP 8 style discipline — naming conventions, indentation, line length; the canonical formatter (Black) and linter (Ruff).; A first lab: solve 5 LeetCode easy/medium problems in Python; document the Pythonic vs C-translation versions; commit to GitHub.	CO1
M2	**Python Data Structures** Lists — mutable ordered sequences; the canonical operations (append, pop, slicing, sorting).; Tuples — immutable ordered sequences; when to prefer them over lists.; Sets — mutable unordered collections; set operations (union, intersection, difference); when to use them for membership-testing speed.; Dicts — mutable key-value mappings; the canonical operations; the order-preservation guarantee since Python 3.7.; Comprehensions — list, set, dict comprehensions; the readable alternative to map+filter+lambda.; Generators and iterators — yield, generator expressions; the lazy-evaluation pattern; memory-efficiency wins.; A lab: implement a small text-analytics tool using built-in data structures only — word count, top-10 most common words, longest-word-finder, palindrome-detector across a 10K-line text file.	CO2

	Title & Topics	CO
M3	**Object-Oriented Programming in Python** Classes and objects — class definition, instantiation, instance attributes, instance methods.; The __init__ method and other dunder methods — __str__, __repr__, __eq__, __hash__, __len__, __getitem__.; Inheritance — single and multiple; the MRO (Method Resolution Order) at intuition.; Encapsulation — name-mangling with double underscore; the conventional single-underscore for "private".; Properties via @property — getters/setters Pythonic style.; Context managers — the with statement; defining your own via __enter__/__exit__ or contextlib.; A lab: design a small library-management OOP system — Book, Member, Library classes with proper inheritance; implement search/borrow/return with proper exception handling.	CO3
M4	**File I/O, Exceptions, Modules** Built-in file I/O — open() with context manager, reading lines, writing strings.; JSON via the standard json module — loads/dumps for serialization.; CSV via the standard csv module — DictReader/DictWriter for column-aware processing.; Exception handling — try/except/finally/else, raising exceptions, custom exception classes.; The exception hierarchy — Exception → ValueError, KeyError, IOError, etc.; the canonical pattern for catching specific exceptions.; Modules and packages — import, from-import, the __init__.py convention, organizing larger projects.; A lab: build a CSV-to-JSON converter that handles malformed rows gracefully (logs and skips them); add unit tests; submit as a 3-file Python package.	CO4
M5	**Scientific Python Stack** NumPy fundamentals — arrays vs lists, vectorized operations, broadcasting; the performance gain over Python loops.; NumPy operations — element-wise arithmetic, reductions (sum, mean, max), boolean indexing, fancy indexing.; Pandas DataFrame — the canonical data-analysis structure; reading CSV/Excel, basic operations (select, filter, group-by, merge).; Pandas data cleaning — handling missing values (dropna, fillna), type conversions, outlier detection.; Matplotlib basics — pyplot vs object-oriented API; the canonical plot types (line, bar, scatter, histogram).; Seaborn for statistical visualization — the higher-level, pretty-by-default alternative.; A lab: load a real Indian dataset (e.g., RBI policy rates or NSE historical prices) into Pandas; clean missing values; produce 4 visualizations exploring the data; export the cleaned data as JSON.	CO5
M6	**Modern Python Tooling** Virtual environments — why isolation matters; venv vs uv (the modern faster alternative); the canonical project setup.; Package management — pip vs uv; pyproject.toml as the modern manifest; lock files.; Testing with pytest — fixtures, parametrize, assertions; the discipline of "tests live alongside code".; Type hints — the typing module, mypy/Pyright as the static checker; the gradual-typing path Python is on.; AI coding assistants for Python — Claude / Copilot / Cursor; productivity gains for boilerplate; the discipline of treating output as draft.; When AI assistants fail in Python — subtle off-by-one, missing edge cases, misuse of mutable default arguments, performance pitfalls of comprehensions vs generators on large data.; Modern Python deployment at intuition — Docker for reproducible runtimes; pyinstaller for native binaries; serverless options (Lambda, Cloud Run).; A final-week lab: ship a small CLI tool — read CSV, transform, write — packaged with pyproject.toml, tested with pytest (>80% coverage), type-checked with mypy, deployed via uv tool install.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Fluent Python — Clear, Concise, and Effective Programming**	Luciano Ramalho	O'Reilly	2022 (2nd Ed.)
2	**Python Crash Course — A Hands-On, Project-Based Introduction to Programming**	Eric Matthes	No Starch Press	2023 (3rd Ed.)
3	**Effective Python — 90 Specific Ways to Write Better Python**	Brett Slatkin	Addison-Wesley	2019 (2nd Ed.)
4	**Automate the Boring Stuff with Python**	Al Sweigart	No Starch Press	2019 (2nd Ed.)
5	**Python for Data Analysis — Data Wrangling with pandas, NumPy, and Jupyter**	Wes McKinney	O'Reilly	2022 (3rd Ed.)
6	**Robust Python — Write Clean and Maintainable Code**	Patrick Viafore	O'Reilly	2021
7	**Architecture Patterns with Python — Enabling Test-Driven Development, Domain-Driven Design, and Event-Driven Microservices**	Harry Percival & Bob Gregory	O'Reilly	2020

Renewable Energy Systems & Grid Integration

25EE4110

L-T-P-S	3-0-2-0	Credits	4	Contact Hrs/Wk	5
Semester	S07	Mode	Regular	Category	PCC
Pre-requisite	EGTD				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze the renewable-energy landscape and resource assessment (solar irradiance, wind speed) to evaluate the potential of a site.	BTL4 · Analyze
CO2	Analyze solar photovoltaic systems — the cell/module/array, the I–V characteristic, MPPT and system design — for grid-connected and standalone applications.	BTL4 · Analyze
CO3	Analyze wind energy conversion systems — turbine aerodynamics, generators and control — and their power-curve behaviour.	BTL4 · Analyze
CO4	Apply power-electronic interfaces for renewables (PV inverters, MPPT converters, grid-tie inverters) using PEWC converter knowledge.	BTL3 · Apply
CO5	Analyze energy storage and emerging clean-energy carriers — batteries, pumped hydro, and green hydrogen — for balancing variable renewables.	BTL4 · Analyze
CO6	Analyze grid-integration of renewables — interconnection standards, grid codes, grid-forming inverters and the stability challenges of inverter-based resources.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO6	PO7	PSO1
CO1	H	H	–	–	H	H
CO2	H	H	–	–	–	H
CO3	H	H	–	–	–	H
CO4	H	–	H	–	–	H
CO5	H	H	–	–	H	H
CO6	H	–	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Renewable Landscape and Resource Assessment** The energy transition and India's 500-GW-by-2030 target.; Renewable sources overview: solar, wind, hydro, biomass, geothermal.; Solar resource: irradiance, sun path, the solar constant, measurement.; Wind resource: wind speed distribution (Weibull), wind atlas, measurement.; Capacity factor and the variability/intermittency challenge.; Levelised Cost of Energy (LCOE).; Site selection and feasibility.; Assessing a site for the project.	CO1
M2	**Solar Photovoltaic Systems** The PV cell: physics overview, the I–V and P–V characteristic.; Effect of irradiance and temperature.; Modules, strings and arrays; mismatch and shading.; Maximum Power Point Tracking (MPPT) algorithms.; PV system types: grid-tied, standalone, hybrid.; System sizing and component selection.; Solar-plant layout and balance-of-system.; Designing a grid-connected PV system for the site.	CO2
M3	**Wind Energy Conversion Systems** Wind turbine types: HAWT vs VAWT.; Aerodynamics: the power in the wind, the Betz limit, the power coefficient.; The turbine power curve: cut-in, rated, cut-out.; Generators for wind: DFIG and full-converter (PMSG).; Pitch and yaw control; MPPT for wind.; Onshore vs offshore.; Wind-farm layout and wake effects.; Analysing a wind system for the site.	CO3
M4	**Power-Electronic Interfaces for Renewables** Why renewables need power electronics (variable DC/AC → grid-quality AC).; PV inverters: string, central, microinverters.; DC-DC converters and MPPT implementation (topologies from PEWC).; Grid-tie inverters: synchronisation, current control.; Power quality: harmonics, anti-islanding.; Inverter standards and certification.; Reactive-power support from inverters.; Specifying the inverter/converter chain for the project.	CO4

	Title & Topics	CO
M5	**Energy Storage and Green Hydrogen** Why storage: balancing variable renewables, the duck curve.; Battery storage for the grid (chemistry depth shared with EVBT).; Pumped-hydro and other mechanical storage.; Green hydrogen: electrolysis, India's Green Hydrogen Mission.; Fuel cells (concept).; Sizing storage for a renewable plant.; Hybrid renewable + storage systems.; Adding storage to the project plant.	CO5
M6	**Grid Integration of Renewables** Interconnection standards (IEEE 1547) and Indian grid codes.; Impact of high renewable penetration on the grid (links to MPSG).; Grid-following vs grid-forming inverters.; Stability of inverter-based resources (low inertia).; Forecasting renewable generation.; Curtailment and the need for flexibility.; Microgrids and rural electrification.; A grid-integration study for the project plant for the demo.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Renewable Energy: Power for a Sustainable Future**	Godfrey Boyle (ed.)	Oxford University Press	2012 (3rd Ed.)
2	**Solar Photovoltaics: Fundamentals, Technologies and Applications**	Chetan Singh Solanki	PHI	2015 (3rd Ed.)
3	**Wind Energy Explained: Theory, Design and Application**	J. F. Manwell, J. G. McGowan & A. L. Rogers	Wiley	2009 (2nd Ed.)
4	**Renewable and Efficient Electric Power Systems**	Gilbert M. Masters	Wiley	2013 (2nd Ed.)
5	**Grid Integration of Solar Photovoltaic Systems**	Mahesh Kumar & Anurag K. Srivastava	CRC Press	2019

Regular (Semester) Track — Year 1

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
S01	25UC1203	DTFI	Design Thinking For Innovation	HSS	0-0-4-0	2	4	-
	25UC1102	LSFE	Language Skills For Engineers	HSS	0-0-4-0	2	4	-
	25MT1002	DIMS	Discrete Mathematics	BSC	2-2-0-0	4	4	-
	25SC1101	PSTC	Problem Solving Through C	ESC	3-0-2-4	5	9	-
	25EC1102	DISD	Digital System Design	ESC	3-0-2-0	4	5	-
	25EC1101	BEEC	Basic Electrical And Electronic Circuits	ESC	3-0-2-0	4	5	-
	25UC0017	IKGS	Indian Ancient Knowledge Systems - Vedic Mathematics	AUC	2-0-0-0	0	2	-
S02	25SC1203	DAST	Data Structures	ESC	3-0-2-4	5	9	PSTC
	25SC2007	PYPG	Python Programming	ESC	2-0-2-0	3	4	-
	25UC1204	CSFE	Communication Skills For Engineers	HSS	0-0-4-0	2	4	LSFE
	25EC1204	COAA	Computer Organization And Architecture	PCC	3-0-0-0	3	3	-
	25MT1001	LACE	Linear Algebra And Calculus For Engineers	BSC	2-2-0-0	4	4	-
	25EC1203	FOIS	Fundamentals Of Iot And Sensors	ESC	2-0-2-0	3	4	-
	25ME1103	DTWP	Design Tools Workshop	ESC	0-0-4-0	2	4	-
Summer 1	25IE2040	SOIP	Social Internship	PRI	0-0-0-4	0	4	-
	25UC0032	ISDG	Insights To Sustainable Development Goals	AUC	2-0-0-0	0	2	-

Year 1 — Courses: 16 · Credits: 43 · Contact Hours/Week: 71

Regular (Semester) Track — Year 2

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
S03	25SC2008	FSWD	Full Stack Web Development	ESC	0-0-2-4	2	6	DAST
	25EC2206	EBSD	Embedded System Design And Iot	PCC	2-0-0-4	3	6	DISD
	25MT2022	MFEE	Mathematics For Electrical Engineering	BSC	2-2-0-0	4	4	-
	25EC1302	AECD	Analog Electronic Circuit Design	PCC	2-0-2-0	3	4	DISD
	SCE-1	SCE1	Science Elective – 1 (Choose one)	SE	3-0-2-0	4	5	-
	25EE2101	ECES	Electric Circuits & Energy System Fundamentals	PCC	2-0-2-0	3	4	BEEC
	VAC-1	VAC1	Value Added Course 1 - Sports	VAC	0-0-0-2	0	2	-
S04	25EE2202	EMDE	Electric Motors, Drives & Electromechanical Systems	PCC	3-0-2-0	4	5	BEEC
	25EE2203	EGTD	Electric Power Generation Transmission And Distribution	PCC	3-1-0-0	4	4	ECES
	25AD2102	AIML	Artificial Intelligence And Machine Learning	ESC	3-0-2-0	4	5	PYPG
	25EC2308	ESDT	Embedded Software Development	PCC	2-0-0-4	3	6	EBSD
	25EE2204	PEWC	Power Electronics & Wide-Bandgap Converters	PCC	2-0-0-4	3	6	ECES
	25EC2309	IPCI	Iot Protocols And Cloud Integration	PCC	2-0-2-0	3	4	EBSD
	VAC-2	VAC2	Value Added Course 2 - Technical Global Certification 1	VAC	0-0-0-2	0	2	-
Summer 2	25IE3041	TEIP	Technical Internship	PRI	0-0-0-4	0	4	-
	25UC0009	ECEN	Ecology And Environment	AUC	2-0-0-0	0	2	-

Year 2 — Courses: 16 · Credits: 40 · Contact Hours/Week: 69

Regular (Semester) Track — Year 3

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
S05	25CS3112	DLAC	Deep Learning With Accelerated Computing	PCC	2-0-0-4	3	6	AIML
	25EE3105	CSEA	Control Systems For Electrical & Energy Applications	PCC	3-0-2-0	4	5	-
	25CS3113	EATM	Embedded Ai And Tiny Ml	PCC	2-0-2-0	3	4	EBSD
	25EE3106	MPSG	Modern Power Systems & Smart Grid Technologies	PCC	2-0-0-4	3	6	EGTD
	25CS2207	GAPE	Generative Ai And Ai Platform Engineering	PCC	2-0-0-4	3	6	AIML
	PE-1	PEC1	Professional Elective - 1	PEC	2-0-0-4	3	6	-
	VAC-3	VAC3	Value Added Course 3 - Technical Global Certification 2	VAC	0-0-0-2	0	2	-
S06	FL	FRLG	Foreign Language	HSS	3-0-0-0	3	3	-
	25EE3207	IAPS	Industrial Automation, Plc & Scada Systems	PCC	3-0-2-0	4	5	CSEA
	25EE3208	PSPC	Power System Protection And Control	PCC	2-0-0-4	3	6	EGTD
	PE-2	PEC2	Professional Elective - 2	PEC	2-0-0-4	3	6	PE-1
	PE-3	PEC3	Professional Elective - 3	PEC	2-0-0-4	3	6	PE-1
	25IE4101	TMPR	Term Paper	PRI	0-0-6-0	3	6	-
	VAC-4	VAC4	Value Added Course 4 - Technical Global Certification 3	VAC	0-0-0-2	0	2	-

Year 3 — Courses: 14 · Credits: 38 · Contact Hours/Week: 69

Regular (Semester) Track — Year 4

Term	Code	Acr	Course	Cat	L-T-P-S	CR	CH	Pre-req
S07	25EE4109	EVBT	Electric Vehicle Systems & Battery Technologies	PCC	3-0-0-0	3	3	PEWC
	25EE4110	REGI	Renewable Energy Systems & Grid Integration	PCC	3-0-2-0	4	5	EGTD
	PE-4	PEC4	Professional Elective - 4	PEC	2-0-0-4	3	6	PE-1
	OEC-1	OEC1	Open Elective – 1 (Pick a stream)	OE	4-0-0-0	4	4	-
	25IE4053	CAP1	Capstone Project - 1	PRI	0-0-8-8	6	16	TMPR
	25UC0026	HVEG	Human Values, Gender Equality & Professional Ethics	HSS	2-0-0-0	2	2	-
S08	PE-5	PEC5	Professional Elective - 5	PEC	2-0-0-4	3	6	PE-1
	SCE-2	SCE2	Science Elective – 2 (Choose one)	SE	3-0-2-0	4	5	-
	OEC-2	OEC2	Open Elective – 2 (Pick a stream)	OE	4-0-0-0	4	4	-
	MGEL	MGEL	Management Elective (Pick a stream)	MGEL	4-0-0-0	4	4	-
	25IE4054	CAP2	Capstone Project - 2	PRI	0-0-8-8	6	16	CAP1
	25UC0008	INCT	Indian Constitution	AUC	2-0-0-0	0	2	-

Year 4 — Courses: 12 · Credits: 43 · Contact Hours/Week: 73

Regular (Semester) Track total — Courses: 58 · Credits: 164 · Contact Hours/Week: 282

ALL MANDATORY COURSES · Y26 B.TECH

Electrical & Electronics Engineering

Detailed content for every mandatory course — each formatted as its individual course handout (one course per page), including both Regular and PBL variants (distinct course codes) where offered, sorted A–Z by course title. Experiential PRI courses are in a separate document.

49 course handouts (Regular + PBL combined where identical), A–Z by title.

Agentic Embedded Systems & Intelligent Edge Orchestration AGEM 26EC4118E

L-T-P-S	3-0-0-6	Credits	3	Contact Hrs/Wk	9
Trimester	T10	Mode	PBL (Project-Based)	Category	PCC
Pre-requisite	ESDI				

Course Outcomes (COs)

	Statement	BTL
CO1	Evaluate edge-orchestration architectures — cloud-heavy, gateway-coordinated, fully-distributed, edge-agentic — against latency, bandwidth, privacy, and reliability requirements and select an appropriate topology.	BTL5 · Evaluate
CO2	Apply edge-compute architecture knowledge — MCU-class (STM32H7), embedded-Linux-class (Raspberry Pi), edge-GPU-class (Jetson Orin, Coral) — and select hardware tiers based on inference workload and power budget.	BTL3 · Apply
CO3	Apply edge-cloud communication patterns — MQTT for telemetry, gRPC streaming for richer interactions, offline-first design with store-and-forward, the bandwidth-and-cost discipline — to build resilient fleets.	BTL3 · Apply
CO4	Apply distributed edge inference — model-pipeline placement (which model runs where), cascade inference (cheap model triggers expensive one), split inference for large models, the latency-vs-cost optimisation — to design efficient inference systems across an edge fleet.	BTL3 · Apply
CO5	Apply agentic-AI patterns from AAAE adapted to the edge — on-device agents (lightweight LLMs or rules-based), cloud-coordinated multi-agent planning, the edge-fall-through pattern when cloud is unreachable — to build edge fleets with autonomy.	BTL3 · Apply
CO6	Apply fleet-operations practices — OTA at fleet scale with A/B rollout and rollback, telemetry-driven monitoring, model versioning across heterogeneous devices, safety supervision and kill-switches — to operate intelligent edge fleets reliably.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PSO1
CO1	H	H	H	H	–	H
CO2	H	–	H	–	M	–
CO3	H	–	H	–	H	H
CO4	H	–	H	–	H	H
CO5	H	–	H	–	–	H
CO6	H	–	H	H	–	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The Edge-Coordinator System as a Whole** A working SmartCampus fleet on day one — students see four MCU nodes around the room reporting state to a Jetson gateway which reports to a cloud coordinator; one node is killed mid-demo and the system gracefully degrades.; The canonical edge-orchestration stack: cloud coordinator → gateway (regional aggregator) → edge nodes → embedded sensing/action.; The four edge-orchestration archetypes: cloud-heavy (everything goes up), gateway-coordinated (gateway is the brain), fully-distributed (peers coordinate), edge-agentic (each device decides locally with cloud as fallback).; The four engineering pressures that drive the choice: latency, bandwidth, privacy, reliability under disconnection.; Why this combination matters in 2026 onward: smart cities, industrial IoT, agricultural sensing, retail intelligence — all are edge fleets, all are getting agentic.; A reading of SmartCampus's codebase: the four MCU-firmwares (continuation of EATM), the gateway service (continuation of AAAE patterns at the edge), the cloud coordinator, the OTA pipeline.; Tracing one event end-to-end: occupancy detected on Node-3 → MQTT to gateway → gateway aggregates with other nodes → cloud-agent reasons about anomaly → notification → action.; What this course owns vs upstream: orchestration here; on-device ML in EATM; RTOS in ESDT; cloud agentic in AAAE.	CO1

	Title & Topics	CO
M2	**Edge Compute Architectures** The edge-compute spectrum: MCU-class (KB RAM, mW power) → embedded-Linux-class (GB RAM, W power) → edge-GPU-class (10s of GB RAM, 10s of W power).; MCU-class tier: STM32H7 / nRF52 / ESP32-S3; on-device tiny models; sub-1mA power; the territory of EATM.; Embedded-Linux tier: Raspberry Pi 4/5, Rockchip RK3588 boards; classical-ML and small DL workloads; ~5W power.; Edge-GPU tier: NVIDIA Jetson Orin Nano (40 TOPS @ 7-15W), Jetson Orin NX (100 TOPS @ 10-25W); transformer inference, vision DL at frame rate.; Edge-NPU tier: Google Coral, Hailo-8, Rockchip NPUs; INT8 acceleration without the GPU power budget.; Tier selection: latency requirement, model size, power budget, cost-per-unit; the engineering matrix that picks hardware.; Cross-tier model deployment: a model trained once, quantised differently per tier; the vendor toolchain landscape.; The Jetson Orin software stack: JetPack, CUDA on Jetson, TensorRT optimisations specific to Jetson, DeepStream for vision pipelines.	CO2
M3	**Edge-Cloud Communication** The edge-cloud communication budget: every byte costs power and money; designing protocols with this in mind.; MQTT as the canonical lightweight pub-sub: QoS levels, retained messages, last-will-and-testament; the broker as the central nervous system.; MQTT topic hierarchy design: per-device, per-region, per-tenant; the discipline that allows scaling.; MQTT vs alternatives: AMQP (heavier), CoAP (UDP-based, even lighter), WebSocket (web-friendly); when each fits.; gRPC streaming for richer edge-cloud interactions: bidirectional streams, server-side streaming for telemetry, the protobuf-based efficiency.; Offline-first design: the device must continue to function without connectivity; store-and-forward of telemetry; reconciliation on reconnect.; Bandwidth optimisation: delta encoding, message compression, data-rate adaptation; the discipline of shipping what matters.; Security at the edge-cloud boundary: TLS with mutual auth (mTLS), per-device certificates, certificate rotation in OTA; the DPP (Device Provisioning Protocol) at preview level.	CO3
M4	**Distributed Edge Inference** The model-placement problem: where does each model run? On the MCU, on the gateway, in the cloud? The latency-cost-privacy trade-off matrix.; Cascade inference: a cheap model on every device, an expensive model on the gateway only invoked when needed; the canonical pattern for vision and audio.; Split inference for large models: the model is partitioned across devices; the early-layers-on-edge-late-layers-in-cloud variant.; Federated inference at intuition level: aggregating predictions from many devices without centralising raw data.; Latency-cost optimisation in practice: a worked example with the SmartCampus fleet showing per-tier inference costs and end-to-end latency.; Model warm-state management: keeping models loaded vs cold-loading; the trade-off with memory pressure on shared edge-GPU.; Inference pipelines using DeepStream / NVIDIA Triton on Jetson: declarative pipelines, batching, multi-stream concurrency.; The privacy story: why edge inference matters for regulated workloads (healthcare, finance, surveillance); DPDP / GDPR alignment.	CO4
M5	**Agentic Behaviour at the Edge** Adapting AAAE's agentic patterns to the edge: every device runs a small agent (rules-based or tiny-LLM) that decides locally; cloud coordinates.; The edge-fall-through pattern: if cloud is reachable, use the powerful cloud agent; if not, fall through to a tinier on-device agent that does the best it can.; Tiny-LLMs on the edge: 1B-parameter quantised models running on Jetson Orin; the capability frontier.; Hybrid rules-and-LLM at the edge: deterministic rules for the known cases, LLM for the long tail; the cost/predictability optimisation.; Multi-agent coordination across an edge fleet: planner-on-cloud + executors-on-edge; the latency-budgeted coordination protocol.; Local-first decision-making: when each device must decide before any network round-trip is possible (safety-critical industrial control).; Memory at the edge: the device's history of observations; what to forget vs what to keep; the bandwidth-aware sync.; A walk-through of SmartCampus's agentic layer: the rules-based local decider on each MCU, the Jetson-resident planner agent, the cloud coordinator.	CO5
M6	**Fleet Operations** OTA at fleet scale: revisiting ESDT's OTA pipeline now for hundreds-to-thousands of devices; the staged rollout pattern.; A/B rollout for firmware AND for models: per-device cohort assignment, success-metric monitoring, automatic rollback on regression.; Heterogeneous fleet management: different hardware versions, different firmware versions, different model versions in the wild simultaneously; the compatibility-matrix discipline.; Telemetry-driven monitoring: the same Golden Signals from SDSR adapted to the edge — latency, accuracy, drift, power consumption, connectivity health.; Drift detection at the edge: per-device input-distribution monitoring; aggregated drift detection at the gateway; the retraining trigger.; Safety supervision: each device runs an independent safety supervisor that can override the agent (e.g. shut down on overheating); the design discipline.; Kill-switch architecture: the company-wide ability to disable a model, a feature, or an entire device; the discipline of reversibility.; A glance at the regulatory landscape: data-residency laws, edge-AI compliance (DPDP in India, GDPR in EU); what fleet operators must track.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**AI at the Edge — Solving Real-World Problems with Embedded Machine Learning**	Daniel Situnayake & Jenny Plunkett	O'Reilly	2023

#	Title	Authors	Publisher	Year
2	**Edge Computing — Models, Technologies and Applications**	Javid Taheri & Schahram Dustdar (eds.)	IET	2020
3	**Designing Distributed Systems — Patterns and Paradigms for Scalable, Reliable Services**	Brendan Burns	O'Reilly	2018
4	**IoT and Edge Computing for Architects — Implementing Edge and IoT Systems from Sensors to Clouds with Communication Systems, Analytics, and Security**	Perry Lea	Packt	2020 (2nd Ed.)
5	**Building Machine Learning Powered Applications**	Emmanuel Ameisen	O'Reilly	2020
6	**MQTT Essentials — A Lightweight IoT Protocol**	Gastón C. Hillar	Packt	2017
7	**AI Engineering — Building Applications with Foundation Models**	Chip Huyen	O'Reilly	2024

Analog Electronic Circuit Design

AECD REG · 26EC1203 PBL · 26EC1203E

L-T-P-S	2-0-0-4 / 3-0-0-6	Credits	3	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S04 · T04	Mode	Regular & PBL	Category	PCC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Read and analyse a complete analog signal chain as a whole system - its blocks, signal flow and specifications - using the linear circuit-analysis toolkit (Kirchhoff's laws, network theorems, AC/phasor and frequency-response methods) established here because it exists nowhere else.	BTL3 · Apply
CO2	Design the signal chain's active stages using the operational amplifier as a black-box building block - applying feedback to build precise amplifiers, instrumentation amplifiers, active filters and oscillators.	BTL6 · Create
CO3	Open the op-amp - explain and analyse the analog-IC building blocks inside it (the differential pair, current mirror, active load, cascode, gain and output stages, and multistage amplifiers) - the explicit foundation for VLSI.	BTL6 · Create
CO4	Open the building block - bias a single transistor, build its small-signal model, and design and analyse single-stage BJT and MOSFET amplifiers including their frequency response.	BTL6 · Create
CO5	Open the device - explain the operation of the diode, BJT and MOSFET - and design diode circuits and the regulated power supply the system runs on.	BTL6 · Create
CO6	Reach the semiconductor bedrock, complete the power-amplifier and voltage-regulator stages, and integrate and measure the complete discrete signal chain - then map the path forward to VLSI and board-level design.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO8	PO11
CO1	**H**	M	M	L	M	–	L
CO2	**H**	**H**	M	M	M	–	L
CO3	**H**	**H**	**H**	M	M	–	L
CO4	**H**	**H**	**H**	M	M	–	L
CO5	**H**	**H**	**H**	M	M	–	M
CO6	**H**	M	**H**	M	**H**	M	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - The Whole Analog Signal Chain & the Language to Read It** Outside-In day one: build and MEASURE a complete analog signal chain - a sensor/audio front-end with a power supply, amplification, conditioning/filtering, an output stage and a test oscillator - using IC black boxes (an op-amp IC, a regulator IC, a power module); the working system you will open, layer by layer.; The signal chain as a system: blocks and signal flow (source → preamp → gain/conditioning → filter → output → load), and the key specifications (gain, bandwidth, input/output impedance, filtering, power) each block must meet.; Circuit quantities and elements: voltage, current, power; ideal sources; the resistor, capacitor and inductor and their v-i relationships.; Kirchhoff's laws (KVL, KCL) and systematic analysis: node-voltage and mesh-current methods; voltage/current dividers.; Network theorems: superposition, Thevenin and Norton equivalents, maximum power transfer - and the loading idea that motivates buffers.; First-order transients: the RC and RL circuit and time constants - the behaviour behind coupling and timing.; AC steady-state analysis: sinusoids, phasors and complex impedance; analysing R-L-C circuits in the frequency domain.; Frequency response and Bode-plot intuition: how a circuit's gain and phase vary with frequency; the decibel and cutoff frequency.; Resonance in RLC circuits (introduction): resonant frequency, bandwidth and quality factor Q - the basis of tuning and filtering.; Why this is the language for the whole course: every stage you open in the modules ahead is analysed with exactly these tools - on the working system you just built.	CO1

	Title & Topics	CO
M2	**Module 2 - Designing with the Op-Amp (the Building Block, as a Black Box)** Opening the first layer (functionally): the chain's amplification, conditioning, filtering and signal generation are all done with the op-amp - the most powerful analog building block - which we first USE as an ideal black box.; The ideal operational amplifier: the model and the two 'golden rules'; why very high gain plus feedback gives precise, predictable behaviour.; Negative feedback: the concept and the four feedback topologies; the benefits - gain stability, increased bandwidth, modified input/output impedance, reduced distortion - and the idea of stability.; Core op-amp amplifiers: the inverting and non-inverting amplifier, the voltage follower (buffer), and summing and difference amplifiers.; The instrumentation amplifier: amplifying a small differential sensor signal with high CMRR - the chain's front-end.; The integrator and differentiator and their uses.; Op-amp non-idealities and their design impact: input offset voltage, bias/offset currents, finite open-loop gain, finite bandwidth (gain-bandwidth product), slew rate, and CMRR/PSRR.; Active filters: why active (op-amp) filters beat passive ones; first-order low-pass/high-pass; the second-order Sallen-Key low-pass, high-pass and band-pass (cutoff, Q, response).; Oscillators and the Barkhausen criterion: RC oscillators (Wien-bridge, phase-shift); LC and crystal oscillators (overview) - building the chain's test signal source.; Designing and building the chain's gain, instrumentation, filtering and oscillator stages with op-amp ICs - a fully working active signal chain (the op-amp's internals are opened in Module 3).	CO2
M3	**Module 3 - Opening the Op-Amp: the Analog-IC Building Blocks (the VLSI Foundation)** Opening the op-amp black box: the chip you used in Module 2 is built from a small set of transistor-level cells - here they are.; The differential amplifier (the diff pair): the BJT and MOSFET differential pair; differential and common-mode signals; common-mode rejection ratio (CMRR) - the input stage of essentially every op-amp and analog IC.; Current mirrors and current sources: the basic current mirror (BJT and MOS) used to set and copy bias currents precisely - how an analog chip biases itself.; Active loads: replacing resistors with current sources/mirrors to get high gain in a small area - the essence of IC amplifier design.; The cascode: stacking devices for high gain and wide bandwidth (taming the Miller effect) - a key VLSI structure.; Multistage amplifiers: cascading stages, inter-stage loading, overall gain; the gain stage with an active load and the output stage.; Putting it together - the op-amp's internal architecture: a differential input stage + a high-gain active-load stage + an output stage, all biased by current mirrors (e.g., a two-stage CMOS op-amp).; Mapping back: how the op-amp's datasheet specs (gain, CMRR, slew rate, bandwidth) arise from these internal blocks.; Why this is the VLSI foundation: the circuit you USED as a discrete designer is what you DESIGN as a VLSI engineer - this is the bridge to CMOS VLSI and Analog VLSI.; Analysing and simulating these building blocks (and a discrete op-amp/amplifier) to see the chip's insides in action - the transistors inside are opened in Module 4.	CO3
M4	**Module 4 - Opening the Block: the Single-Transistor Amplifier (BJT & MOSFET)** Opening the building block: each cell of Module 3 is built from single-transistor amplifiers - how does one transistor amplify?; DC biasing and the operating point (Q-point): why a stable bias is essential; BJT biasing (voltage-divider/self-bias and stability) and MOSFET biasing.; The small-signal concept: separating the DC bias from the small AC signal; linearising around the Q-point.; Small-signal models: the BJT hybrid-pi model (gm, r-pi, ro) and the MOSFET small-signal model (gm, ro) - the basis of all amplifier analysis.; Single-stage BJT configurations: common-emitter (voltage gain), common-collector / emitter-follower (buffer), common-base; gain and input/output impedance.; Single-stage MOSFET configurations: common-source (gain), common-drain / source-follower (buffer), common-gate - the VLSI counterparts of the BJT stages.; Choosing a configuration: matching gain, input and output impedance to the role in the signal chain.; Frequency response of a single stage: coupling/bypass capacitors and the low-frequency cutoff; device/load capacitance and the high-frequency cutoff; the Miller effect; bandwidth and the gain-bandwidth idea.; How the diff pair and gain stage of Module 3 are built from these single-transistor stages - connecting the layers.; Designing, simulating and building a discrete transistor amplifier stage (BJT and a MOS version) - the discrete preamplifier of the chain.	CO4
M5	**Module 5 - Opening the Device: Diodes, BJTs, MOSFETs & the Power Supply** Opening the transistor: what a diode, a BJT and a MOSFET actually are and how they work - the devices every amplifier is built from.; Semiconductors and the PN junction (operational level): conduction, doping and the junction; the depletion region (intuition - deep physics is the bedrock of Module 6 and CMOS VLSI's).; The diode: the i-v characteristic, forward/reverse behaviour, practical models (ideal, constant-drop, piecewise-linear); the Zener diode.; The BJT: structure (NPN/PNP), the i-v characteristics and the regions of operation (cutoff, active, saturation) - the discrete workhorse.; The MOSFET: structure (NMOS/PMOS), the i-v characteristics and the regions of operation (cutoff, triode, saturation) - the device VLSI is built from.; Rectifier circuits: half-wave, full-wave (centre-tapped) and the bridge rectifier; average/RMS and the role of the diode.; Filtering and ripple: the smoothing capacitor; ripple voltage and its dependence on load and capacitance.; Wave-shaping diode circuits: clippers (limiters) and clampers and where they are used; Zener voltage reference.; Building the project's power supply: from AC to a clean DC rail (rectifier + filter + Zener), replacing the regulator-IC black box of Module 1 with an understood, built supply (full regulators in Module 6).; How the device characteristics explain the small-signal parameters (gm, etc.) used in Module 4 - connecting the layers.	CO5

	Title & Topics	CO
M6	**Module 6 - The Bedrock, Power & Regulation, the Whole Re-Grounded & the Frontier** Reaching the bedrock: a fuller picture of the semiconductor and the PN junction - the physics beneath the device models (intuition; depth is CMOS VLSI's) - so nothing remains a black box.; Power amplifiers and output stages: Class A, Class B, Class AB and Class C; efficiency, crossover distortion and the push-pull stage - the stage that drives the chain's real load.; Thermal considerations and heat sinking (introduction) for power stages.; Voltage regulators: linear series and shunt regulators; the IC regulator (78xx / LM317) and the Zener reference; line and load regulation; an awareness note on switching regulators - completing the supply opened in Module 5.; The integrated, measured system (capstone): assembling the complete discrete signal chain - regulated supply, preamplifier, gain/instrumentation and filtering, output power stage and test oscillator - now understood and built top to bottom; measuring end-to-end gain, bandwidth, distortion and noise.; Measurement and verification: the oscilloscope, function generator and multimeter to characterise the system, reconciling theory, simulation and measurement.; The frontier - from discrete to integrated: how the very stages you opened (diff pair, current mirror, gain and output stages) become an analog IC - the explicit bridge to CMOS VLSI and Analog VLSI.; PCB and board-level frontier: from breadboard to a designed board (KiCad) - the discrete-design path.; The forward map: how this single foundation feeds Modern Communication Systems, CMOS/Analog VLSI, Signal Processing and Embedded Systems.; Reflection: an analog system understood and built from the whole system down to the device - and the strong base for both the chip and the board.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Microelectronic Circuits**	Adel S. Sedra & Kenneth C. Smith	Oxford University Press	2020 (8th ed.)
2	**Fundamentals of Microelectronics**	Behzad Razavi	Wiley	2021 (2nd ed.)
3	**Design with Operational Amplifiers and Analog Integrated Circuits**	Sergio Franco	McGraw-Hill	2015 (4th ed.)
4	**Electronic Devices and Circuit Theory**	Robert Boylestad & Louis Nashelsky	Pearson	2014 (11th ed.)
5	**Analysis and Design of Analog Integrated Circuits**	Paul R. Gray, Paul J. Hurst, Stephen H. Lewis & Robert G. Meyer	Wiley	2009 (5th ed.)
6	**Electronic Devices and Circuits / Linear Integrated Circuits**	S. Salivahanan et al.	McGraw-Hill / Tata McGraw-Hill	2019

Autonomous Robotic Systems & Human-Robot Interaction

ARBS 26EC4117E

L-T-P-S	3-0-3-0	Credits	3	Contact Hrs/Wk	6
Trimester	T10	Mode	PBL (Project-Based)	Category	PCC
Pre-requisite	RBPR				

Course Outcomes (COs)

	Statement	BTL
CO1	Evaluate autonomous-robot-system architectures — purely deliberative, layered hybrid (Brooks-style subsumption to modern Three-Layer), behaviour-tree-orchestrated, foundation-model-orchestrated — against task complexity, real-time requirements, and safety needs.	BTL5 · Evaluate
CO2	Apply robot autonomy architecture — mission management with behaviour trees, task planning at intuition level, deliberative-reactive hybrid execution, the mode-switching state machine — to give a robot multi-task capability.	BTL3 · Apply
CO3	Apply human-robot interaction patterns — voice recognition + LLM-based intent understanding, gesture recognition at intuition level, visual feedback (display, lights), the intent-disclosure principle, social-signal recognition — to enable natural human-robot communication.	BTL3 · Apply
CO4	Apply social and human-aware navigation — proxemics-aware cost maps, the social-force model, person-tracking-aware planning, the freezing-robot problem revisited at social depth, group-aware navigation — to operate robots safely around humans.	BTL3 · Apply
CO5	Analyze learning-based autonomy at intuition level — reinforcement learning for robotics, the sim-to-real gap and how it is bridged, imitation learning, foundation models for robotics (RT-2, Open X-Embodiment) — and reason about when learning-based components belong in a robot stack.	BTL4 · Analyze
CO6	Apply safety, reliability, and deployment-lifecycle practices — independent safety supervisors at depth, the certification landscape (ISO 13482 for personal-care robots, ISO 10218 for industrial) at preview level, the field-deployment lifecycle, the post-deployment monitoring discipline — to deliver robots that are safe in the real world.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO9	PSO1
CO1	H	H	H	H	–	–	–	H
CO2	H	–	H	–	H	–	–	H
CO3	H	–	H	–	–	H	M	H
CO4	H	–	H	–	–	H	–	H
CO5	H	H	M	–	–	–	–	H
CO6	H	–	H	H	–	H	M	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The Autonomous Robot as a System** A working HelpBot on day one — students give the robot a voice command ("HelpBot, go meet me at the printer"), watch it confirm intent on its display, navigate there socially, and announce arrival before any internals.; The complete autonomous-robot stack visible: HRI (voice in, TTS out, display) → autonomy (behaviour-tree mission manager) → navigation (RMNC stack) → perception (RPCS stack) → safety supervisor (always watching) → hardware (FRMS-built KartBot).; Robot autonomy architecture history: Brooks-style subsumption (purely reactive), classical deliberative (sense-plan-act), the modern Three-Layer architecture (deliberative + executive + reactive), behaviour-tree orchestrated (current mainstream), foundation-model orchestrated (research frontier).; The autonomy-vs-supervision trade-off in robotics: tele-op, shared autonomy, supervised autonomy, full autonomy; how each is chosen by application.; The Three-Laws-of-Robotics naivety vs the real safety-engineering discipline: hard-coded safety supervisors, layered safety, formal-verification-where-it-counts.; A reading of HelpBot's architecture: how the HRI layer feeds the mission manager, how the mission manager invokes the Nav2 stack, how the safety supervisor monitors everything from outside.; Tracing one user request end-to-end: voice → STT → LLM intent classifier → mission manager → behaviour tree → Nav2 → KartBot motors → arrival announcement.; What this course owns vs upstream: ARHR owns autonomy + HRI + safety + deployment; the upstream chain owns foundations + perception + motion.	CO1
M2	**Robot Autonomy Architecture** Mission management as the highest-level concern: a queue of tasks, the currently-executing task, the abort/resume semantics.; Behaviour trees revisited at the autonomy depth: composite nodes for mission decomposition, decorators for retry/timeout/fallback, the integration with Nav2 already seen in RMNC.; BehaviorTree.CPP at depth: writing custom nodes, plugging into the Nav2 BT navigator, the testability discipline.; Task planning at intuition level: HTN (Hierarchical Task Networks), PDDL (Planning Domain Definition Language); when planning beats hand-coded behaviour trees.; The deliberative-reactive split in modern robots: a deliberate plan at low frequency (1Hz), a reactive controller at high frequency (50Hz), the decoupling that makes both work.; Mode-switching state machines: the canonical states (idle, navigating, interacting, recovering, faulted); the transitions and the guards.; Foundation models for high-level robot reasoning: using a cloud LLM to interpret novel commands; the latency-and-reliability discipline that wraps it.; A walk-through of HelpBot's mission manager: the BT structure, the modes, the transitions; how a user request becomes an executed mission.	CO2
M3	**Human-Robot Interaction** Voice input on a robot: revisiting EATM's on-device keyword-spotting (for wake-word) + cloud STT (for the body of the command); the offline-fall-through mode.; LLM-based intent understanding: using a cloud LLM (continuation of AAAE) to map free-form voice to structured intent; the discipline of validating before acting.; Speech output: TTS quality matters; on-device options vs cloud TTS; the intelligibility-vs-latency trade-off.; Visual feedback on a small robot display: the discipline of showing what the robot is about to do (intent disclosure).; Light-based feedback: ring LEDs as a low-bandwidth status display; the design language (waiting, listening, moving, error).; Gesture recognition at intuition level: pointing, "stop" gestures, beckoning; the deep-learning-for-gestures landscape.; Social signal recognition: detecting "the human is busy", "the human is engaging", "the human looks distressed"; the responsibility this confers.; The intent-disclosure principle: the robot must communicate what it intends to do BEFORE doing it; the design pattern that makes humans trust robots.	CO3
M4	**Social and Human-Aware Navigation** Proxemics: Hall's zones (intimate, personal, social, public); how robots respect them.; Proxemics-aware cost maps: inflating obstacles around tracked humans more than around walls; the layered-cost-map plug-in for Nav2.; The social-force model revisited from RMNC at depth: humans as agents in a force field; the robot navigates as one such agent.; Person tracking integration: the RPCS tracker outputs feed into a social-cost-map; identities persist across short occlusions.; The freezing-robot problem at social depth: the robot must maintain forward progress in a corridor with oncoming people; the merge-or-yield decision.; Group-aware navigation: don't walk through a group of people having a conversation; the conversational-formation detector at intuition level.; Speed-and-acceleration discipline near humans: hard caps; the comfort-vs-progress trade-off.; Cultural variation in proxemics at preview level: the same robot may need different defaults for different deployment regions; the design discipline.	CO4

	Title & Topics	CO
M5	**Learning-Based Autonomy at Intuition Level** Reinforcement learning for robotics at intuition level: state, action, reward; the policy and value-function abstractions.; The sim-to-real gap: why a policy trained in simulation often fails on real hardware; domain randomisation as the canonical mitigation.; Imitation learning: behavioural cloning, DAgger; learning from human demonstrations.; Inverse RL at intuition level: learning the reward function from demonstrations, then planning under it.; Diffusion policies: the recent trend of generating trajectories with diffusion models; why it beats simple behavioural cloning on multimodal tasks.; Foundation models for robotics: RT-2, Open X-Embodiment, the embodied-LLM trend; the capability frontier as of 2026.; The hybrid pattern: classical planning-and-control for the safety-critical core, learning-based perception on top, learning-based high-level reasoning for novel tasks; the layered-trust architecture.; When NOT to use learning: for the core safety supervisor, for hard real-time control loops, for any function that must be certified; the discipline of "use learning where the alternative is worse, not because it's fashionable".	CO5
M6	**Safety, Reliability, and Deployment** The independent safety supervisor at depth: a separate process (or chip) that monitors the system and can override; the design principle of "the supervisor cannot be bypassed by the autonomy stack".; Hardware safety: e-stop button, safety-rated bumpers, safety-rated speed limiters; the discipline of "if the software fails, the hardware contains the failure".; The certification landscape at preview level: ISO 13482 (personal-care robots), ISO 10218 (industrial robots), ISO 26262 (automotive functional safety), DO-178C (aviation); what each requires.; Functional-safety reasoning at intuition level: hazard analysis, ASIL/SIL levels, failure-mode analysis; the discipline that gets a robot through compliance.; Real-world deployment lifecycle: pilot deployment → controlled rollout → general availability → fleet operation; the staged-trust pattern.; Post-deployment monitoring: per-deployment telemetry, anomaly detection, the field-incident response process.; The continuous improvement loop: deployments produce data → data trains better models → models go through the entire safety pipeline → deployed; the rate at which robots get better is governed by this loop.; Case studies of real-world robot deployments and their failure modes: Knightscope, Roomba, Boston Dynamics Spot in industrial settings; the lessons.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Springer Handbook of Robotics**	Bruno Siciliano & Oussama Khatib (eds.)	Springer	2016 (2nd Ed.)
2	**Probabilistic Robotics**	Sebastian Thrun, Wolfram Burgard & Dieter Fox	MIT Press	2005
3	**Human-Robot Interaction — An Introduction**	Christoph Bartneck, Tony Belpaeme, Friederike Eyssel, Takayuki Kanda, Merel Keijsers & Selma Šabanović	Cambridge University Press	2024 (2nd Ed.)
4	**Behavior Trees in Robotics and AI — An Introduction**	Michele Colledanchise & Petter Ögren	CRC Press	2018
5	**Robot Learning — Algorithms and Applications**	Pieter Abbeel et al. (eds., open lecture-notes compilation)	UC Berkeley / Free PDF	2024
6	**A Survey of Behavior Trees in Robotics and AI (Robotics and Autonomous Systems journal article)**	Matteo Iovino, Edvards Scukins, Jonathan Styrud, Petter Ögren & Christian Smith	Elsevier	2022
7	**Modern Robotics — Mechanics, Planning, and Control**	Kevin M. Lynch & Frank C. Park	Cambridge University Press	2017

Communication Skills for Engineers

CSFE REG · 26UC1204 PBL · 26UC1204E

L-T-P-S	0-0-4-0 / 3-0-0-0	Credits	2	Contact Hrs/Wk	4 / 3
Term (Reg · PBL)	S02 · T02	Mode	Regular & PBL	Category	HSS
Pre-requisite	LSFE				

Course Outcomes (COs)

	Statement	BTL
CO1	Demonstrate effective listening - comprehending, note-taking and responding accurately across accents, contexts and academic/professional content - at a certification-aligned level.	BTL4 · Analyze
CO2	Speak fluently, accurately and intelligibly in extended professional contexts - presentations, public speaking, group discussions and interviews - at a certification-aligned level.	BTL6 · Create
CO3	Read efficiently and critically - skimming, scanning, comprehending and evaluating academic, technical and professional texts - at a certification-aligned level.	BTL4 · Analyze
CO4	Write clear, well-structured extended texts - reports, technical documents, proposals and certification-style tasks - at a certification-aligned level.	BTL6 · Create
CO5	Deliver effective presentations and public speaking - structuring, designing visuals, and presenting with confidence and handling questions - in academic and professional settings.	BTL6 · Create
CO6	Integrate LSRW and professional communication for high-stakes scenarios - interviews, group discussions and the workplace - and demonstrate certification readiness.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO7	PO8	PO9	PO10	PO11
CO1	–	–	H	–	M
CO2	–	–	H	–	M
CO3	–	M	H	–	M
CO4	–	–	H	M	M
CO5	–	H	H	–	M
CO6	L	M	H	M	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - Listening Skills (the 'L' of LSRW)** Why listening is the most-used and least-taught language skill; the cost of poor listening in engineering work.; Types of listening: for gist, for specific information, for detail, and for inference - and when each is needed.; Listening across accents and varieties of English: training comprehension of global/native and non-native speakers (as certifications and workplaces require).; Note-taking while listening: methods (Cornell, outlining, symbols/abbreviations) for lectures, meetings and instructions.; Listening to academic and professional content: lectures, presentations, technical talks, and podcasts.; Listening in interaction: following a conversation/meeting, catching turn-cues, and responding appropriately.; Inference and critical listening: detecting attitude, opinion, implication and emphasis beyond the literal words.; Common listening-comprehension task types (as in proficiency tests): multiple choice, gap-fill, matching - strategies for each.; Listening with AI: using AI/audio tools for graded listening practice and feedback, and verifying comprehension.; Self-assessing and improving listening: diagnosing weaknesses (speed, vocabulary, accent) and targeted practice.	CO1

	Title & Topics	CO
M2	**Module 2 - Speaking Skills & Fluency (the 'S' of LSRW)** The components of speaking proficiency: fluency, accuracy, range (vocabulary/grammar), pronunciation and coherence - the criteria certifications assess.; Pronunciation for intelligibility: stress, rhythm and intonation at sentence level; clarity over accent.; Extended speaking: organising a spoken response with structure (point-reason-example), and speaking to time.; Fluency development: reducing hesitation and fillers, paraphrasing when stuck, and building automaticity through practice.; Interactive speaking: group discussions - contributing, building on others, agreeing/disagreeing, and managing turn-taking.; Speaking under assessment conditions: the spoken tasks typical of proficiency tests (describe, compare, opine, discuss) and strategies.; Public speaking foundations: managing nerves, voice projection, pace, eye contact and body language.; Impromptu and prepared speaking: thinking on your feet vs structured delivery.; Speaking with AI practice partners: rehearsing spoken tasks and getting fluency/pronunciation feedback - verified.; Self-assessing speaking against certification criteria and targeted improvement.	CO2
M3	**Module 3 - Reading Skills (the 'R' of LSRW)** Reading purposes and strategies: skimming for gist, scanning for specifics, and intensive reading for full comprehension.; Reading speed and efficiency: techniques to read faster without losing comprehension.; Comprehending complex texts: main idea vs supporting detail, text structure and signposting.; Reading technical and academic English: handling dense, information-rich engineering texts, manuals and papers (the kind engineers must read).; Vocabulary in reading: inferring meaning from context; coping with unknown words without stopping.; Critical reading: distinguishing fact from opinion, detecting bias, evaluating arguments and evidence.; Reading-comprehension task types (as in proficiency tests): multiple choice, true/false/not-given, matching headings, gap-fill - strategies for each.; Reading for research and synthesis: extracting and integrating information from multiple sources (links to writing in Module 4).; Reading with AI: using AI to support (not replace) comprehension - summarising for checking, clarifying - while reading critically and verifying.; Self-assessing reading speed/comprehension against certification benchmarks and targeted practice.	CO3
M4	**Module 4 - Writing Skills: Reports, Technical Documents & Extended Writing (the 'W' of LSRW)** From sentences to extended writing: building on T01's drafting foundation to longer, structured documents.; The technical report: structure (summary, introduction, body, conclusion, recommendations), and writing each section.; Other professional documents: proposals, project documentation, formal letters and structured summaries.; Describing data and visuals in writing: charts, graphs and processes (a common certification and engineering task).; Coherence and cohesion at length: paragraphing, transitions, and maintaining a logical thread across a long document.; Academic/formal register and objectivity; precise, concise technical writing.; Argumentative and discursive writing: presenting and supporting a position (a common certification task type).; Certification-style writing tasks: the typical prompts (e.g., describe a graph, write an email/letter, write an essay) and how they are scored.; Editing and proofreading extended writing; using AI for feedback and revision while keeping one's own voice and verifying.; Referencing and integrity (foundational): citing sources and avoiding plagiarism in extended/technical writing.	CO4
M5	**Module 5 - Presentations & Public Speaking** The presentation as the engineer's high-stakes communication: where work is judged and careers advance.; Structuring a presentation: opening, clear message, logical body, strong close; signposting for the listener.; Designing effective slides/visuals: signal over noise, clarity, and supporting (not replacing) the speaker (distinct from product pitching, which is Design Thinking's).; Delivery skills: voice, pace, pause, emphasis, eye contact, posture and gesture; managing nervousness.; Presenting technical content clearly to varied audiences (expert and non-expert).; Handling questions and discussion: listening to the question, answering concisely, and managing difficult questions.; Group/team presentations: coordinating, handing over, and consistency.; Academic presentations: seminars, technical talks and (foundationally) conference-style presentation.; Rehearsal and feedback: using recording and AI feedback to refine delivery - verified and reflective.; Self- and peer-assessment of presentations against clear criteria.	CO5
M6	**Module 6 - Integrated Communication, Interviews & Certification Readiness** Integrating the four skills: real communication uses LSRW together (a meeting: listen, read materials, speak, write minutes).; Group Discussions (GD): the placement-critical skill - contributing substance, structure, and assertive-yet-respectful interaction (building on T01 etiquette).; Job interviews: preparing and practising - common questions, the STAR method for behavioural answers, and confident, clear responses.; Telephonic/video interviews and virtual communication: the modern reality and its etiquette (building on T01).; Workplace communication scenarios integrated: meetings, briefings, updates and documentation as combined-skill events.; The recognised English-proficiency certification (e.g., Cambridge Linguaskill): its structure, the four skills it tests, scoring (e.g., CEFR levels), and how it is used by employers/universities.; Certification preparation: practice tests, timing, task strategies for each skill, and interpreting/improving scores.; Building the certification-ready LSRW portfolio: assembling evidence of each skill and a practice-test result.; Lifelong communication improvement: self-assessment, feedback-seeking and continued practice (incl. responsible AI use).; Readiness check: a mock integrated assessment mirroring the certification, with reflection and an improvement plan.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Communication Skills for Engineers**	Sunita Mishra & C. Muralikrishna	Pearson	2011
2	**Technical Communication**	Meenakshi Raman & Sangeeta Sharma	Oxford University Press	2015 (3rd ed.)
3	**Practical English Usage**	Michael Swan	Oxford University Press	2016 (4th ed.)
4	**Official Cambridge Guide / Linguaskill & IELTS-family Preparation Materials**	Cambridge Assessment English	Cambridge University Press	current editions
5	**The Official Guide to TOEIC Tests**	Educational Testing Service (ETS)	ETS	current edition
6	**Talk Like TED / The Quick and Easy Way to Effective Speaking**	Carmine Gallo / Dale Carnegie	St. Martin's Press / Pocket Books	2014 / 1990

Computational Foundations for AI

CFAI REG · 26SC1306 PBL · 26SC1306E

L-T-P-S	2-0-0-4 / 3-0-0-6	Credits	3	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S02 · T03	Mode	Regular & PBL	Category	ESC
Pre-requisite	None (Regular) · MTAI (PBL)				

Course Outcomes (COs)

	Statement	BTL
CO1	Write Python programs for data and AI - building on prior programming experience - and work fluently in the modern computational environment (notebooks, packages, version control).	BTL3 · Apply
CO2	Represent and manipulate data as arrays with NumPy, and realise the linear algebra of AI (vectors, matrices, operations) computationally and efficiently.	BTL4 · Analyze
CO3	Load, clean, transform and combine real-world tabular data with pandas - the data-wrangling skills that dominate real AI work.	BTL4 · Analyze
CO4	Explore and visualise data and compute its descriptive statistics - exploratory data analysis (EDA) - to understand a dataset before modelling it.	BTL4 · Analyze
CO5	Implement the mathematics of learning in code - probability and statistics, a loss function, and gradient-descent optimisation - the computational engine that trains a model.	BTL6 · Create
CO6	Build, train and evaluate a first machine-learning model and assemble the end-to-end pipeline, and explain what machine learning is - the bridge to the Machine Learning course.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO8	PO9	PO11
CO1	**H**	M	M	L	**H**	–	–	M
CO2	**H**	**H**	M	M	**H**	–	–	M
CO3	**H**	**H**	M	**H**	**H**	–	M	M
CO4	**H**	**H**	**H**	M	**H**	–	–	M
CO5	**H**	**H**	**H**	**H**	**H**	–	–	M
CO6	**H**	**H**	**H**	M	**H**	M	M	**H**

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - From Java to Python & the Computational Environment** The north star (Outside-In framing): how does a computer learn from data? - the end-to-end pipeline (raw data → wrangle → explore → learn → predict) you will build, and the branch-flexible running project.; Why Python for data and AI: its role as the lingua franca of data science, AI and scientific computing.; Python for those who already program (the Java bridge): syntax, dynamic typing, indentation, and the key differences from Java.; Core Python: data types, variables, operators, strings; control flow; functions and arguments.; Python's data containers: lists, tuples, dictionaries and sets, and when to use each.; Pythonic idioms: list/dict comprehensions, slicing, iteration, and writing clean, readable Python.; Modules, packages and the ecosystem: import, pip, and the scientific stack overview (NumPy, pandas, matplotlib, scikit-learn).; The computational environment: Jupyter notebooks (and Colab), the REPL, and reproducible workflows.; Version control with Git/GitHub: tracking and sharing code and notebooks (reproducibility).; Reading and writing data and files in Python - the entry point to the data pipeline.	CO1
M2	**Module 2 - Data as Arrays: NumPy & the Math of AI in Code** Opening the first stage: data in AI is numbers in arrays - the NumPy ndarray as the fundamental data structure.; Creating and inspecting arrays; shape, dimensions and data types.; Indexing, slicing and boolean masking of arrays.; Vectorisation and why it matters: replacing loops with array operations for speed and clarity.; Broadcasting: how arrays of different shapes combine.; Realising linear algebra in code (from Mathematics for AI): vectors, matrices, element-wise operations, the dot product and matrix multiplication.; Common numerical operations: aggregations (sum, mean, min/max), axis operations, and reshaping.; Random numbers and reproducible randomness (seeds) - needed for sampling and ML.; Numerical thinking: floating-point, precision and efficiency awareness.; Representing a real dataset as arrays - the project data in numerical form.	CO2

	Title & Topics	CO
M3	**Module 3 - Working with Real Data: pandas & Data Wrangling** Opening the wrangle stage: real data arrives messy - pandas is the tool to tame it.; The pandas Series and DataFrame: the tabular data structures of data science.; Loading data: reading CSV and other formats; inspecting and summarising a DataFrame.; Selecting and filtering: by label and position (loc/iloc), and by condition.; Cleaning data: handling missing values, duplicates, and incorrect types.; Transforming data: creating and modifying columns, applying functions, and mapping.; Grouping and aggregation: split-apply-combine (groupby) to summarise data.; Combining data: merging and joining datasets, and concatenation.; Tidy data and reshaping (pivoting/melting) - getting data into a usable form.; Wrangling the project's real dataset into clean, analysis-ready form.	CO3
M4	**Module 4 - Seeing the Data: Visualisation & Exploratory Data Analysis** Opening the explore stage: before modelling, understand the data - the purpose of exploratory data analysis (EDA).; Plotting with matplotlib: the figure/axes model; line, scatter and bar plots; labels and titles.; Statistical visualisation with seaborn: distributions, histograms, box plots and pair/relationship plots.; Descriptive statistics in code (from Mathematics for AI): mean, median, variance, standard deviation, quantiles.; Distributions and their shape: skew, spread and outliers - and what they mean for modelling.; Relationships between variables: correlation, scatter relationships, and the caution that correlation is not causation.; Visualising categories and groups; comparing subgroups.; The EDA workflow: questions, plots, summaries and the insights that guide modelling.; Communicating findings: clear, honest, well-labelled visualisations.; A full EDA of the project's dataset - the insights that will shape the model.	CO4
M5	**Module 5 - The Math of Learning: Probability, Statistics & Optimisation in Code** Opening the learning stage: what does it mean for a machine to learn? - fitting a model to data by minimising error.; Probability in code (from Mathematics for AI): random variables, common distributions, sampling and simulation.; Statistics in code: estimating quantities from data, sampling variability, and the idea of inference.; The idea of a model: parameters, predictions, and the gap between prediction and truth.; The loss (cost) function: measuring how wrong a model is (e.g., mean squared error).; Optimisation - the engine of learning: minimising the loss; the gradient and what it tells us.; Gradient descent: implementing the update rule that improves the model step by step; the learning rate.; Watching learning happen: tracking the loss as it decreases, and convergence.; Why this is the heart of machine learning: nearly all of ML is loss + optimisation.; Implementing a loss and gradient descent on the project data - the learning engine, built by hand.	CO5
M6	**Module 6 - A First Learning Machine: Features, Model, Evaluation & the AI Frontier** Opening the model stage: from data to features - the feature matrix and the target.; Training and testing: the train/test split and why we never evaluate on training data (generalisation).; A first model from the foundations: linear regression trained by the gradient descent of Module 5 (the from-scratch model).; A first model with scikit-learn: the fit/predict workflow, and a simple classifier (e.g., k-nearest neighbours).; Evaluating a model: regression and classification metrics (error, accuracy) and reading them honestly.; Underfitting and overfitting (introduction): why a model can fail to generalise - the central ML concern.; The end-to-end pipeline integrated (the project): raw data → wrangle → explore → learn → evaluate.; What machine learning is (and is not): an honest framing and the bridge to the Machine Learning course.; Responsible AI (awareness): data bias, fairness and the ethics of learning from data.; The forward map: how this foundation feeds Machine Learning and the deep-learning, NLP, generative-AI and branch-specific AI courses ahead.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Python for Data Analysis**	Wes McKinney	O'Reilly	2022 (3rd ed.)
2	**Python Data Science Handbook**	Jake VanderPlas	O'Reilly	2023 (2nd ed.)
3	**Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow**	Aurelien Geron	O'Reilly	2022 (3rd ed.)
4	**Mathematics for Machine Learning**	Marc Peter Deisenroth, A. Aldo Faisal & Cheng Soon Ong	Cambridge University Press	2020
5	**Automate the Boring Stuff with Python**	Al Sweigart	No Starch Press	2019 (2nd ed.)

Control Systems for Electrical & Energy Applications

CSEE REG · 26EE2306 PBL · 26EE2306E

L-T-P-S	2-0-4-0 / 3-0-6-0	Credits	4	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S05 · T06	Mode	Regular & PBL	Category	PCC
Pre-requisite	ECES				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply mathematical modelling (differential equations, transfer functions, block-diagram and signal-flow reduction) to represent electrical and electromechanical systems.	BTL3 · Apply
CO2	Analyze time-domain behaviour — transient and steady-state response, system order, and steady-state error — of first- and second-order control systems.	BTL4 · Analyze
CO3	Analyze stability using the Routh–Hurwitz criterion and the root-locus method, and relate pole locations to performance.	BTL4 · Analyze
CO4	Analyze frequency-domain behaviour via Bode and Nyquist plots, gain and phase margins, to assess relative stability and robustness.	BTL4 · Analyze
CO5	Apply classical controller and compensator design — PID tuning, lead/lag compensation — to meet transient and steady-state specifications for an electrical loop.	BTL3 · Apply
CO6	Apply state-space representation and an introduction to digital control to model and control modern electrical/energy systems, including a simulated case study.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PSO1
CO1	H	H	–	H
CO2	H	H	–	H
CO3	H	H	–	H
CO4	H	H	–	H
CO5	H	–	H	H
CO6	H	–	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**System Modelling and Transfer Functions** Open-loop vs closed-loop control; the feedback principle and why electrical systems need it.; Modelling electrical and mechanical elements; analogies between domains.; The Laplace transform as the control engineer's tool; transfer functions.; Block-diagram algebra and reduction.; Signal-flow graphs and Mason's gain formula.; Modelling a DC motor (armature + mechanical) as a transfer function.; Modelling a DC-DC converter's control-to-output transfer function (topology owned by PEWC).; Standard test inputs: impulse, step, ramp.	CO1
M2	**Time-Domain Analysis** First-order systems: time constant, step response, settling.; Second-order systems: damping ratio, natural frequency, the response families.; Transient specifications: rise time, peak time, overshoot, settling time.; Steady-state error and the system type (0, 1, 2); error constants.; Effect of adding poles and zeros on the response.; Dominant-pole approximation.; Time response of a motor speed loop.; Reading and interpreting simulated step responses.	CO2
M3	**Stability — Routh–Hurwitz and Root Locus** The concept of stability; bounded-input bounded-output.; Characteristic equation and pole locations.; Routh–Hurwitz criterion; special cases (zero rows, zero in first column).; Root-locus construction rules.; Effect of gain on closed-loop pole movement.; Designing gain from the root locus to meet a damping spec.; Stability margins from the root locus.; Root-locus of a motor position loop.	CO3

	Title & Topics	CO
M4	**Frequency-Domain Analysis** Frequency response and the sinusoidal transfer function.; Bode plots: magnitude and phase, asymptotic construction.; Gain margin and phase margin from Bode plots.; The Nyquist stability criterion and encirclements.; Relative stability and robustness interpretation.; Bandwidth and its link to speed of response.; Correlation between time- and frequency-domain specs.; Frequency analysis of a converter voltage loop.	CO4
M5	**Controller and Compensator Design** The PID controller: proportional, integral, derivative actions and their effects.; PID tuning: Ziegler–Nichols and frequency-response-based tuning.; Practical PID: anti-windup, derivative filtering, set-point weighting.; Lead compensator design (phase margin improvement).; Lag compensator design (steady-state error improvement).; Lead-lag compensation.; Designing a PI current/voltage controller for a converter loop.; Verifying the design against transient and steady-state specs.	CO5
M6	**State-Space and Digital Control** State-space representation: state, input, output equations.; State-space from transfer functions and vice versa.; Controllability and observability at concept level.; State-feedback and pole placement (introduction).; Sampling and the z-transform; discrete transfer functions.; Digital PID and discretisation of controllers.; Introduction to modern control in energy systems (grid-following/grid-forming control concept).; A simulated capstone control loop tying the methods together.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Modern Control Engineering**	Katsuhiko Ogata	Pearson	2010 (5th Ed.)
2	**Control Systems Engineering**	Norman S. Nise	Wiley	2019 (8th Ed.)
3	**Automatic Control Systems**	Farid Golnaraghi & Benjamin C. Kuo	Wiley	2017 (10th Ed.)
4	**Feedback Control of Dynamic Systems**	Gene F. Franklin, J. David Powell & Abbas Emami-Naeini	Pearson	2019 (8th Ed.)
5	**Digital Control of Dynamic Systems**	Franklin, Powell & Workman	Addison-Wesley	1997 (3rd Ed.)

Data Structures & Algorithms – 1

DSA1 REG · 26SC1203 PBL · 26SC1203E

L-T-P-S	2-0-0-4 / 3-0-0-6	Credits	3	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S02 · T02	Mode	Regular & PBL	Category	ESC
Pre-requisite	PSPP				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyse the time and space complexity of algorithms with Big-O, and adopt the cost mindset of moving from a brute-force solution to an efficient one.	BTL4 · Analyze
CO2	Think and solve recursively - design recursive solutions, reason about base/recursive cases and the call stack - the seed of the divide-and-conquer, backtracking and dynamic-programming paradigms.	BTL4 · Analyze
CO3	Recognise and apply the core array/string problem-solving patterns - two-pointers, sliding window and prefix sums - to turn brute-force solutions into efficient ones.	BTL6 · Create
CO4	Apply binary search - on sorted data and the powerful binary-search-on-the-answer technique - and search efficiently, reasoning carefully about correctness and boundaries.	BTL4 · Analyze
CO5	Implement and analyse sorting algorithms - elementary and efficient (mergesort, quicksort) - and recognise mergesort/quicksort as a first taste of divide and conquer.	BTL4 · Analyze
CO6	Implement and apply the linear data structures - linked lists, stacks and queues - and their patterns (monotonic stack/queue, deque), and choose the right linear structure for a problem.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO8	PO9	PO11
CO1	M	**H**	M	M	M	–	–	M
CO2	M	**H**	**H**	–	M	–	–	M
CO3	M	**H**	**H**	–	M	–	–	M
CO4	M	**H**	M	–	M	–	–	L
CO5	**H**	**H**	M	–	M	–	–	L
CO6	M	**H**	**H**	–	M	M	M	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - Complexity & Algorithmic Thinking** The north star (Outside-In framing): solving real problems efficiently - and the running problem-solving-engine project and pattern playbook.; What an algorithm is, and what makes one better than another: correctness and efficiency.; Measuring running time: counting operations; the idea of growth with input size.; Asymptotic notation: Big-O (and a note on Big-Omega/Theta); dropping constants and lower-order terms.; The common complexity classes: O(1), O(log n), O(n), O(n log n), O(n-squared), O(2^n) - and what each feels like.; Space complexity: measuring memory, and the time-space trade-off.; Best, average and worst case, and why we usually care about the worst case.; The cost mindset (the core habit): start with a brute-force solution, analyse it, then optimise - the method used throughout the spine.; Estimating feasibility: given input limits, what complexity is acceptable (the interview/contest reasoning).; Analysing simple code: loops, nested loops and basic recursion (a bridge to Module 2).	CO1

	Title & Topics	CO
M2	**Module 2 - Recursion & Recursive Thinking** Why recursion early: it is the seed of divide & conquer, backtracking and dynamic programming - the powerful paradigms of DSA-3 - so the whole spine builds on it.; What recursion is: a function defined in terms of itself; the base case and the recursive case.; The call stack: how recursion executes, and the cost in stack space.; Designing a recursive solution: identify the smaller sub-problem and combine.; Classic recursion: factorial, Fibonacci, power, sum/printing - and analysing their complexity.; Recursion on structures: arrays/strings (e.g., reverse, palindrome) - recursion as a way to traverse.; Generating possibilities (a first taste of backtracking): subsets, permutations - the seed of Module-by-module growth toward DSA-3.; Recursion vs iteration: when each is clearer or cheaper; converting between them.; The danger of naive recursion: recomputation (e.g., naive Fibonacci) - and the hint of memoisation/DP to come (a forward link to DSA-3).; Recursive thinking as a habit - the mental move that unlocks the hardest problems later.	CO2
M3	**Module 3 - Arrays, Strings & the Core Patterns** Arrays and strings as the workhorse data: storage, indexing, and the cost of operations (insert/delete/search).; Dynamic arrays: how they grow (amortised cost) - the array behind most languages' lists.; The brute-force baseline: solving array/string problems the obvious (slow) way, and analysing it - the starting point to optimise.; The two-pointer pattern: opposite-ends and same-direction pointers (e.g., pair-sum in a sorted array, reversing, partitioning) - turning O(n-squared) into O(n).; The sliding-window pattern: fixed and variable windows (e.g., max/min subarray, longest substring with a property) - the pattern for contiguous-range problems.; The prefix-sum pattern: precomputing cumulative sums for O(1) range queries (and difference arrays) - trading space for time.; Recognising which pattern a problem needs: the signals (sorted? contiguous? range query?) that point to two-pointers vs window vs prefix sums.; Combining patterns and using a hash map for counting/lookups within them (a forward link to hashing in DSA-2).; String-specific problems: searching, matching and transformation with these patterns.; Complexity of the patterns and why they win - the thought process made rigorous.	CO3
M4	**Module 4 - Searching & Binary Search** The searching problem: linear search and its O(n) cost - the baseline.; Binary search on a sorted array: the algorithm, its O(log n) cost, and why sorting enables it.; Getting binary search right: the boundary/invariant reasoning that makes it correct (the classic source of bugs).; Variants: first/last occurrence, lower/upper bound, search in a rotated array.; Binary-search-on-the-answer (the powerful pattern): when the answer is monotonic, binary-search the answer space (e.g., minimum capacity/speed/size that works) - a technique that surprises with how widely it applies.; Recognising binary-search-on-the-answer problems: the monotonic-feasibility signal.; Searching in 2-D / structured data (introduction).; Comparing search strategies and choosing by the data and the query.; Complexity of searching and the role of pre-sorting (a link to Module 5).; Search patterns as interview staples - and the thought process behind them.	CO4
M5	**Module 5 - Sorting (and a First Taste of Divide & Conquer)** Why sorting matters: it enables binary search, the two-pointer pattern, grouping and much more (the connective tissue of the course).; Elementary sorts: bubble, selection and insertion sort - how they work and their O(n-squared) cost (and when insertion sort is actually good).; Mergesort: the divide-and-conquer idea (split, sort halves, merge); its guaranteed O(n log n); stability.; Quicksort: partitioning and divide and conquer; average O(n log n) and the worst case; pivot choice.; Mergesort/quicksort as a FIRST TASTE of divide and conquer - the recursion of Module 2 applied to a paradigm (an explicit forward link to DSA-3).; The O(n log n) lower bound for comparison sorting (awareness) - why we cannot do better by comparisons alone.; Non-comparison sorts (awareness): counting/radix sort and when linear-time sorting is possible.; Stability and in-place sorting, and choosing a sort for a situation.; Using a library sort vs implementing one - and why understanding it still matters.; Sorting as the enabler of other patterns - applied in the project engine.	CO5
M6	**Module 6 - Linked Lists, Stacks & Queues (+ Their Patterns)** Linked lists: singly, doubly and circular; insertion/deletion; the trade-off vs arrays (O(1) insert vs no random access).; Linked-list patterns: the two-pointer/fast-slow technique (cycle detection, middle, n-th from end) - recursion/pointers in action.; Stacks: the LIFO structure; array and linked implementations; applications (undo, call stack, expression evaluation, balanced parentheses).; The monotonic-stack pattern: solving next-greater/next-smaller and histogram-style problems elegantly - a high-value interview pattern.; Queues: the FIFO structure; circular queue; applications (scheduling, buffering, BFS - a forward link to DSA-2 graphs).; The deque (double-ended queue) and the monotonic-deque pattern: sliding-window maximum in O(n) - tying back to Module 3.; Choosing the right linear structure: matching the operations a problem needs to the structure.; Putting it together: the linear structures and patterns in the project engine.; The synthesis: the cost mindset, recursion and the patterns as one toolkit.; The bridge to DSA-2 (for the seven continuing branches) and an honest, complete close for ME and CE: what comes next (non-linear structures) and why.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Introduction to Algorithms (CLRS)**	Cormen, Leiserson, Rivest & Stein	MIT Press	2022 (4th ed.)

#	Title	Authors	Publisher	Year
2	**Grokking Algorithms**	Aditya Bhargava	Manning	2024 (2nd ed.)
3	**Data Structures and Algorithms in Java**	Michael T. Goodrich, Roberto Tamassia & Michael H. Goldwasser	Wiley	2014 (6th ed.)
4	**Elements of Programming Interviews / Cracking the Coding Interview**	Aziz, Lee & Prakash / Gayle Laakmann McDowell	EPI / CareerCup	2018-2020
5	**Data Structures and Algorithms Made Easy**	Narasimha Karumanchi	CareerMonk	2016 (2nd ed.)

Data Structures & Algorithms – 2

DSA2 REG · 26SC1305 PBL · 26SC1305E

L-T-P-S	2-0-2-4 / 3-0-6-0	Credits	4	Contact Hrs/Wk	8 / 9
Term (Reg · PBL)	S03 · T03	Mode	Regular & PBL	Category	ESC
Pre-requisite	DSA1				

Course Outcomes (COs)

	Statement	BTL
CO1	Implement and traverse trees and binary search trees, read traversal as a recursion pattern, and explain why a hierarchy outperforms a linear structure for searching and ordered data.	BTL4 · Analyze
CO2	Implement self-balancing trees and heaps (priority queues) and apply their patterns - top-k, k-way merge, running median - explaining how balance and priority guarantee efficiency.	BTL4 · Analyze
CO3	Implement hash tables (with collision resolution) and tries, analyse their performance, and apply the dictionary patterns - frequency/grouping, the complement idea, and the LRU cache.	BTL6 · Create
CO4	Represent graphs and traverse them with BFS and DFS, and apply the graph-modelling patterns - grid-as-graph, flood-fill, multi-source BFS, cycle detection and topological ordering.	BTL4 · Analyze
CO5	Apply weighted-graph algorithms as greedy-on-graphs - shortest paths (Dijkstra) and minimum spanning trees (Prim, Kruskal with Union-Find) - the seed of the greedy paradigm.	BTL6 · Create
CO6	Choose and combine the right data structures for a problem, integrate them into a working engine, analyse its efficiency, and transition into the algorithm-design paradigms.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO8	PO9	PO11
CO1	H	H	M	–	M	–	–	L
CO2	H	H	H	–	M	–	–	L
CO3	H	H	H	–	M	–	–	M
CO4	H	H	H	M	M	–	–	–
CO5	H	H	H	M	M	–	–	–
CO6	M	H	H	–	M	M	M	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - Trees & Binary Search Trees** The seam from DSA-1: recall what the linear structures and patterns can and cannot do - searching a sorted list is still O(n) or inflexible, so we need a hierarchy (and the running engine + pattern playbook continue).; Tree terminology: root, node, edge, parent/child, leaf, height, depth, subtree.; Binary trees: structure, properties, and array vs linked representations.; Tree traversals as a recursion pattern: in-order, pre-order, post-order (recursive and iterative) and level-order (BFS on a tree) - DSA-1's recursion applied to a structure.; Binary Search Trees (BST): the ordering property; search, insert and delete operations.; BST efficiency: why operations are O(h), and the best case O(log n).; The degradation problem: how a BST can become unbalanced and degrade to O(n) - motivating balanced trees (Module 2).; Tree problem patterns: traversal-based problems (height, diameter, lowest common ancestor - awareness), and recursion on trees.; Complexity analysis of tree operations (building on DSA-1's Big-O).; Implementing and testing a BST - the searchable core of the project engine.	CO1

	Title & Topics	CO
M2	**Module 2 - Balanced Trees & Heaps (Priority Queues) + Their Patterns** The motivation: a BST can degrade - we need guaranteed efficiency (balanced trees) and efficient access to the min/max (heaps).; Self-balancing trees: the idea of balance; AVL trees - balance factor and rotations (single and double).; Awareness of other balanced trees: red-black trees and B-trees, and where they are used (databases, file systems).; Heaps: the binary heap (min-heap/max-heap) and the heap property; array representation and the index formulas.; Heap operations: insert (sift-up) and extract-min/max (sift-down), heapify, and their O(log n) / O(n) costs.; Heapsort and the priority-queue abstract data type.; The heap patterns (the thought process): top-k elements, k-way merge of sorted lists, and the running-median (two-heaps) pattern - high-value interview patterns.; Choosing between a balanced tree and a heap for a workload (ordered iteration vs min/max access).; Amortised vs worst-case thinking (a deepening of DSA-1's complexity) where relevant.; Implementing a heap/priority queue and a heap pattern - the prioritisation capability of the engine.	CO2
M3	**Module 3 - Hashing & Tries: The Dictionary Problem (+ Patterns)** The motivation: even a balanced tree is O(log n) - can we get O(1) lookup? - the dictionary problem.; Hash functions, collisions (the pigeonhole principle), and collision resolution by chaining and by open addressing (probing).; Load factor, rehashing and dynamic resizing; amortised O(1) operations.; Performance of hashing: average O(1) and the worst case, and what affects it.; The hashing patterns (the thought process): frequency counting, grouping (anagrams/buckets), the complement idea (two-sum), and de-duplication - patterns that make many problems O(n).; Tries (prefix trees) - a first-class structure: structure, insert/search, and prefix queries.; Trie applications and patterns: autocomplete, dictionary/spell-check, IP routing and prefix matching; when a trie beats a hash table for string keys.; The LRU cache pattern: combining a hash table with a doubly-linked list for O(1) get/put - the data-structures-meets-system-design pattern.; Hash maps and hash sets, and choosing hashing vs a tree vs a trie.; Implementing a hash table and a trie - the instant-lookup capability of the engine.	CO3
M4	**Module 4 - Graphs: Modelling & Traversal (+ Patterns)** The motivation: trees model hierarchies, but relationships and networks (maps, grids, social graphs, dependencies) need graphs.; Graph terminology and representations: adjacency matrix vs adjacency list and their trade-offs.; Breadth-first search (BFS) using a queue: levels and shortest unweighted paths.; Depth-first search (DFS) using recursion/stack: discovery/finish and structure.; The modelling patterns (the thought process): seeing a grid/matrix as a graph, and flood-fill (connected regions, islands) - the pattern that unlocks a whole class of problems.; Multi-source BFS and the level/wave idea (e.g., nearest-source, spreading) - a powerful BFS pattern.; Connectivity and connected components; cycle detection in directed and undirected graphs.; Topological sorting of a DAG (ordering dependencies) and detecting a cycle.; Choosing BFS vs DFS for a problem, and the complexity of representation and traversal.; Building and traversing the graph - the network capability of the engine.	CO4
M5	**Module 5 - Weighted Graphs & Greedy-on-Graphs** The motivation: real networks are weighted - we need the shortest/cheapest route and the most efficient network.; Weighted graphs and the shortest-path problem.; Dijkstra's shortest-path algorithm using a priority queue (the Module-2 heap): how it works, why it is GREEDY, and its complexity.; Negative weights and the Bellman-Ford algorithm (awareness), and where Dijkstra fails.; The minimum spanning tree (MST) problem: connecting a network at least cost.; Prim's MST (a greedy algorithm using a priority queue) and Kruskal's MST (greedy, using Union-Find).; The Union-Find (disjoint-set) structure: operations, union by rank and path compression, and its near-constant amortised cost - a structure and an amortised-analysis showcase.; Greedy-on-graphs as the GREEDY SEED: why these algorithms make a locally-best choice that is globally optimal - the intuition the greedy paradigm (DSA-3) formalises.; Applications: routing and navigation, network and infrastructure design, clustering.; Implementing shortest path and MST - the routing/optimisation capability of the engine.	CO5
M6	**Module 6 - Choosing & Combining Structures + the Bridge to the Paradigms** The synthesis: a comparative map of all the structures (from DSA-1's linear structures to trees, balanced trees, heaps, hash tables, tries and graphs) - which to use for which problem.; Choosing the right structure: matching the operations a problem needs to the structure that does them efficiently (and combining structures, e.g., the LRU cache, Dijkstra's heap).; The integrated engine (the project): the BST, heap, hash table/trie and graph algorithms working together, complexity-analysed.; Space-time trade-offs and amortised vs worst-case performance (beyond Big-O).; Completing the pattern playbook with the structure patterns (top-k, LRU, flood-fill, greedy-on-graphs).; The bridge to Data Structures & Algorithms - 3 (the computing branches): the design paradigms - divide & conquer (seeded by mergesort/quicksort in DSA-1), greedy (seeded by Dijkstra/MST here), dynamic programming (seeded by naive recursion in DSA-1) and backtracking (seeded by generating subsets/permutations in DSA-1).; The advanced/range structures and algorithms deferred to DSA-3: segment trees and Fenwick (BIT) trees for range queries, string algorithms (KMP, suffix structures, Aho-Corasick) and advanced graph algorithms (network flow, strongly connected components).; An honest, complete close for ECE/EEE/BT: how this data-structures foundation serves their systems and computational courses.; The frontier: data structures in databases, operating systems, AI and competitive programming.; The forward map and interview readiness: how to keep building on the problem judges.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Introduction to Algorithms (CLRS)**	Cormen, Leiserson, Rivest & Stein	MIT Press	2022 (4th ed.)
2	**Data Structures and Algorithms in Java**	Michael T. Goodrich, Roberto Tamassia & Michael H. Goldwasser	Wiley	2014 (6th ed.)
3	**Algorithms**	Robert Sedgewick & Kevin Wayne	Addison-Wesley	2011 (4th ed.)
4	**Elements of Programming Interviews / Cracking the Coding Interview**	Aziz, Lee & Prakash / Gayle Laakmann McDowell	EPI / CareerCup	2018-2020
5	**Data Structures and Algorithms Made Easy**	Narasimha Karumanchi	CareerMonk	2016 (2nd ed.)

Deep Learning with Accelerated Computing

DLRN 26AD2304

L-T-P-S	2-0-0-4	Credits	3	Contact Hrs/Wk	6
Semester	S05	Mode	Regular	Category	PCC
Pre-requisite	MLRN				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply neural-network foundations — perceptrons, multi-layer networks, backpropagation, common activation functions, weight initialization, regularization (dropout, batch norm) — to train classification and regression models.	BTL3 · Apply
CO2	Apply convolutional neural networks — convolution, pooling, the canonical architectures (LeNet, AlexNet, VGG, ResNet at intuition), transfer learning — to image-classification problems.	BTL3 · Apply
CO3	Apply sequence models — RNN, LSTM, GRU at intuition, the transformer attention mechanism, encoder/decoder architectures — to sequence-modeling tasks.	BTL3 · Apply
CO4	Apply GPU-accelerated training — CUDA at intuition, PyTorch GPU programming, mixed-precision training, gradient accumulation, distributed training basics — to scale model training beyond single-GPU memory.	BTL3 · Apply
CO5	Apply inference optimization — quantization (INT8, FP16), pruning at intuition, ONNX as the cross-framework format, TensorRT / Triton serving — to deploy models for production latency.	BTL3 · Apply
CO6	Analyze the modern model-development lifecycle — experiment tracking with Weights and Biases / MLflow, the API-vs-self-hosted decision, foundation models as starting points, the path to LLM fine-tuning depth (NLPL).	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	H	–	–	H
CO2	H	–	H	–	H
CO3	H	–	H	–	H
CO4	H	–	H	H	H
CO5	H	–	H	H	H
CO6	H	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Neural Network Foundations** The perceptron — linear classifier, decision boundary, the limitation (XOR problem); the historical context.; Multi-layer perceptrons — composition of linear-and-nonlinear layers; the universal approximation theorem at intuition.; Backpropagation — chain rule applied to compute gradients of loss w.r.t. all weights; the canonical hand-derivation on a 2-layer network.; Activation functions — sigmoid, tanh, ReLU, GELU, Swish; the canonical choice (ReLU) and when alternatives win.; Loss functions — MSE for regression, cross-entropy for classification; the canonical pairings.; Optimization revisited — SGD, momentum, Adam (from MFAI but applied here); learning-rate schedules.; Regularization — dropout (intuition), batch normalization, weight decay; the bias-variance lens.; A first lab: in PyTorch, train a small MLP on MNIST classification; vary depth, width, dropout rate; observe the train/val curves.	CO1

	Title & Topics	CO
M2	**Convolutional Neural Networks** Why convolution — translation invariance, parameter sharing, local connectivity; the canonical motivation for image data.; Convolution operation — kernels, stride, padding; the connection to image processing classical filters.; Pooling — max pooling vs average pooling; downsampling and the receptive-field expansion.; CNN architectures — LeNet (1989), AlexNet (2012), VGG (2014), ResNet (2015); the pattern of "deeper is better with skip connections".; Transfer learning — using pre-trained ImageNet weights, fine-tuning vs feature-extraction; the canonical PyTorch hub pattern.; Modern vision — Vision Transformers (ViT), CLIP, foundation models for vision; honest framing on when CNN still wins.; A lab: fine-tune a pre-trained ResNet-50 on CIFAR-10; compare against training from scratch; observe the convergence and accuracy difference.	CO2
M3	**Sequence Models and Transformers** Recurrent neural networks (RNN) — the canonical motivation for sequence data; vanilla RNN, vanishing gradients.; LSTM and GRU at intuition — gated alternatives; the canonical use cases (language modeling, time-series).; Attention mechanism — query, key, value; the canonical formula softmax(QK^T/sqrt(d))V (revisited from MFAI).; Self-attention vs cross-attention — the computational structure; the connection to multi-head attention.; The transformer architecture — encoder + decoder; positional embeddings; layer normalization.; Why transformers won — parallelization across sequence positions, scalability to hundreds of millions / billions of parameters.; Foundation models at intuition — GPT, BERT, T5; the pre-train-and-fine-tune paradigm; the bridge to NLPL/GAPE.; A lab: in PyTorch, implement a small transformer from scratch (3-layer, 8-head) for a toy translation task; verify against torch.nn.Transformer.	CO3
M4	**GPU-Accelerated Training** Why GPUs for DL — massive parallelism, matrix-multiply throughput; the SIMT model from COAA recap.; CUDA at intuition — kernels, threads, blocks, grids; the canonical "Hello CUDA" example; depth in DLAC PBL.; PyTorch GPU programming — model.cuda(), tensor.to(device); the canonical patterns and the canonical pitfalls (data on wrong device).; Mixed-precision training — FP16 forward, FP32 master weights; PyTorch torch.cuda.amp; the canonical 2-3x speedup on H100/A100.; Gradient accumulation — simulating larger batch sizes when GPU memory is the limit.; Distributed training basics — data parallel via PyTorch DDP; the cross-GPU all-reduce pattern; the bridge to multi-node training (depth in NLPL).; Profiling — torch.profiler, nvtop, the canonical bottleneck-finding workflow.; A lab: train a CNN on CIFAR-10 once on CPU, once on GPU; measure the speedup; enable mixed precision; measure again.	CO4
M5	**Inference Optimization and Deployment** The training-vs-inference asymmetry — training cares about throughput, inference cares about latency.; Quantization — INT8 and FP16 quantization; post-training quantization vs quantization-aware training; the canonical 2-4x speedup.; Pruning at intuition — removing low-importance weights; structured vs unstructured pruning.; Knowledge distillation at intuition — student model learning from teacher; the canonical ~10x parameter reduction.; ONNX — Open Neural Network Exchange; the canonical cross-framework deployment format.; TensorRT and Triton — NVIDIA's production-grade inference stack; the canonical deployment for low-latency requirements.; Serving frameworks — TorchServe, TF Serving, vLLM (LLM-specific); the contemporary 2026 landscape.; A lab: take a trained ResNet-50; convert to ONNX; deploy via Triton in a Docker container; measure latency vs PyTorch eager mode.	CO5
M6	**Modern Model Development Lifecycle** Experiment tracking — Weights and Biases (W&B), MLflow, Neptune; the canonical "version control for ML experiments" stack.; Hyperparameter optimization — Bayesian optimization, Optuna, the canonical W&B sweeps integration.; Foundation models as starting points — Hugging Face Hub as the model-zoo; the from-scratch-vs-fine-tune-vs-API decision.; API-vs-self-hosted decision — when GPT-4 / Claude / Gemini APIs win; when self-hosted Llama / Qwen / Mistral wins; the cost / latency / privacy / customization matrix.; The path forward to LLM depth — NLPL covers from-scratch transformer training and vLLM serving; GAPE covers the full GenAI ecosystem at survey depth; AAAE covers agents.; India's AI compute landscape — Yotta NM1, AI4Bharat compute, IndiaAI mission GPU procurement; the implications for Indian AI engineers.; AI-augmented ML development — Claude / Copilot / Cursor for ML code; the discipline of running every generated training loop through profiling.; A final-week lab: train a small image classifier on a real Indian dataset (e.g., Indian crop disease, Indian sign language); track in W&B; quantize for deployment; deploy a demo via Streamlit / Gradio.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Deep Learning**	Ian Goodfellow, Yoshua Bengio & Aaron Courville	MIT Press	2016
2	**Dive into Deep Learning**	Aston Zhang, Zachary C. Lipton, Mu Li & Alexander J. Smola	Cambridge University Press	2023
3	**Deep Learning with PyTorch**	Eli Stevens, Luca Antiga & Thomas Viehmann	Manning	2020

#	Title	Authors	Publisher	Year
4	**Neural Networks and Deep Learning — A Textbook**	Charu C. Aggarwal	Springer	2023 (2nd Ed.)
5	**Programming Massively Parallel Processors — A Hands-on Approach**	David B. Kirk & Wen-mei W. Hwu	Morgan Kaufmann	2022 (4th Ed.)
6	**Designing Machine Learning Systems — An Iterative Process for Production-Ready Applications**	Chip Huyen	O'Reilly	2022
7	**Efficient Deep Learning — Fast and Accurate Inference**	Gaurav Menghani	Manning	2024

Deep Learning with Accelerated Computing

DLRN 26AD2304E

L-T-P-S	3-0-0-6	Credits	3	Contact Hrs/Wk	9
Trimester	T07	Mode	PBL (Project-Based)	Category	PCC
Pre-requisite	MLRN				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze a deep-learning system as a layered system — served model, inference-optimisation pipeline, trained checkpoints, architecture, tensor primitives, GPU hardware — and trace one inference request from API to GPU kernel.	BTL4 · Analyze
CO2	Apply inference-optimisation techniques — post-training quantisation (INT8, FP16), structured and unstructured pruning, knowledge distillation, ONNX export, TensorRT compilation, batching strategies — to deploy models within latency and cost budgets.	BTL3 · Apply
CO3	Apply deep-learning training practices — PyTorch autograd, modern optimisers (AdamW, Lion), learning-rate scheduling, mixed precision (AMP), gradient accumulation, distributed data parallel (DDP), checkpointing — to train large models reliably across multiple GPUs.	BTL3 · Apply
CO4	Analyze deep-learning architectures — convolutional networks (ResNet, EfficientNet), transformer blocks at depth (attention math, residual connections, layer-norm placement), and the modern hybrids (Vision Transformers, ConvNeXt) — and choose architectures based on task and compute budget.	BTL4 · Analyze
CO5	Apply tensor-primitive fluency — PyTorch tensor operations, broadcasting, contiguity, in-place vs functional ops, autograd computation graph, profiling with torch.profiler — to write efficient deep-learning code.	BTL3 · Apply
CO6	Apply GPU and accelerated-computing concepts — CUDA programming model (threads, blocks, grids), GPU memory hierarchy (registers, shared, global), the principles of high-performance kernels, occupancy reasoning — to understand and influence what runs on the GPU.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	H	H	–	H
CO2	H	–	H	H	H
CO3	H	–	H	–	H
CO4	H	H	–	–	H
CO5	H	–	–	H	H
CO6	H	H	–	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The Deep Learning System as a Whole** A working deep-learning system on day one — students send images of Indian traffic to VisionPort and see object-detection results in real time before any internals.; The canonical deep-learning system stack: client → API gateway → Triton Inference Server → optimised model (TensorRT engine) → GPU → response.; The training-vs-inference split: orders-of-magnitude differences in compute, memory, latency, and the optimisations each side does.; The data-architecture-training-evaluation-deployment lifecycle revisited from MLNG, now at deep-learning scale.; A reading of the VisionPort codebase: data pipeline, training loop, architecture file, evaluation harness, optimisation script, Triton config.; GPU economics: H100 vs A100 vs L40S vs consumer 4090; per-hour cost; throughput-per-dollar reasoning.; Tracing one inference request end-to-end: HTTP arrival → image preprocessing → tensor batching → TensorRT engine → GPU kernel → postprocessing → response with bounding boxes.; What this course owns vs what comes downstream: deep-learning depth + GPU programming here; LLM engineering at depth in NLPL; embedded AI in EATM (same trimester, different hardware envelope).	CO1

	Title & Topics	CO
M2	**Inference Optimisation** The inference-cost equation: latency, throughput, GPU memory, power; the engineering job of pushing all four favourably.; Quantisation: FP32 → FP16 → INT8 → INT4; the accuracy-vs-cost trade-off; per-tensor vs per-channel quantisation.; Post-training quantisation (PTQ): calibration data, the ranges-and-zero-points workflow.; Quantisation-aware training (QAT): straight-through estimator, fake quantisation; when QAT recovers accuracy lost by PTQ.; Pruning: structured (remove channels, heads), unstructured (zero individual weights); the sparsity hardware actually exploits.; Knowledge distillation: teacher-student training; soft-label loss; how distillation compresses a 7B-parameter model into a usable 1B.; Compilation pipelines: PyTorch model → ONNX → TensorRT engine; kernel fusion; the speedup magic.; Serving infrastructure: Triton Inference Server, ONNX Runtime, vLLM (preview, deeper coverage in NLPL); batching dynamics (static vs dynamic vs continuous).	CO2
M3	**Training Deep Networks** PyTorch autograd at depth: the dynamic computation graph, the backward pass mechanics, custom autograd functions.; Modern optimisers: SGD with momentum revisited, AdamW (the deep-learning workhorse), Lion as the recent simpler alternative; the choice criteria.; Learning-rate scheduling: warmup, cosine annealing, one-cycle; the engineering of the most-tuned hyperparameter.; Mixed-precision training (AMP): FP16/BF16 with FP32 master weights; the loss-scaling trick that prevents underflow.; Gradient accumulation: simulating a larger batch size when memory is the constraint; the discipline of effective batch.; Gradient clipping: by norm, by value; preventing the explosive-gradient blow-up.; Distributed Data Parallel (DDP): each GPU holds a replica, gradients all-reduce; why DDP beats DataParallel for almost all use cases.; Checkpointing: full vs sharded; resuming training; the discipline of "always be able to resume" for long jobs.	CO3
M4	**Architectures** Convolutional networks revisited: stride, padding, dilation, receptive field; why CNNs dominate vision.; ResNet at depth: skip connections, the gradient-flow argument; the depth-vs-trainability win.; EfficientNet and the compound scaling law: balancing width, depth, resolution.; Object detection architectures: two-stage (Faster R-CNN at intuition level), one-stage (YOLO family at depth, RetinaNet); the speed-accuracy frontier.; The transformer block at depth: attention math fully derived (Q, K, V matrices, scaled dot-product, multi-head), positional encoding (sinusoidal vs learned vs RoPE), residual + layer-norm placement (pre-norm vs post-norm).; Vision Transformers (ViT): the image-as-tokens trick; patch embedding; why ViTs beat CNNs at scale.; Modern hybrids: ConvNeXt (CNNs designed with transformer lessons), Swin Transformer (hierarchical attention); the architecture race.; Choosing an architecture by task and budget: object detection on edge vs in the cloud; image classification at low data; the cookbook practitioners use.	CO4
M5	**Tensor Primitives and PyTorch** Tensors revisited from CFAI/NumPy: PyTorch tensors with device placement (cpu vs cuda) and gradient tracking.; Broadcasting rules: the alignment-from-the-right rule; the silent-bug class of unintended broadcasting.; Contiguity: row-major storage in PyTorch; .contiguous() and .reshape() vs .view(); when an operation forces a copy.; In-place vs functional operations: torch.add vs torch.add_; the autograd-safety implications.; Strides at intuition level: why transpose is free, why reshape sometimes is not.; The autograd computation graph: forward saves activations, backward consumes them; gradient checkpointing as the memory-vs-compute trade.; Mixed precision in practice: torch.cuda.amp.autocast and GradScaler; the patterns that just work and the ones that surprise.; Profiling: torch.profiler with TensorBoard; reading a PyTorch profile to find the bottleneck (CPU-side, GPU-side, kernel launches).	CO5
M6	**GPU and Accelerated Computing** GPU vs CPU: thousands of simple cores vs few complex cores; the SIMT execution model.; CUDA programming model: threads → warps (32) → blocks → grids; the hierarchy that maps onto hardware.; GPU memory hierarchy: registers (per-thread), shared memory (per-block), global memory (per-device), the L1/L2 caches; the latency-and-bandwidth ladder.; A first CUDA kernel: vector add written by hand; understanding thread-index arithmetic.; A worked GEMM kernel walkthrough: the naïve version (memory-bound), the tiled version (uses shared memory), the cuBLAS version (the gap to the hand-rolled tiled).; Occupancy and warp efficiency: divergent branches kill performance; coalesced memory access matters above all.; GPU profiling: NVIDIA Nsight Compute for kernel-level analysis; identifying memory vs compute boundedness.; A glance at the modern accelerated-computing landscape: TPU, AMD MI300, Cerebras, Groq, Tenstorrent; what each is good for; why CUDA still wins for general ML.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Deep Learning**	Ian Goodfellow, Yoshua Bengio & Aaron Courville	MIT Press	2016
2	**Dive into Deep Learning**	Aston Zhang, Zachary C. Lipton, Mu Li & Alexander J. Smola	Cambridge University Press	2023

#	Title	Authors	Publisher	Year
3	**Programming Massively Parallel Processors — A Hands-on Approach**	David B. Kirk & Wen-mei W. Hwu	Morgan Kaufmann	2022 (4th Ed.)
4	**CUDA by Example — An Introduction to General-Purpose GPU Programming**	Jason Sanders & Edward Kandrot	Addison-Wesley	2010
5	**Deep Learning with PyTorch**	Eli Stevens, Luca Antiga & Thomas Viehmann	Manning	2020
6	**Computer Vision — Algorithms and Applications**	Richard Szeliski	Springer	2022 (2nd Ed.)
7	**Efficient Deep Learning — Fast and Accurate Inference**	Gaurav Menghani	Manning	2024

Design Thinking for Innovation & Entrepreneurship

DTIE REG · 26UC1137 PBL · 26UC1137E

L-T-P-S	0-0-4-0 / 3-0-0-0	Credits	2	Contact Hrs/Wk	4 / 3
Term (Reg · PBL)	S01 · T01	Mode	Regular & PBL	Category	HSS
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply the design-thinking mindset and human-centred discovery to empathise with users and frame a sharp, validated problem worth solving.	BTL3 · Apply
CO2	Generate, structure and select innovative ideas using creativity techniques and innovation frameworks, moving from a framed problem to a compelling concept.	BTL6 · Create
CO3	Prototype concepts rapidly across fidelities using modern tools - 3D printing, AR/VR, AI-assisted tools and Figma/UI prototyping - choosing the right prototype for the question being asked.	BTL6 · Create
CO4	Apply user-experience and usability principles to design and test a product experience, iterating from user feedback toward a market-ready solution (Deliver).	BTL4 · Analyze
CO5	Model a venture using entrepreneurship-lifecycle frameworks and navigate the deep-tech startup journey from idea to a fundable, go-to-market-ready business.	BTL4 · Analyze
CO6	Conduct market study and analytics to validate demand and size an opportunity, and craft and deliver compelling stories and investor-grade pitches that sell the product and the team.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO2	PO3	PO6	PO8	PO9	PO10	PO11
CO1	M	M	M	**H**	**H**	M	M
CO2	**H**	**H**	M	**H**	**H**	–	M
CO3	M	**H**	–	**H**	**H**	–	M
CO4	M	**H**	–	**H**	**H**	–	M
CO5	M	**H**	L	**H**	**H**	–	M
CO6	M	M	M	**H**	**H**	**H**	**H**

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - Design Thinking Foundations & Empathy (Discover + Define)** What design thinking is and why it matters: human-centred, iterative, bias-to-action innovation; the innovator's mindset (curiosity, empathy, comfort with ambiguity, optimism).; The design-thinking process models: the d.school five stages (empathise, define, ideate, prototype, test) and the Double Diamond (discover, define, develop, deliver).; Divergent vs convergent thinking and the rhythm of opening up and narrowing down across the diamonds.; Empathy and user research methods: observation, contextual inquiry, user interviews, and the principles of asking non-leading questions ('the mom test' idea).; Sense-making tools: empathy maps, personas, customer journey maps, and 'how-might-we' questions.; Needs vs solutions vs features; latent vs stated needs; the Jobs-To-Be-Done lens (what 'job' a user hires a product to do).; Problem framing and definition: point-of-view statements, problem statements, and reframing a problem to widen or sharpen it.; Identifying assumptions and the riskiest unknowns; the idea of validating a problem before building a solution.; Stakeholders, context and constraints; designing for inclusion and accessibility; ethics of user research and consent.; Framing a deep-tech opportunity: recognising problems where an engineering/technology advantage creates defensible value.	CO1

	Title & Topics	CO
M2	**Module 2 - Ideation, Creativity & Innovation (Develop)** The psychology of creativity: fluency, flexibility, originality; overcoming fixation and the fear of bad ideas; deferring judgement.; Structured ideation techniques: brainstorming and brainwriting, SCAMPER, mind-mapping, analogous inspiration, 'worst possible idea', and Crazy-8s.; Lateral thinking and provocation; combining and remixing ideas; building on others' ideas ('yes, and').; Types and dimensions of innovation: incremental vs radical vs disruptive; sustaining vs disruptive innovation; product, process, service and business-model innovation.; Innovation frameworks: the innovation funnel, the Three Horizons, and where deep-tech innovation sits (technology push vs market pull).; Idea selection and convergence: clustering/affinity mapping, dot-voting, and criteria-based screening (desirability-feasibility-viability).; Evaluating ideas against the desirability (do people want it?), feasibility (can it be built?), viability (can it sustain a business?) lenses.; Concept development: turning a selected idea into a concept statement and a value hypothesis.; The role of generative AI as an ideation partner: using AI tools to expand, challenge and refine ideas responsibly (concept, not implementation).; From concept to testable hypothesis: defining what must be true for the idea to succeed and what to prototype first.	CO2
M3	**Module 3 - Prototyping & Modern Prototyping Technologies (Develop)** What a prototype is and is not: a prototype is a question made tangible; prototyping to learn vs prototyping to sell.; Fidelity spectrum: paper and low-fidelity prototypes, wireframes, mockups, interactive prototypes, and high-fidelity functional prototypes - and when each is appropriate.; The Minimum Viable Product (MVP) concept and the smallest experiment that answers the riskiest question.; Rapid physical prototyping with 3D printing: the workflow concept (model → slice → print), what additive manufacturing is good/poor at, design-for-printing basics, and when a physical prototype is worth it (relevant to deep-tech/hardware ventures).; Immersive prototyping with AR/VR: when an augmented or virtual experience is the right prototype (spatial products, training, visualisation), and the concept of prototyping an experience rather than an object.; AI-assisted prototyping: using generative AI for rapid concept art, copy, mockups, synthetic user data and even draft UI - and the limits and ethics of AI-generated artefacts.; Digital UI prototyping with Figma and similar tools: frames, components, auto-layout, and clickable interactive prototypes as the dominant way to prototype software products (concepts and workflow, not deep tooling).; Storyboards and 'Wizard-of-Oz' prototypes: faking the back-end to test the front experience.; Choosing the right prototype: matching the prototype type and fidelity to the learning goal, the audience, time and cost.; Prototyping for deep-tech ventures: de-risking a technology claim with a focused proof-of-concept while using cheap prototypes for the experience around it.	CO3
M4	**Module 4 - UX Design, Usability & Testing (Deliver)** What UX is (and how it differs from UI): the total experience vs the visual/interactive surface.; Foundational UX principles: Norman's design principles (affordances, signifiers, feedback, mapping, constraints) and the gulf of execution/evaluation.; Usability heuristics (Nielsen's 10) and the idea of heuristic evaluation.; Interaction and information design basics: hierarchy, flow, navigation, and reducing cognitive load ('don't make me think').; Visual design fundamentals for non-designers: layout, alignment, contrast, typography, colour and consistency; design systems and reusable components (concept).; Accessibility and inclusive design: designing for diverse abilities and contexts (concept and why it matters legally and ethically).; Usability testing: planning a test, recruiting a handful of users, think-aloud protocol, observing rather than leading, and why five users find most issues.; Capturing and prioritising findings: severity rating, and turning observations into design changes.; The iteration loop: test → learn → refine; how UX evidence drives prototype changes toward 'market-ready'.; Defining 'market-ready' for an experience: quality bar, edge cases, onboarding and first-use experience.	CO4
M5	**Module 5 - Entrepreneurship Lifecycle & Deep-Tech Ventures** The entrepreneurship lifecycle: ideation, validation, early traction, growth, scale, and exit - and what matters at each stage.; Business model design: the Business Model Canvas and the Lean Canvas; mapping customers, value, channels, revenue and costs.; The Value Proposition Canvas: fit between customer jobs/pains/gains and the product's pain-relievers/gain-creators.; Lean startup thinking: build-measure-learn, validated learning, pivots, and avoiding building things nobody wants.; Deep-tech ventures specifically: long R&D horizons, technology and IP as the moat, higher capital needs, and the lab-to-market journey.; Intellectual property basics for founders: patents, trademarks, copyright and trade secrets (awareness, not law); why IP matters for deep tech.; Funding and the ecosystem: bootstrapping, grants, incubators/accelerators, angels and venture capital; the Indian deep-tech ecosystem (incubators, government innovation missions) at an awareness level.; Unit economics and viability basics: revenue models, cost structure, the concept of a path to profitability (no heavy finance - that is for management electives).; Go-to-market basics: the first customers, channels, positioning and the idea of product-market fit.; Team, execution and founder mindset: roles, equity awareness, resilience, and ethical/responsible entrepreneurship.	CO5

	Title & Topics	CO
M6	**Module 6 - Market Study, Analytics & Storytelling / Pitching** Market research fundamentals: primary vs secondary research; qualitative vs quantitative; designing surveys and interviews that avoid bias.; Sizing the opportunity: TAM-SAM-SOM, bottom-up vs top-down estimation, and reasoning transparently about assumptions.; Competitive and industry analysis: competitor mapping, positioning maps, and frameworks (e.g., SWOT, Porter's five forces) at an applied level.; Customer validation and traction metrics: interviews, landing-page/sign-up tests, and the basic metrics that show demand (awareness level).; Reading and presenting data: turning survey/market data into clear, honest charts and insights; basic analytics literacy and avoiding misleading visuals.; The science and craft of storytelling: narrative structure, the hero's-journey framing with the customer as hero, and 'made to stick' principles (simple, unexpected, concrete, credible, emotional, story).; Story for products and ventures: crafting the problem-solution-impact narrative and the 'why now / why us' arc.; Pitch construction: the canonical pitch-deck structure (problem, solution, market, product, business model, traction, team, ask) and tailoring it to the audience (customer vs investor vs jury).; Presentation and delivery skills for pitching: structure, pacing, visual slide design (signal over noise), handling Q&A, and persuasive, confident delivery.; Pitching formats: the elevator pitch, the demo, the investor pitch and the competition pitch - and selling yourself and your team, not just the product.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Change by Design**	Tim Brown	Harper Business	2019 (revised ed.)
2	**The Design of Everyday Things**	Don Norman	Basic Books	2013 (revised ed.)
3	**Sprint: How to Solve Big Problems and Test New Ideas in Just Five Days**	Jake Knapp, John Zeratsky & Braden Kowitz	Simon & Schuster	2016
4	**The Lean Startup**	Eric Ries	Crown Business	2011
5	**Business Model Generation / Value Proposition Design**	Alexander Osterwalder & Yves Pigneur (with Greg Bernarda & Alan Smith)	Wiley	2010 / 2014
6	**Resonate / slide:ology**	Nancy Duarte	Wiley / O'Reilly	2010 / 2008
7	**The Mom Test**	Rob Fitzpatrick	Founder Centric	2013

Digital Design & Computer Architecture

DDCA REG · 26EC1202 PBL · 26EC1202E

L-T-P-S	2-0-0-4 / 3-0-0-6	Credits	3	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S01 · T02	Mode	Regular & PBL	Category	ESC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Operate and explain a complete processor as a whole system - its instruction set (RISC-V), the fetch-decode-execute cycle and the datapath as connected black boxes - and read its instructions and data in binary/two's complement.	BTL3 · Apply
CO2	Open the first layer - explain and build the single-cycle datapath and the control unit that decodes each instruction and steers the datapath - replacing the provided black boxes while the CPU keeps running.	BTL6 · Create
CO3	Open the storage layer - explain and build the register file and the instruction/data memory the datapath uses, with addressing - replacing the provided black boxes while the CPU keeps running.	BTL6 · Create
CO4	Open the state-and-sequencing layer - design and build the sequential logic (latches, flip-flops, registers, counters and finite state machines) the storage and control are made of - replacing the provided black boxes while the CPU keeps running.	BTL6 · Create
CO5	Open the computation layer - design and build the combinational logic and the Arithmetic Logic Unit (adders, multiplexers, decoders, the ALU) - replacing the provided black box while the CPU keeps running.	BTL6 · Create
CO6	Open the bedrock - logic gates, Boolean algebra and the bit - so nothing is a black box; then make the machine wholly your own by building and EXTENDING it (a new instruction and a memory-mapped peripheral) and reason about performance and modern architecture.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO8	PO9	PO11
CO1	**H**	M	M	L	M	–	–	L
CO2	**H**	**H**	**H**	L	M	–	–	L
CO3	**H**	**H**	**H**	M	M	–	–	L
CO4	**H**	**H**	M	M	M	–	–	L
CO5	**H**	**H**	**H**	M	M	–	–	M
CO6	**H**	M	**H**	M	**H**	M	M	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - The Whole Computer: the Instruction Set, Programs, the Fetch-Decode-Execute Cycle & the Datapath as Black Boxes** Outside-In day one: assemble a COMPLETE working RISC-V CPU from provided black-box blocks and run a real program on it - the whole machine you will open, one layer at a time, until nothing is a black box.; What a computer is, at the top level: the stored-program (von Neumann/Harvard) model; instructions and data living in memory; hardware vs software.; The Instruction Set Architecture (ISA) as the contract between hardware and software; why RISC-V (open, clean, modern - industry, India's microprocessor mission, research).; A bounded RISC-V subset (about ten instructions): arithmetic/logic (add, sub, and, or, slt), an immediate (addi), memory (lw, sw), a branch (beq) and a jump (jal) - enough to write real programs and to be the target the whole course builds.; Reading the machine - the data-representation language: binary and hexadecimal, and two's-complement signed numbers; how an instruction and a data word are encoded in bits (the language needed to read instructions/data now; the gate-level meaning of bits is the bedrock, Module 6).; Instruction formats: the fields (opcode, registers, immediate) and how to decode them.; RISC-V assembly (foundational): reading and writing short programs; how a simple high-level idea (a loop, an if) becomes instructions (links to the Java course); run and step them in Ripes.; The fetch-decode-execute cycle: the loop every processor runs, observed on the working CPU.; The datapath as connected BLACK BOXES: the program counter, instruction memory, register file, ALU, control unit and data memory - what each does and how they connect to execute an instruction (HOW each works is opened in the modules that follow).; Tooling & workflow orientation: the visual simulator (Logisim/Digital) with provided black-box blocks, the RISC-V visualiser (Ripes), and starting a Git repository.	CO1
M2	**Module 2 - Opening the Architecture: the Single-Cycle Datapath & the Control Unit** Opening the first black box: how the datapath we ran in Module 1 actually executes each instruction class, and how the control unit steers it.; The single-cycle datapath organisation: how the PC, instruction memory, register file, ALU and data memory are wired so one instruction completes per clock.; Instruction flow by class: how an arithmetic, a load/store and a branch/jump instruction each move data through the datapath.; The control unit: decoding the opcode (and function fields) into the control signals (register-write, ALU-operation, memory-read/write, branch, multiplexer selects) that steer the datapath.; Single-cycle control is combinational: the control unit as a truth-table / decoder mapping opcode to control signals (built from provided logic primitives now; the primitives themselves are opened later).; Multiplexers in the datapath (as black boxes for now): how control signals select between paths (e.g., ALU source, write-back source) - their internals are opened in Module 5.; Branch and jump: how the control and datapath change the program counter for control flow.; Building and REPLACING the control unit and datapath wiring with the student's own, dropping it into the working CPU and re-running the program (the functional units remain provided black boxes).; Verifying by tracing one instruction end-to-end through the datapath and naming the control signals at each step.; What remains boxed: the register file and memory (Module 3), the sequential elements inside them (Module 4), the ALU (Module 5) and the gates (Module 6).	CO2
M3	**Module 3 - Opening the Storage: Memory & the Register File** Opening the storage black boxes: the datapath reads and writes a register file and a memory - how do they work?; Memory as addressable storage: the idea of an address and the data stored at it; address and data lines (conceptual).; The register file: the CPU's small, fast register set; reading two registers and writing one per clock; how addressing selects a register (decoders/multiplexers used here as provided blocks).; Instruction memory and data memory: where the program and data live; read and write behaviour; how the datapath fetches instructions and loads/stores data.; Memory organisation: words, addresses, and how memory is laid out and accessed.; The memory hierarchy and WHY it exists (foundational): the speed-vs-size-vs-cost trade-off - registers, cache, main memory, storage; the idea of locality (cache depth is later).; The von Neumann bottleneck (awareness): why memory access often limits performance.; Building and REPLACING the register file and the memories with the student's own, dropping them into the CPU and re-running the program (the registers/flip-flops inside them remain provided black boxes).; Verifying the storage: read/write tests and watching the program's data move through the built register file and memory.; What remains boxed: the sequential elements (registers, flip-flops) inside the storage (Module 4), the ALU (Module 5) and the gates (Module 6).	CO3

	Title & Topics	CO
M4	**Module 4 - Opening State & Sequencing: Sequential Logic** Opening the sequential black boxes: the registers, the program counter and the step-by-step behaviour we relied on are built from state elements - how do they hold and sequence values?; Why memory and feedback: the limit of combinational logic and the need for circuits that hold state.; Latches and flip-flops: the SR latch, the D latch and the D flip-flop; level-sensitive vs edge-triggered behaviour (concrete, simulated).; The clock and synchronous design: why a clock and edge-triggering make complex digital systems reliable; setup/hold time (foundational).; Registers: a row of flip-flops storing a multi-bit value - the storage element inside the register file (Module 3).; Counters: synchronous counters; the program counter as a counter that steps through instructions (the PC you met in the datapath).; Finite State Machines (FSMs): states, transitions, state diagrams and tables - the technique behind sequential control.; Shift registers (brief) and their uses.; Building and REPLACING the registers, the program counter and any state elements with the student's own (built from flip-flops; the flip-flops use provided gate primitives), dropping them into the CPU and re-running the program.; What remains boxed: the ALU's combinational logic (Module 5) and the gates that everything reduces to (Module 6).	CO4
M5	**Module 5 - Opening Computation: Combinational Logic & the ALU** Opening the compute black box: the ALU that does the datapath's arithmetic and logic is built from combinational logic - how?; Combinational logic: output depends only on current inputs (no memory) - the defining idea.; Logic simplification with Karnaugh maps (K-maps): fewer gates means faster, cheaper, lower-power hardware.; Arithmetic circuits: the half adder and full adder; the ripple-carry adder; carry propagation and why wide adders are slower.; Subtraction with two's complement: the adder/subtractor with a control input (using the two's-complement language from Module 1).; Data-routing blocks opened: the multiplexer (the datapath's selectors, Module 2) and the decoder (the register file's addressing, Module 3) - now built, not boxed.; Comparators and other standard combinational blocks (for the branch condition).; The Arithmetic Logic Unit (ALU): combining add, subtract, AND, OR and set-less-than into one block with an operation-select input - the datapath's calculator.; Building and REPLACING the ALU (and the datapath's MUXes/decoders) with the student's own, built from provided gate primitives, dropping it into the CPU and re-running the program.; What remains boxed: only the logic gates themselves - the bedrock opened in Module 6.	CO5
M6	**Module 6 - The Bedrock & the Machine Made Wholly Yours: Gates, Bits, Build, Extend & the Frontier** Opening the last black box: the gate. Logic gates (AND, OR, NOT, NAND, NOR, XOR) - truth tables, symbols and behaviour - the primitive every block above was built from.; Boolean algebra: variables, operations and key laws (building on the propositional logic from Mathematics for Computation, T01); from a truth table to a gate circuit (sum-of-products).; Universality of NAND/NOR: any logic can be built from one gate type - so the entire machine reduces to one primitive; building a gate from NAND.; The bit at the gate level: how a logic gate physically realises 0/1 (logic levels), and a brief awareness of the transistor beneath the gate (depth is CMOS VLSI's).; Closing the decomposition: with gates open, the whole CPU is now transparent top to bottom - the student has opened and rebuilt every layer.; The machine is wholly yours - BUILD-complete: the entire single-cycle CPU now runs on the student's own blocks at every layer.; EXTEND it (build additional designs with RISC-V): add a NEW instruction (e.g., XOR / a custom op) - the change to the ALU and the control unit, on a template.; Extend it with the outside world: add a simple MEMORY-MAPPED OUTPUT peripheral (drive LEDs / a display) so the CPU affects something - the seed of embedded design.; Performance and modern architecture (awareness): clock period and CPI; why single-cycle is limited; pipelining, caching and parallelism - the direction of modern computing.; From your CPU to the frontier and the forward map: how this foundation scales to embedded/IoT microcontrollers, VLSI chips and edge-AI accelerators, and how the later courses build on exactly this.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Digital Design and Computer Architecture: RISC-V Edition**	Sarah L. Harris & David Harris	Morgan Kaufmann	2021
2	**Computer Organization and Design: RISC-V Edition**	David A. Patterson & John L. Hennessy	Morgan Kaufmann	2020 (2nd ed.)
3	**Digital Design**	M. Morris Mano & Michael D. Ciletti	Pearson	2018 (6th ed.)
4	**Digital Fundamentals**	Thomas L. Floyd	Pearson	2014 (11th ed.)
5	**The RISC-V Reader: An Open Architecture Atlas**	David Patterson & Andrew Waterman	Strawberry Canyon	2017
6	**The Elements of Computing Systems (Nand2Tetris)**	Noam Nisan & Shimon Schocken	MIT Press	2021 (2nd ed.)

Ecology & Environment

ECEN REG · 26UC0009 PBL · 26UC0009E

L-T-P-S	2-0-0-0 / 3-0-0-0	Credits	0	Contact Hrs/Wk	2 / 3
Term (Reg · PBL)	3S · 3S	Mode	Regular & PBL	Category	AUC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand ecosystem structure, function, and ecological balance.	BTL2 · Understand
CO2	Understand biodiversity, its value, and conservation approaches.	BTL2 · Understand
CO3	Apply principles of sustainable natural-resource use.	BTL3 · Apply
CO4	Analyze environmental pollution — sources, effects, and control.	BTL4 · Analyze
CO5	Analyze climate change and global environmental issues.	BTL4 · Analyze
CO6	Analyze environmental management, law, and AI/IoT-enabled monitoring.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO5	PO6	PO7	PSO1	PSO2
CO1	–	–	M	**H**	M	–
CO2	–	–	M	**H**	M	–
CO3	–	–	**H**	**H**	M	–
CO4	M	–	–	**H**	–	M
CO5	–	–	**H**	**H**	–	M
CO6	–	M	–	**H**	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Ecosystems & Ecology** Ecosystem structure and function.; Energy flow and trophic levels.; Food chains, webs, and ecological pyramids.; Biogeochemical cycles (C, N, P, water).; Ecological succession.; Population and community ecology.; Major biomes and Indian ecosystems.; Ecological balance and stability. **Labs:** Field/observation survey of a local ecosystem and its food web.; Analyse sample ecological data (species counts) to assess ecosystem health. **Case Study:** Case Study — a degraded vs restored ecosystem (e.g., a revived lake/wetland in India): analyse how ecological structure broke down and how restoration brought it back.	CO1
M2	**Biodiversity & Conservation** Levels of biodiversity.; India as a megadiverse nation.; Biodiversity hotspots (Western Ghats, Himalaya).; Threats — habitat loss, invasive species, poaching.; IUCN status and endangered species.; In-situ conservation (parks, reserves).; Ex-situ conservation.; Ecosystem services and their value. **Labs:** Biodiversity survey of a campus/local area with species logging.; Map threats and design a conservation plan for a local species/habitat. **Case Study:** Case Study — a conservation success/failure in India (e.g., Project Tiger / a vanishing species): analyse the threats, the intervention, and the outcome.	CO2

	Title & Topics	CO
M3	**Natural Resources & Sustainability** Renewable vs non-renewable resources.; Water resources and scarcity.; Forest and land resources.; Energy resources and the transition.; Mineral and food resources.; Resource depletion and over-exploitation.; Carrying capacity and ecological footprint.; Sustainable resource management. **Labs:** Calculate your ecological/water footprint and propose reductions.; Audit a resource (water/energy) use on campus and recommend savings. **Case Study:** Case Study — a water/resource crisis in India (e.g., a city's 'day zero' or groundwater depletion): analyse the causes and sustainable solutions.	CO3
M4	**Environmental Pollution** Air pollution — sources and effects.; Water pollution and quality.; Soil contamination.; Noise pollution.; Solid-waste management.; E-waste and plastic pollution.; Pollution and human health.; Pollution-control measures and technology. **Labs:** Measure/interpret a local pollution indicator (air AQI / water quality).; Design a waste-segregation or pollution-reduction plan for a setting. **Case Study:** Case Study — Delhi's air pollution / a polluted river (Ganga/Yamuna): analyse the sources, health impacts, and the policy + tech responses.	CO4
M5	**Climate Change & Global Issues** Greenhouse effect and global warming.; Causes and evidence of climate change.; Impacts — sea level, extremes, agriculture.; Ozone depletion.; Acid rain and trans-boundary issues.; International agreements (Kyoto, Paris).; India's climate commitments (NDCs, net-zero 2070).; Mitigation and adaptation. **Labs:** Analyse climate/temperature trend data for a region.; Build a carbon-footprint estimate and a mitigation plan for a unit. **Case Study:** Case Study — a climate impact in India (e.g., extreme heat, floods, or a coastal threat): analyse the climate linkage and the adaptation/mitigation response.	CO5
M6	**Environmental Management, Law & AI** Environmental Impact Assessment (EIA).; Environmental laws of India (Air, Water, EP Acts).; Pollution Control Boards (CPCB/SPCB).; Green technology and clean production.; Circular economy.; AI/IoT for environmental monitoring.; Remote sensing for environment.; Environmental ethics and the engineer's role. **Labs:** Conduct a mini-EIA for a hypothetical project.; Design an AI/IoT-based environmental-monitoring concept for a problem. **Case Study:** Case Study — AI/IoT environmental monitoring (e.g., sensor-based air-quality or satellite forest monitoring): analyse how technology improves environmental management.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Textbook of Environmental Studies for Undergraduate Courses, 2nd Ed.**	Erach Bharucha	Universities Press (UGC)	2013
2	**Environmental Studies: From Crisis to Cure, 4th Ed.**	R. Rajagopalan	Oxford University Press	2016
3	**Environmental Science: Earth as a Living Planet, 9th Ed.**	Daniel B. Botkin & Edward A. Keller	Wiley	2014
4	**Ecology and Environment, 12th Ed.**	P. D. Sharma	Rastogi Publications	2017
5	**Environmental Science: A Global Concern, 14th Ed.**	William P. Cunningham & Mary Ann Cunningham	McGraw-Hill	2017
6	**Silent Spring**	Rachel Carson	Houghton Mifflin	2002
7	**The Sixth Extinction: An Unnatural History**	Elizabeth Kolbert	Henry Holt	2014

Electric Circuits & Energy System Fundamentals

ECES REG · 26EE1201 PBL · 26EE1201E

L-T-P-S	2-0-2-4 / 3-0-6-0	Credits	4	Contact Hrs/Wk	8 / 9
Term (Reg · PBL)	S02 · T02	Mode	Regular & PBL	Category	PCC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Describe the electrical energy system as a whole - power flow from generation to load, and the electrical quantities and AC/DC forms involved - and analyse simple circuits with the fundamental laws (Ohm, Kirchhoff) that read it.	BTL3 · Apply
CO2	Open the network layer - analyse DC circuits systematically with node and mesh methods and the network theorems (superposition, Thevenin, Norton, maximum power transfer, star-delta).	BTL4 · Analyze
CO3	Open the AC behaviour - analyse sinusoidal-steady-state circuits with phasors and impedance, and compute AC power (real, reactive, apparent), power factor and resonance - the language of the energy system.	BTL6 · Create
CO4	Open the three-phase and transient layers - analyse balanced three-phase systems (the form power is generated and transmitted in) and the transient response of first- and second-order circuits to switching.	BTL4 · Analyze
CO5	Open the magnetics - analyse magnetic circuits and electromagnetic induction, and explain and analyse the transformer, the device that makes the high-voltage grid possible.	BTL4 · Analyze
CO6	Reach the electromagnetic bedrock and the energy-system fundamentals - generation, transmission, distribution and the grid - integrate the complete energy-system analysis, and map the frontier (renewables, smart grid, EVs) and the path forward.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO11
CO1	**H**	**H**	M	L	M	–	–	L
CO2	**H**	**H**	M	M	M	–	–	L
CO3	**H**	**H**	M	M	M	–	–	L
CO4	**H**	**H**	M	M	M	M	–	L
CO5	**H**	**H**	M	M	M	–	–	M
CO6	**H**	M	**H**	M	M	M	L	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - The Whole Electrical Energy System & the Laws to Read It** Outside-In day one: build and MEASURE a small, complete electrical energy system - a source, a transmission-line model, a transformer, distribution and loads - and watch power flow from generation to load; the working system you will open, layer by layer.; The electrical energy system as a whole: generation, transmission, distribution and utilisation; the system as connected black boxes (source, line, transformer, loads).; Electrical quantities: charge, current, voltage (potential difference), electrical power and energy; units and conventions.; Direct current (DC) vs alternating current (AC): why the grid is AC; an overview of single-phase and three-phase at the system level.; Circuit elements: the resistor, capacitor and inductor and their v-i relationships; ideal voltage and current sources; the load.; Ohm's law and power dissipation; energy and the kilowatt-hour (the unit you are billed in).; Kirchhoff's laws (KVL and KCL): the conservation laws that govern every circuit.; Series and parallel combinations; voltage and current dividers - the everyday tools.; Reading the system with these laws: applying them to the loads and the line of the energy system you built.; Tooling & workflow: the bench (supply, multimeter, oscilloscope, wattmeter), the simulator (SPICE / Scilab-Xcos), Python for calculations, and starting the project log.	CO1

	Title & Topics	CO
M2	**Module 2 - Opening the Network: DC Circuit Analysis & Theorems** Opening the network: the loads and lines of the energy system are circuits - here is how to analyse them systematically.; Node-voltage analysis: a systematic method for any circuit; choosing the reference node and writing the equations.; Mesh-current analysis: the dual method; choosing mesh currents and writing the equations.; Source transformation and combining sources - simplifying before solving.; Superposition: analysing one source at a time in a linear circuit and adding the results.; Thevenin's theorem: reducing any linear network seen by a load to a single source and resistance - the engineer's workhorse.; Norton's theorem: the current-source dual; converting between Thevenin and Norton.; Maximum power transfer: when a load draws the most power, and why this matters for matching.; Star-delta (Y-Delta) transformation: simplifying networks that are neither series nor parallel (and a first link to three-phase).; Applying these methods to the energy system's loads and source - the analytical 'opening' of the network.	CO2
M3	**Module 3 - Opening the AC Behaviour: Sinusoidal Steady-State & AC Power** Opening the AC behaviour: the grid is AC - here is how AC circuits behave and how power flows in them.; Sinusoids: amplitude, frequency, phase; RMS (effective) value and why it matters for power.; Phasors: representing sinusoids as complex numbers; phasor arithmetic (building on complex numbers from mathematics).; Impedance and admittance of R, L and C; the phasor (frequency-domain) circuit and analysing it with the Module-2 methods.; AC steady-state analysis of R-L-C circuits; the impedance triangle.; AC power: instantaneous power; real (active) power, reactive power and apparent power; the power triangle; the power-factor angle.; Power factor: what it means physically, why a poor (lagging) power factor wastes capacity and costs money, and POWER-FACTOR CORRECTION (adding capacitance) - the single most important practical AC-power concept.; Complex power and the conservation of complex power.; Resonance: series and parallel RLC resonance; resonant frequency, quality factor Q and bandwidth - the basis of tuning and filtering.; Measuring it: real/reactive power and power factor on the bench (the wattmeter), and correcting the power factor of a real load.	CO3
M4	**Module 4 - Opening Three-Phase & the Transient: Three-Phase Systems & Transient Response** Opening three-phase: power is generated, transmitted and largely consumed as three-phase - here is why and how.; Why three-phase: constant power delivery, efficient transmission, and the rotating field (the seed of motors, Electric Motors & Drives).; Balanced three-phase sources and the phase sequence; star (Y) and delta (Delta) connections.; Line and phase voltages and currents in star and delta; the root-three relationships.; Three-phase power: real, reactive and apparent power in balanced systems, and how it is measured (the two-wattmeter method - awareness).; Unbalanced three-phase (awareness) and why balance matters.; Transient response: why circuits do not change instantly; the meaning of a transient.; First-order transients: the RC and RL circuit - natural and step (forced) response, the time constant, charging/discharging (switching behaviour).; Second-order transients: the RLC circuit - the natural response and the over-, critically- and under-damped cases (oscillation and ringing).; Why transients matter for energy systems: switching, energisation and the seed of fault/protection behaviour (Power System Protection, later).	CO4
M5	**Module 5 - Opening the Magnetics: Magnetic Circuits, Induction & the Transformer** Opening the magnetics: the transformer in the system steps voltage up and down - how, and on what physics?; Magnetic fields and magnetic circuits: magnetomotive force (MMF), flux, reluctance and the magnetic-circuit analogy to electric circuits; the B-H curve and saturation (intuition).; Electromagnetic induction: Faraday's law and Lenz's law - the basis of every transformer and machine.; Self-inductance and mutual inductance; coupled coils.; The transformer - the device that makes transmission possible: the ideal transformer (turns ratio, voltage/current transformation, impedance reflection).; The real transformer: the equivalent circuit (winding resistance, leakage and magnetising reactance, core loss).; Transformer performance: voltage regulation and efficiency; losses (copper and iron).; Transformer types (awareness): autotransformer and the three-phase transformer; where each is used in the grid.; The principle of electromechanical energy conversion (awareness): how the same induction principle underlies rotating machines (the bridge to Electric Motors & Drives).; Analysing and measuring a transformer: turns ratio, regulation and efficiency on the bench - the magnetics layer of the energy system.	CO5

	Title & Topics	CO
M6	**Module 6 - The Bedrock & the Whole Energy System: Generation, T&D, the Grid & the Frontier** Reaching the bedrock: the charge-and-field physics beneath circuits and the energy system (intuition; deep field theory is the Electromagnetic Fields science elective).; Generation: how electrical energy is produced - thermal, hydro, nuclear, and the renewables (solar PV, wind) - at a fundamentals level.; Transmission and distribution: why high voltage for transmission (loss reduction), the role of transformers, and the step down to distribution and the consumer (overview; detailed G/T&D is Electric Power G/T/D, T05).; The single-line (one-line) diagram: how power engineers draw the whole system; the per-unit idea (introduction).; Power quality and power factor at system scale: why utilities care about reactive power and power factor across the grid.; The integrated analysis (capstone): analysing the whole energy system end-to-end - source, line, transformer, distribution and loads - in DC/AC/three-phase terms, with power-factor and efficiency, and measuring it.; The frontier - the modern energy transition: renewable integration and the National Green Hydrogen Mission; the smart grid; electric vehicles and energy storage.; AP/Telangana and national context: the grid, renewable targets and the energy-engineering opportunity.; Reflection: an energy system understood from the whole down to the physics - the cornerstone of the EEE journey.; The forward map: how this foundation feeds machines, power systems, power electronics, control and the smart-grid/renewable/EV courses.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Fundamentals of Electric Circuits**	Charles K. Alexander & Matthew N. O. Sadiku	McGraw-Hill	2021 (7th ed.)
2	**Engineering Circuit Analysis**	William H. Hayt, Jack E. Kemmerly & Steven M. Durbin	McGraw-Hill	2018 (9th ed.)
3	**Introductory Circuit Analysis**	Robert L. Boylestad	Pearson	2015 (13th ed.)
4	**Electric Machinery Fundamentals**	Stephen J. Chapman	McGraw-Hill	2012 (5th ed.)
5	**Electrical Machines, Drives, and Power Systems**	Theodore Wildi	Pearson	2013 (6th ed.)
6	**Basic Electrical Engineering / Power System Engineering**	D. P. Kothari & I. J. Nagrath	McGraw-Hill (India)	2019

Electric Motors, Drives & Electromechanical Systems

EMDE REG · 26EE2203 PBL · 26EE2203E

L-T-P-S	2-0-0-4 / 3-0-6-0	Credits	3 / 4	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S04 · T05	Mode	Regular & PBL	Category	PCC
Pre-requisite	ECES				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply transformer fundamentals — ideal vs real transformer, equivalent circuit, OC / SC tests, voltage regulation, efficiency, three-phase connections (Y-Y, Y-Δ, Δ-Y, Δ-Δ) and vector groups — to design and analysis problems on single-phase and three-phase transformers including auto-transformers and instrument transformers.	BTL3 · Apply
CO2	Apply DC-machine fundamentals — construction, EMF equation, types (separately-excited, shunt, series, compound), torque–speed characteristics, starting methods, armature-voltage and field-weakening speed control — to canonical traction, manufacturing-drive and high-precision-control problems.	BTL3 · Apply
CO3	Apply three-phase induction-motor analysis — construction, rotating magnetic field, slip, equivalent circuit, torque–slip characteristic, starting methods (DOL, star–delta, autotransformer, soft-starter, VFD) and IS / IEC IE2 / IE3 / IE4 / IE5 energy-efficiency standards — to industrial-motor sizing and selection problems.	BTL3 · Apply
CO4	Apply synchronous-machine fundamentals — construction (salient-pole vs cylindrical-rotor), EMF generation, armature reaction, equivalent circuit, voltage regulation (synchronous-impedance, MMF, ZPF methods), parallel operation, V- and inverted-V-curves — to generator-and-motor analysis in power-plant and industrial-drive contexts.	BTL3 · Apply
CO5	Analyse special electric machines — BLDC motor, PMSM, switched-reluctance motor, stepper motor, universal motor and linear motor — their construction, operating principles, characteristic curves and canonical EV / automation / aerospace applications, with attention to the rare-earth supply-chain context.	BTL4 · Analyze
CO6	Analyse modern motor-drive systems — EV traction-motor design, regenerative-braking strategies, sensorless vector control, motor-drive simulation tools (MATLAB Simscape Electrical, PSIM, PLECS), the AI-augmented motor-control workflow, and the Indian EV-and-drives industry ecosystem.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO7	PO11	PSO1
CO1	**H**	**H**	**H**	M	–	–	**H**
CO2	**H**	**H**	**H**	–	–	–	**H**
CO3	**H**	**H**	**H**	M	M	–	**H**
CO4	**H**	**H**	**H**	–	–	–	**H**
CO5	**H**	**H**	**H**	**H**	M	–	**H**
CO6	**H**	**H**	**H**	**H**	M	M	**H**

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Transformers — Single-Phase, Three-Phase, Special-Purpose** Ideal transformer — turns ratio, voltage transformation, current transformation; the canonical $V_1/V_2 = N_1/N_2 = I_2/I_1$ set.; Real transformer — winding resistances, leakage reactance, magnetising branch; the canonical T-equivalent circuit and approximate equivalent circuit.; Open-circuit and short-circuit tests — the canonical lab procedure to extract equivalent-circuit parameters; canonical IS 2026 test norms.; Voltage regulation — definition, all-day efficiency, condition for maximum efficiency; the canonical ½×full-load result.; Three-phase transformers — Y-Y, Y-Δ, Δ-Y, Δ-Δ connections; vector groups (Yd11, Dyn11); the canonical 30° phase-shift and "don't parallel different vector groups" rule.; Auto-transformer — the canonical "copper-saving" advantage; canonical starting-of-induction-motor application; safety implications.; Instrument transformers — current transformer (CT), potential transformer (PT); the canonical metering and protection applications; canonical Indian standards (IS 2705, IS 3156).; A first lab — perform OC and SC tests on a 1 kVA single-phase lab transformer; extract equivalent-circuit parameters; predict full-load voltage regulation; verify by direct measurement.	CO1
M2	**DC Machines — Construction, EMF, Characteristics & Control** DC machine construction — stator (field winding), rotor (armature), commutator and brushes; the canonical mechanical-to-electrical (generator) and electrical-to-mechanical (motor) conversion.; EMF equation E = (Φ Z N P) / (60 A); the canonical generator-action equation derivation.; Types of DC machines — separately-excited, shunt, series, compound (cumulative / differential, long-shunt / short-shunt); the canonical application map for each.; Torque equation T = (Φ Z I_a P) / (2π A); speed–torque characteristics for each type; the canonical T-N curves.; Starting methods — 3-point starter, 4-point starter; the canonical reason DC motors require external starters (high starting current at low back-EMF).; Speed-control methods — armature-voltage control (Ward-Leonard system), field-weakening control; canonical historical applications.; DC machine in 2026 context — declining new applications (BLDC has replaced DC in most consumer / EV / appliance uses); still dominant in Indian Railways legacy locomotives and high-precision research applications.; Lab — characterise a small DC shunt motor; measure speed–torque curves at different field currents; verify against textbook predictions.	CO2
M3	**Three-Phase Induction Motors** Construction — stator (3-phase winding), rotor (squirrel-cage vs wound-rotor / slip-ring); the canonical "workhorse of industrial motors" (~70% of industrial electrical-energy consumption).; Rotating magnetic field — the canonical three-phase-currents-produce-rotating-field result; synchronous speed N_s = 120f/P.; Slip and rotor frequency — s = (N_s − N_r)/N_s; canonical relation to rotor frequency f_r = s·f.; Equivalent circuit — referred to stator; the canonical T-equivalent; per-phase-impedance approach.; Torque-slip characteristic — starting torque, maximum (pull-out) torque, full-load torque; the canonical complete T-s curve.; Starting methods — DOL (Direct On-Line), star-delta starter, autotransformer starter, soft-starter, VFD; the canonical industrial-selection map.; Speed-control methods — pole-changing (consequential-pole, Dahlander), rotor-resistance (slip-ring motors only), V/f control via VFD; the canonical modern dominance of VFD-based control.; Energy-efficiency standards — IE1 / IE2 / IE3 / IE4 / IE5 per IS 12615 / IEC 60034-30-1; canonical Indian BEE 5-star rating; the canonical 5-star IE3+ induction motor as the 2026 industrial standard.; Lab — perform no-load and blocked-rotor tests on a 3-phase induction motor; extract equivalent-circuit parameters; predict efficiency at full load; verify against measured values.	CO3
M4	**Synchronous Machines — Generators & Motors** Construction — salient-pole (low-speed hydro / diesel generators) vs cylindrical / non-salient-pole (high-speed thermal / nuclear / gas-turbine generators); the canonical N_s = 120f/P relationship.; EMF generation — the canonical phasor diagram approach; armature winding factors (distribution factor, pitch factor).; Armature reaction — the canonical demagnetising and cross-magnetising effects under leading vs lagging vs unity power-factor loads.; Equivalent circuit — synchronous reactance X_s, the canonical voltage-vector approach.; Voltage regulation — synchronous-impedance method (EMF method), MMF method, ZPF method; the canonical "why three methods" trade-off.; Parallel operation of synchronous generators — the canonical synchronisation conditions (voltage match, frequency match, phase sequence, phase angle); the canonical synchroscope and dark / bright-lamp methods; load-sharing under varying excitation and prime-mover input.; V-curves and inverted-V-curves — the canonical reactive-power-control characteristic; applications in synchronous condenser for power-factor correction at substations.; Synchronous motor — canonical constant-speed-drive and power-factor-correction applications; starting methods (damper winding, pony motor, slip-frequency starting).; Lab — synchronise a small lab synchronous generator to the lab grid; observe synchronisation conditions; operate at unity, leading and lagging power factor; sketch V-curve.	CO4

	Title & Topics	CO
M5	**Special Electric Machines** BLDC motor (Brushless DC) — permanent-magnet rotor with electronic commutation; the canonical EV-and-appliance traction motor; Hall-sensor-based 6-step commutation.; PMSM (Permanent-Magnet Synchronous Motor) — sinusoidal back-EMF; the canonical high-performance EV-drive choice (Tesla Model 3, Ola Electric S1 Pro, Ather 450X, TVS iQube).; Switched-reluctance motor (SRM) — no permanent magnets, no rotor winding; canonical applications in industrial drives and rare-earth-free EV concepts.; Stepper motor — open-loop position control; the canonical 3D-printer / CNC-machine / robotics-actuator use cases; full-step / half-step / micro-step.; Universal motor — operates on both AC and DC; the canonical kitchen-appliance and power-tool motor.; Linear motors — linear-induction and linear-synchronous; canonical Maglev train and high-speed industrial-conveyor applications; the Shanghai Maglev and SR Magneglide / IIT-M LIM research benchmarks.; Modern motor-design context — the rare-earth supply-chain concern (China-dominated NdFeB market); India's response — rare-earth-mineral diversification, ferrite-magnet PMSM design, SRM for rare-earth-free EVs.; Lab — characterise a small BLDC motor; measure back-EMF waveform; implement Hall-sensor-based 6-step commutation on an Arduino / STM32; observe step response.	CO5
M6	**Modern Motor-Drive Systems & EV Traction** EV traction-motor system architecture — the canonical PMSM / BLDC + battery + inverter + motor-control unit (MCU); canonical Tesla / Ola Electric / Tata EV / Mahindra Last Mile Mobility design choices.; Field-Oriented Control (FOC) / Vector Control — the canonical d-q decoupling enabling DC-motor-like control of PMSM and induction motors; canonical industrial-VFD and EV-MCU embedded firmware.; Regenerative braking — the canonical bidirectional inverter operation; regen-vs-friction-brake blending logic; canonical Indian-EV implementation context.; Sensorless motor control — eliminating Hall sensors / encoders via back-EMF observation (Luenberger observer, sliding-mode observer, MRAS); the canonical cost-reduction and reliability-improvement technique.; Modern simulation tools — MATLAB Simscape Electrical (Simulink-native), PSIM (specialised for power electronics + motor drives), PLECS (precision drives); the canonical industrial-tooling map.; Microcontroller and DSP platforms for motor control — STM32 G-series / F-series, TI C2000 (Piccolo / Delfino), NXP S32K; the canonical EV-grade automotive MCU choice.; AI-augmented motor control — neural-network-based speed estimation, ML-tuned PI gains, model-predictive control (MPC); the canonical 2025+ research direction; LLM coding assistants (Claude / GPT-4) for motor-control firmware boilerplate.; Indian motor-and-drives industry context — Crompton Greaves, Bharat Bijlee, Kirloskar Electric, Siemens-Crompton JV (legacy industrial); Ola Electric, Ather, Tata Motors EV, Mahindra (EV); the canonical 2025-2030 EV-driven manufacturing capacity expansion.; Final-week lab — implement V/f open-loop control of an induction motor on STM32 / TI C2000 motor-control kit; demonstrate speed-control with safety-aware ramp-up and protection logic.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Electric Machinery**	A. E. Fitzgerald, Charles Kingsley Jr. & Stephen D. Umans	McGraw-Hill	2014 (7th Ed.)
2	**Electric Machinery Fundamentals**	Stephen J. Chapman	McGraw-Hill Education	2020 (6th Ed.)
3	**Electric Drives**	Ned Mohan	Wiley	2014 (3rd Ed.)
4	**Electric Motor Drives — Modeling, Analysis, and Control**	R. Krishnan	Prentice Hall	2001
5	**Permanent Magnet Synchronous and Brushless DC Motor Drives**	R. Krishnan	CRC Press	2009
6	**Modern Power Electronics and AC Drives**	Bimal K. Bose	Prentice Hall	2001
7	**Electrical Machinery**	P. S. Bimbhra	Khanna Publishers	2018 (7th Ed.)

Electric Power Generation, Transmission and Distribution

EGTD REG · 26EE2204 PBL · 26EE2204E

L-T-P-S	3-1-0-0 / 3-3-0-0	Credits	4	Contact Hrs/Wk	4 / 6
Term (Reg · PBL)	S04 · T05	Mode	Regular & PBL	Category	PCC
Pre-requisite	ECES				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply generation-technology fundamentals — thermal, hydro, nuclear, solar PV, wind, pumped-storage and battery storage — to compare the operating principle, cost structure and emission profile of each, with specific reference to the canonical Indian generation mix and the 2030 / 2050 net-zero pathway.	BTL3 · Apply
CO2	Apply power-plant economics — load curves, load-duration curve, load factor, plant capacity factor, plant utilisation factor, base-load vs peaking-plant economics, two- / three-part tariffs, LCOE — to compute and benchmark generation cost in the Indian DISCOM / UDAY context.	BTL3 · Apply
CO3	Apply transmission-line analysis — short, medium and long line models, π-model and T-model, ABCD parameters, voltage regulation, line efficiency — to predict the steady-state performance of an Indian-grid transmission corridor.	BTL3 · Apply
CO4	Apply transmission-system engineering — overhead-line conductor selection (ACSR, AAAC, ACCC, HTLS), tower types, insulator design, sag and tension calculation, corona, underground cables, HVDC, GIS — to design problems on Indian 765 / 400 / 220 / 132 kV systems.	BTL3 · Apply
CO5	Analyse distribution-system design — radial, ring-main and mesh topologies, distribution-transformer sizing, voltage regulation, on-load tap-changing, power-factor improvement, AT&C losses, AMI — in the Indian-DISCOM context (UDAY, RDSS, Saubhagya, Smart Meter National Programme).	BTL4 · Analyze
CO6	Analyse modern smart-grid systems — SCADA, IEC 61850 substation automation, WAMS / PMU, microgrids, grid-connected renewables, AI-augmented power-system operation — in the Indian National Smart Grid Mission (NSGM) rollout context.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO7	PO11	PSO1
CO1	H	H	H	–	–	H	–	H
CO2	H	H	M	–	–	–	M	H
CO3	H	H	H	–	M	–	–	H
CO4	H	H	H	M	–	–	–	H
CO5	H	H	H	–	–	–	M	H
CO6	H	H	H	–	H	H	–	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Generation Technologies & the Indian Mix** Thermal power plants — coal-fired (the dominant Indian generation, ~50% of capacity in 2026), gas-fired, oil-fired; the canonical Rankine cycle; supercritical and ultra-supercritical units.; Indian coal context — Coal India Limited (CIL), the canonical lignite / bituminous / sub-bituminous mix; the canonical 2050 net-zero pathway and CCUS (carbon capture, utilisation and storage) intuition.; Hydroelectric power plants — high-head (Pelton, 250+ m), medium-head (Francis, 30-300 m), low-head (Kaplan, < 30 m); canonical Indian hydro context (NHPC, Bhakra, Tehri, Sardar Sarovar).; Nuclear power plants — the canonical PHWR (Pressurised Heavy-Water Reactor) used by NPCIL (Tarapur, Kalpakkam, Kakrapar); India's three-stage program (PHWR → FBR → Thorium); the upcoming Bhavini PFBR.; Solar power plants — PV (utility-scale crystalline-silicon, rooftop) vs CSP (concentrated solar power); canonical Indian grid-scale projects (Bhadla Solar Park 2.25 GW, Pavagada 2.05 GW, Kurnool 1 GW); inverter, MPPT and string-vs-central architecture.; Wind power plants — onshore (Tamil Nadu, Karnataka, Gujarat clusters) vs offshore (early-stage Gujarat / Tamil Nadu MNRE pilots); canonical Suzlon, Vestas, Inox Wind, ReGen Powertech market context.; Storage technologies — pumped hydro (Tehri, Srisailam, Nagarjunasagar reversible units), battery storage (Li-ion grid-scale BESS), hydrogen at intuition; the canonical 2026 grid-scale storage role under renewable-integration push.; Lab — compare 2026 LCOE of solar PV vs coal vs gas vs nuclear in the Indian context using current MNRE / CERC / IEA data; reflect on India's 500 GW non-fossil 2030 target.	CO1
M2	**Power-Plant Economics** Load curves — daily, monthly, annual; the canonical analysis of demand patterns; the canonical Indian-summer-evening peak phenomenon.; Load-duration curve — hourly demand sorted in descending order; the canonical visualisation for capacity planning and energy-mix optimisation.; Load factor, plant capacity factor, plant utilisation factor — definitions, typical Indian-utility values, the canonical performance-metric set.; Base-load vs intermediate vs peaking plants — the canonical economic-dispatch logic; merit order; canonical Indian base-load (NTPC coal, NPCIL nuclear) and peak-load (gas, pumped-hydro) generation mix.; Cost of generation — fixed costs (capital, O&M, insurance, taxes), variable costs (fuel, ash handling, water); the canonical Levelised Cost of Energy (LCOE) formula.; Tariff structures — two-part tariff, three-part tariff, time-of-day (ToD) tariff; the canonical Indian electricity-tariff design across DISCOMs; cross-subsidisation in the Indian context.; India's power-sector financial context — DISCOM aggregate technical and commercial (AT&C) losses (~15-20% in 2026), UDAY scheme, Late Payment Surcharge Rules 2022, Revamped Distribution Sector Scheme (RDSS) 2021-2026.; First lab — given a 24-hour load profile for an Indian DISCOM area, compute load factor and design the optimal generation mix (base-load + intermediate + peaking) for minimum cost; verify with the canonical Indian-grid context.	CO2
M3	**Transmission-Line Modelling** Transmission-line motivation — connecting generators to loads at high voltage to reduce I^2R losses; the canonical Indian-grid hierarchy 765 / 400 / 220 / 132 / 66 / 33 / 11 kV / 415 V.; Line parameters — resistance R, inductance L, capacitance C per unit length; how they're computed for typical conductor configurations (single, bundled, double-circuit, three-phase).; Short transmission line (< 80 km) — only series impedance; the canonical equivalent circuit; voltage regulation by phasor approximation.; Medium transmission line (80-250 km) — nominal π-model and T-model with lumped shunt admittance; the canonical analysis approach.; Long transmission line (> 250 km) — distributed-parameter model; hyperbolic-function (sinh, cosh) ABCD parameters; characteristic impedance Z_c and propagation constant γ.; Voltage regulation and transmission efficiency — the canonical figures-of-merit; the canonical "sending-end vs receiving-end" power-circle diagrams.; Modern HVDC at intuition — high-voltage DC for very long distances and asynchronous interconnections; canonical Indian HVDC links (Champa-Kurukshetra ±800 kV, NER-NR HVDC, Vindhyachal back-to-back).; Lab — in MATLAB / PSCAD, simulate a 220 kV transmission line of 200 km using the π-model; sweep load conditions; observe voltage regulation; compare to actual Indian-grid line parameters.	CO3
M4	**Transmission-System Engineering** Overhead transmission lines — conductor types (ACSR, AAAC, ACCC, HTLS), tower types (lattice / monopole / guyed), insulator types (porcelain, glass, polymer composite); canonical PowerGrid Corporation / state-transco design conventions.; Sag and tension calculation — the canonical parabolic and catenary approximations; the canonical effect of temperature, wind, ice loading per IS 802.; Corona effect — disruptive critical voltage, visual critical voltage; canonical mitigation via bundle conductors (twin / triple / quad bundle); 765 kV and 1100 kV experimental UHV-AC context.; Underground cables — the canonical XLPE construction; oil-filled cables; canonical urban high-voltage applications in Mumbai, Delhi, Bengaluru CBD.; High-voltage AC transmission hierarchy — 765 / 400 / 220 kV inter-state corridors; canonical National Grid backbone; Inter-Regional Energy Transfer Capacity (IRETC) metrics.; HVDC transmission — long-distance bulk-power transfer, asynchronous-grid interconnection; canonical Indian HVDC links (Champa-Kurukshetra 6000 MW, Vindhyachal back-to-back, Rihand-Delhi); LCC vs VSC HVDC.; Gas-Insulated Switchgear (GIS) at intuition — for urban substations and brownfield expansion; the canonical compact-substation technology; canonical applications in Delhi Metro substations and high-density CBD areas.; Lab — design a 220 kV transmission line for a given corridor — select conductor, compute sag for different conditions, estimate corona losses; deliver a design report.	CO4

	Title & Topics	CO
M5	**Distribution-System Design** Distribution voltage levels — primary distribution (33 / 11 kV), secondary (415 / 230 V); the canonical Indian-DISCOM hierarchy; rural vs urban.; Distribution topologies — radial (cheap, low-reliability), ring-main (medium-reliability), interconnected / mesh (high-reliability, expensive); the canonical urban-vs-rural choices.; Distribution-transformer sizing — load forecasting, diversity factor, coincidence factor; the canonical sizing methodology and IS 1180 / IS 2026 transformer-loading norms.; Voltage regulation in distribution — line voltage drop, on-load tap-changer (OLTC) action, off-circuit tap changer; the canonical voltage-band requirement per IS standards.; Power-factor improvement — capacitor banks at substation / feeder / consumer end; the canonical industrial-customer requirement (kVAh billing vs kWh billing); SVCs and STATCOMs at intuition.; Aggregate Technical and Commercial (AT&C) losses — the canonical Indian-DISCOM concern; technical (line and transformer losses) vs commercial (theft, billing inefficiency, non-payment) breakdown; canonical 2026 reduction targets per RDSS.; Smart distribution — Advanced Metering Infrastructure (AMI), Distribution Management Systems (DMS), Outage Management Systems (OMS); the canonical Smart Meter National Programme rollout and RDSS Phase-I / II milestones.; Lab — design an LT distribution network for a residential colony of 100 houses; size the distribution transformer; select conductor; verify voltage-drop compliance; deliver design report.	CO5
M6	**Modern Smart-Grid Systems** SCADA architecture — Remote Terminal Units (RTUs) at substations, master stations at Load Despatch Centres; the canonical Indian POSOCO / NLDC / RLDC / SLDC hierarchy.; IEC 61850 substation automation — the canonical modern protocol; GOOSE (Generic Object Oriented Substation Event) messaging; Sampled Values (SV); MMS (Manufacturing Message Specification) reporting.; Wide-Area Measurement System (WAMS) — Phasor Measurement Units (PMUs) with GPS-time-stamped phasor data; canonical PMU deployment in Indian grid since 2015 URTDSM project; SynchroPhasor Data Concentrators.; Microgrids — small grids with local generation (rooftop PV, diesel, biomass) and storage that can island and re-synchronise; canonical campus / industrial-park applications (IIT-Madras, IIT-Bombay microgrid pilots).; Grid-connected renewables challenges — variability, ramp rates, frequency response, low-inertia issues; the canonical 2024+ engineering problems with high RE penetration.; AI in power-system operation — load forecasting (ML-based), RE generation forecasting, anomaly detection on PMU data, congestion-management RL agents; the canonical 2025+ research direction at IISc, IIT-M, IIT-B.; India's smart-grid roadmap — National Smart Grid Mission (NSGM), Smart-Grid pilots (Mysuru, Naroda, Puducherry, Pune), AMI rollout under RDSS; the canonical 2025-2030 outlook.; Final-week lab — simulate a microgrid in MATLAB Simulink with solar PV + battery storage + diesel backup; demonstrate islanded and grid-connected operation; reflect on the canonical engineering challenges.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Power System Analysis and Design**	J. Duncan Glover, Thomas J. Overbye & Mulukutla S. Sarma	Cengage Learning	2017 (6th Ed.)
2	**Electrical Power Systems**	C. L. Wadhwa	New Age International	2009 (6th Ed.)
3	**Power System Engineering**	D. P. Kothari & I. J. Nagrath	McGraw-Hill Education India	2019 (3rd Ed.)
4	**A Course in Power Systems**	J. B. Gupta	S. K. Kataria & Sons	2013
5	**Power System Protection and Switchgear**	B. Ravindranath & N. Chander	New Age International	2011 (2nd Ed.)
6	**Smart Grid — Technology and Applications**	Janaka Ekanayake, Kithsiri Liyanage, Jianzhong Wu, Akihiko Yokoyama & Nick Jenkins	Wiley	2012
7	**Renewable Energy Integration — Practical Management of Variability, Uncertainty, and Flexibility in Power Grids**	Lawrence E. Jones	Academic Press	2017 (2nd Ed.)

Electric Vehicle Systems & Battery Technologies

EVBT REG · 26EE4110 PBL · 26EE4110E

L-T-P-S	1-0-2-4 / 3-0-0-6	Credits	3	Contact Hrs/Wk	7 / 9
Term (Reg · PBL)	S07 · T10	Mode	Regular & PBL	Category	PCC
Pre-requisite	PEWB				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze electric-vehicle architectures (BEV, HEV, PHEV, FCEV) and the powertrain components and their interactions.	BTL4 · Analyze
CO2	Analyze vehicle dynamics and the traction power/energy requirement — the drive cycle, force balance and motor sizing — for an EV.	BTL4 · Analyze
CO3	Analyze battery cell chemistries, characteristics (energy/power density, cycle life, safety) and pack architecture for EV applications.	BTL4 · Analyze
CO4	Apply Battery Management System functions — state-of-charge/health estimation, cell balancing, thermal and safety management — to a battery pack.	BTL3 · Apply
CO5	Apply traction-drive and regenerative-braking control concepts (using PEWC converters and CSEA control as building blocks) to the EV powertrain.	BTL3 · Apply
CO6	Analyze EV charging infrastructure (AC/DC, standards, fast charging) and vehicle-to-grid integration, including the Indian standards and policy context.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO6	PO7	PSO1
CO1	H	H	–	–	–	H
CO2	H	H	–	–	–	H
CO3	H	H	–	–	H	H
CO4	H	–	H	–	–	H
CO5	H	–	H	–	–	H
CO6	H	–	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**EV Architectures and Powertrain** The EV transition: drivers, India's e-mobility mission and FAME/PLI policy.; Architectures: BEV, HEV (series/parallel), PHEV, FCEV.; Powertrain components: battery, converter, inverter, motor, transmission.; Energy flow in the EV; efficiency at each stage.; Range, performance and cost trade-offs.; 48 V mild hybrids and two-/three-wheeler EVs (the India volume segment).; Comparison with IC vehicles.; Framing the EV design case for the project.	CO1
M2	**Vehicle Dynamics and Traction Requirements** Forces on a vehicle: rolling resistance, aerodynamic drag, gradient, acceleration.; The tractive-effort and power equation.; Drive cycles (e.g. MIDC/WLTP) and energy consumption.; Motor sizing from the drive cycle.; Choice of traction motor: PMSM, induction, BLDC, SRM (construction owned by EMDE).; Gear ratio and single-speed transmission.; Range estimation from battery capacity and consumption.; Sizing the powertrain for the project vehicle.	CO2
M3	**Battery Cells, Chemistries and Packs** Cell fundamentals: voltage, capacity, C-rate, energy vs power cells.; Chemistries: Li-ion (NMC, LFP), and emerging (sodium-ion, solid-state) at concept level.; Key metrics: energy density, power density, cycle life, calendar life, cost.; Cell formats: cylindrical, prismatic, pouch.; Pack architecture: series/parallel configuration, modules, the pack.; Battery modelling: equivalent-circuit models.; Degradation and ageing mechanisms.; Designing a pack to meet the project voltage/energy target.	CO3

	Title & Topics	CO
M4	**Battery Management Systems (BMS)** Why a BMS: safety, performance, longevity.; State-of-Charge (SoC) estimation: coulomb counting, OCV, Kalman-filter methods.; State-of-Health (SoH) estimation.; Cell balancing: passive vs active.; Thermal management: air, liquid, and battery thermal runaway prevention.; Protection: over/under-voltage, over-current, over-temperature.; BMS architecture and communication (CAN).; Specifying/simulating the BMS for the project pack.	CO4
M5	**Traction Drives and Regenerative Braking** The traction inverter (built from PEWC converter knowledge).; Motor control strategies: scalar vs vector (FOC) control (using CSEA control).; Torque and speed control for traction.; Regenerative braking: recovering energy, the limits.; Wide-bandgap (SiC/GaN) in EV powertrains (devices owned by PEWC).; Powertrain efficiency mapping.; Functional safety (ISO 26262) concept for the powertrain.; Modelling the traction-drive behaviour over a drive cycle.	CO5
M6	**Charging Infrastructure and V2G** AC charging: on-board charger, levels.; DC fast charging: off-board, the high-power path.; Charging standards: CCS, CHAdeMO, and India's connectors/AIS standards.; Charging-station architecture and the grid impact.; Vehicle-to-Grid (V2G) and bidirectional charging.; Smart charging and load management (links to MPSG/REGI).; Battery swapping (the India two/three-wheeler model).; A charging + V2G plan for the project vehicle for the demo.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Electric and Hybrid Vehicles: Design Fundamentals**	Iqbal Husain	CRC Press	2021 (3rd Ed.)
2	**Modern Electric, Hybrid Electric, and Fuel Cell Vehicles**	Mehrdad Ehsani et al.	CRC Press	2018 (3rd Ed.)
3	**Battery Management Systems, Volume I & II**	Gregory L. Plett	Artech House	2015
4	**Electric Powertrain**	John G. Hayes & G. Abas Goodarzi	Wiley	2018
5	**Lithium-Ion Batteries: Fundamentals and Applications**	Yuping Wu	CRC Press	2015

Embedded AI & Tiny ML

EATM 26EC3113

L-T-P-S	2-0-4-0	Credits	4	Contact Hrs/Wk	6
Semester	S05	Mode	Regular	Category	PCC
Pre-requisite	ESWD				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the embedded-AI motivation and constraints — the canonical cloud-vs-edge trade-off (latency, bandwidth, privacy, energy), the canonical TinyML resource envelope (< 1 MB Flash, < 256 KB RAM, < 100 mW power), and the application landscape.	BTL2 · Understand
CO2	Understand modern embedded-platform architectures — ARM Cortex-M0/M4/M7/M33/M55/M85 (with Helium MVE), ESP32 family (S3, C3, P4), STM32, RISC-V (BL808, ESP32-C6), Nordic nRF52840 / nRF54L — and select an appropriate MCU for an embedded-AI workload.	BTL2 · Understand
CO3	Apply neural-network compression techniques — post-training quantization (INT8 / INT4), quantization-aware training, structured / unstructured pruning, knowledge distillation, neural architecture search (MCUNet, MobileNetV3) — to fit large models into MCU memory.	BTL3 · Apply
CO4	Apply embedded-inference frameworks — TensorFlow Lite Micro, ExecuTorch (Meta), ONNX Runtime, Edge Impulse, microTVM, CMSIS-NN (ARM), TensorRT (Jetson) — to deploy a trained model to an MCU or single-board computer.	BTL3 · Apply
CO5	Apply sensor-integration techniques — IMU (accelerometer / gyro / magnetometer), MEMS microphone (PDM / I2S), camera modules (OV2640, OV5640, HiMax HM01B0), environmental sensors (BME280, SHT41, MQ-series) — with on-device pre-processing (windowing, MFCC, FFT, filtering) for ML pipelines.	BTL3 · Apply
CO6	Analyze low-power design, OTA updates, security and end-to-end TinyMLOps for production deployment — deep-sleep / wake-on-event, energy harvesting (solar, RF, vibration), secure boot, signed firmware, model versioning, fleet management — in Indian-context applications (smart agriculture, urban air quality, monsoon flood monitoring).	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PSO1
CO1	H	M	–	–	M	–	–	H
CO2	H	H	H	–	M	–	–	H
CO3	H	H	H	M	H	–	–	H
CO4	H	H	H	–	H	–	–	H
CO5	H	H	H	–	H	–	–	H
CO6	H	H	H	–	H	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Edge-AI Motivation & The TinyML Envelope** Cloud vs edge vs near-edge vs fog computing — the canonical latency / bandwidth / privacy / energy trade-off matrix; canonical Indian examples (Aadhaar Auth on-device biometrics, smart-meter rollout under National Smart Grid Mission, e-Sanjeevani telehealth).; TinyML definition (TinyML Foundation 2019; Reddi / Warden, Harvard CS249r) — the canonical <1 MB Flash, <256 KB RAM, <100 mW resource envelope; canonical "always-on, always-listening, always-watching" model.; The canonical TinyML application taxonomy — keyword spotting (KWS, Hey Siri / OK Google paradigm), visual wake words, anomaly detection, gesture recognition, predictive maintenance, sensor-fusion classification.; Edge-AI on richer platforms — Raspberry Pi (Cortex-A), NVIDIA Jetson Nano / Orin Nano (CUDA edge), Google Coral (Edge TPU), Hailo-8, AMD-Xilinx Kria; the canonical "edge AI" continuum from MCU to edge-server.; Industry context — the canonical Apple Neural Engine, Google Pixel Tensor / Tensor G4, Qualcomm Hexagon NPU, MediaTek APU, Intel Movidius / NPU; Indian-context: Ola Krutrim, Sarvam-1 edge variant.; Energy budget — the canonical microwatt vs milliwatt vs watt regime; coin-cell battery (~1000 mWh) vs LiPo (~5000 mWh) vs solar harvesting (canonical Indian sun-hours: 5-7 per day across most states).; Hardware-software co-design — the canonical MCUNet (Han et al. 2020) approach: search the network jointly with the inference scheduler; canonical 5-10x improvement vs unoptimized deployment.; Hands-on lab — Power-profile a sample TinyML application (Person Detection, MobileNetV2-quantized) on an ESP32-S3 + Joulescope IV meter; measure energy per inference (mJ), idle current ((mu)A), and project battery life on a 2000-mAh LiPo cell; document the energy-vs-accuracy Pareto frontier.	CO1
M2	**Embedded Platforms — MCU Architectures & Boards** ARM Cortex-M family — M0+/M3/M4F (canonical mid-range with FPU), M7 (high-performance with cache), M33 (TrustZone-M security), M55 / M85 (Helium MVE for ML acceleration); the canonical CMSIS-NN library.; ESP32 family (Espressif) — ESP32 (classic, dual-core Xtensa LX6), ESP32-S3 (with vector instructions for AI), ESP32-C3 / C6 (RISC-V), ESP32-P4 (RV64); the canonical Wi-Fi + BLE + low-cost-MCU sweet spot for Indian IoT projects.; Raspberry Pi Pico family — RP2040 (dual Cortex-M0+, no FPU), RP2350 (Cortex-M33 + RISC-V Hazard3, 2024); the canonical sub-$5 board for student projects.; STM32 family — STM32H7 (Cortex-M7 480 MHz), STM32U5 (ultra-low-power Cortex-M33), STM32N6 (with Neural-ART NPU, 2024); canonical STMicroelectronics X-CUBE-AI toolchain.; Nordic nRF families — nRF52840 (Cortex-M4F + BLE), nRF54L15 / nRF54H20 (multi-core, 2024); the canonical low-power BLE wearable / sensor-node platform.; RISC-V on the edge — BL808 (Bouffalo Lab, multi-core RV with AI accelerator), CH32V (WCH, ultra-cheap), SiFive Performance P670; the canonical 2024+ open-ISA momentum, India-context: Shakti (IIT-M), DIR-V mission.; Single-board computers (SBCs) — Raspberry Pi 5 (BCM2712 Cortex-A76), Orange Pi 5 (RK3588), NVIDIA Jetson Orin Nano (Ampere GPU + 8 GB LPDDR5), Google Coral Dev Board Mini (Edge TPU); when an MCU isn't enough.; Hands-on lab — Set up the dev environment for 3 boards: Arduino Nano 33 BLE Sense (Cortex-M4F), ESP32-S3 DevKit, Raspberry Pi Pico W; compile and flash a "hello sensors" sketch on each that reads IMU + temperature + humidity; benchmark MCU performance using CoreMark and the TinyML perf benchmark (MLPerf Tiny).	CO2
M3	**Model Compression for the Edge** Quantization fundamentals — the canonical FP32 → INT8 mapping (4x size reduction, 2-4x speedup on CPU, near-free on NPU); symmetric vs asymmetric quantization; per-tensor vs per-channel scales.; Post-Training Quantization (PTQ) — the canonical "calibration set" approach; representative dataset, percentile-clipping; canonical TFLite Converter + ONNX Runtime quantization APIs.; Quantization-Aware Training (QAT) — the canonical "fake-quantize during training" approach; canonical 1-2% accuracy recovery vs PTQ; PyTorch QAT and TF Model Optimization Toolkit.; Sub-INT8 quantization — INT4 (Bitsandbytes, GPTQ), INT2 (canonical 2024+ for LLMs), 1.58-bit (Microsoft BitNet, 2024); the canonical accuracy-vs-size trade-off; canonical limitations for vision vs language.; Pruning — unstructured (canonical magnitude-based, Han et al. 2015), structured (channel-pruning, filter-pruning), N:M sparsity (NVIDIA Ampere 2:4 hardware-accelerated); the canonical "prune-and-finetune" recipe.; Knowledge Distillation (Hinton et al. 2015) — the canonical teacher-student approach; soft targets, temperature; canonical 2024+ "distillation for LLMs" (DistilBERT, TinyLLaMA, Phi-3-mini); MobileNet families.; Neural Architecture Search for MCUs — MCUNet (Lin et al. NeurIPS 2020), MCUNetV2 (Lin et al. 2021), MicroNets (Banbury et al. MLSys 2021); the canonical "co-search architecture + scheduler" approach.; Hands-on lab — Take a trained Hindi keyword-spotting model (~1 MB FP32, 100 K parameters) and compress it for ESP32-S3: apply PTQ INT8 → measure accuracy drop; apply QAT → recover accuracy; apply 50% structured pruning + retrain; compare final-size / accuracy / inference-latency vs original on a held-out test set.	CO3

	Title & Topics	CO
M4	**Edge Inference Frameworks** TensorFlow Lite Micro (TFLM) — the canonical CMSIS-NN-backed inference framework for MCUs; the canonical Pete Warden + Daniel Situnayake approach; canonical "convert TF → TFLite → TFLM" pipeline.; ExecuTorch (Meta, 2024 GA) — the canonical PyTorch-native edge runtime; canonical advantages over TFLM (TorchScript-free, modular backends); canonical mobile + MCU support.; ONNX Runtime — the canonical interop framework; ORT-Mobile, ORT-Web; canonical advantages for cross-framework portability.; Edge Impulse — the canonical end-to-end no-code TinyML platform (data ingestion → DSP block → NN block → deployment); canonical advantages for rapid prototyping; canonical limitations vs custom pipelines.; microTVM (Apache TVM) — the canonical compiler-based edge deployment; auto-tuning for target MCU; canonical 2-5x speedup over hand-optimized kernels.; Vendor-specific frameworks — STMicroelectronics X-CUBE-AI, NXP eIQ, Renesas e-AI Translator, Silicon Labs Machine Learning Toolkit; the canonical "vendor SDK gives best perf on their MCU" pragmatism.; Larger-edge frameworks — TensorRT (NVIDIA Jetson), OpenVINO (Intel), Edge TPU compiler (Google Coral), Hailo Dataflow Compiler; for "edge AI" beyond MCUs.; Hands-on lab — Deploy a person-detection model (MobileNetV2 SSDLite) two ways to an Arduino Nano 33 BLE Sense + OV7670 camera: (a) via TFLM with CMSIS-NN, (b) via Edge Impulse; benchmark frames/sec, RAM usage, accuracy on a custom Indian-pedestrian dataset; profile cycle-counts per layer.	CO4
M5	**Sensors, Signal Processing & On-Device Features** IMU sensors — the canonical MPU-6050 / MPU-9250 / LSM6DSOX / ICM-42670 (accelerometer + gyroscope, optional magnetometer); I2C / SPI interfaces; sampling rate and filtering (low-pass, high-pass, complementary, Kalman).; MEMS microphones — PDM (Pulse Density Modulation) vs I2S (Inter-IC Sound); canonical examples (Knowles SPH0645, ST MP34DT06J, InvenSense ICS-43434); PDM-to-PCM decimation.; Audio feature extraction — framing + windowing (Hamming, Hann), STFT, MFCC (Mel-Frequency Cepstral Coefficients), log-Mel spectrograms; the canonical KWS pipeline (Hello Edge, Zhang et al. 2017).; Camera modules — OV2640 (UXGA, JPEG), OV5640 (5MP, MIPI), HiMax HM01B0 (ultra-low-power, monochrome); image-signal-processing pipeline: demosaicing, white-balance, gamma; the canonical "resize to 96×96 for MCU" pre-processing.; Environmental sensors — BME280 / BME680 (temp + humidity + pressure + VOC), SHT41 (high-accuracy temp/humidity), MQ-series gas sensors, PM2.5 (Plantower, Sensirion SPS30); canonical applications in India air-quality (CPCB, IndiaSpend / Climate Trends data).; Signal processing on the MCU — ARM CMSIS-DSP (FFT, FIR / IIR filters, matrix ops), ESP32-DSP (Espressif library); canonical optimizations for Cortex-M4F / M7 FPU + SIMD.; Sensor fusion — the canonical complementary / Kalman / Madgwick / Mahony filters for orientation; canonical applications (drone IMU, wearable activity recognition).; Hands-on lab — Build a Hindi keyword-spotter on an Arduino Nano 33 BLE Sense: collect Hindi audio for 4 commands ("chalo", "ruko", "daayen", "baayen") + background, extract MFCC features on-device, train a small CNN on a laptop, deploy via TFLM with INT8 quantization; demonstrate live inference and accuracy.	CO5
M6	**Low Power, Security, OTA & TinyMLOps** Low-power design — the canonical sleep / deep-sleep / hibernation modes; wake-on-IMU-tap (LSM6DSOX), wake-on-sound-event (PDM mic threshold), wake-on-image-difference (HM01B0); RTC-driven scheduling.; Energy harvesting — solar (canonical Indian-sun model: 5-7 kWh/m²/day), thermoelectric (Seebeck effect), vibration (piezo), RF (LoRa-WAN, ambient Wi-Fi); canonical PMIC ICs (BQ25570, AEM10941).; Wireless connectivity — BLE 5.x (sub-mW), LoRa / LoRaWAN (10+ km range, mW-class), NB-IoT / LTE-M (telco network), Zigbee 3.0, Thread / Matter (canonical 2023+ smart-home standard); Indian-context: BSNL LoRa, Reliance Jio NB-IoT.; Secure boot — the canonical chain-of-trust from boot ROM → bootloader → firmware → application; ESP32 secure-boot + flash-encryption; STM32 TrustZone; ARM PSA Certified.; OTA (Over-The-Air) updates — the canonical A/B partition scheme (Espressif ESP-IDF OTA, MCUBoot bootloader); delta updates (Bsdiff, courgette); signed firmware (Ed25519, ECDSA); rollback protection.; TinyMLOps — the canonical "MLOps for the edge" emerging discipline; model versioning, fleet management (Edge Impulse, Arduino Cloud, NXP Edge Insights, AWS IoT Greengrass + SageMaker Edge); the canonical "deploy to 1000 devices, monitor drift, push v2" lifecycle.; Indian-context production deployments — the canonical case studies: SatSure (satellite + edge crop analytics), Kissht (rural fintech KYC), CropIn (precision agriculture in 92 countries), Stellapps (dairy farm IoT), Detect Technologies (industrial predictive maintenance).; Final hands-on lab — Build a complete end-to-end TinyML deployment: a solar-powered ESP32-S3 + BME280 + microphone + SX1262 LoRa node that detects "unusual" agricultural-field acoustic events (engine, gunshot, livestock distress) using an INT8 CNN on MFCC features; runs from a 200-mWh solar harvester; reports via LoRaWAN to a Helium / ChirpStack gateway; supports signed OTA updates; produce an end-of-semester field deployment report with energy + accuracy + connectivity measurements over 7 days.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**TinyML — Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers**	Pete Warden & Daniel Situnayake	O'Reilly Media	2019

#	Title	Authors	Publisher	Year
2	**TinyML Cookbook — Combine artificial intelligence and ultra-low-power embedded devices to make the world smarter**	Gian Marco Iodice	Packt Publishing	2023 (2nd Ed.)
3	**AI at the Edge — Solving Real-World Problems with Embedded Machine Learning**	Daniel Situnayake & Jenny Plunkett	O'Reilly Media	2023
4	**Deep Learning for Vision Systems**	Mohamed Elgendy	Manning Publications	2020
5	**Machine Learning Systems — Principles and Practices of Engineering Artificially Intelligent Systems**	Vijay Janapa Reddi (Editor)	Harvard / CS249r open textbook	2024 (continuously updated)
6	**Practical Deep Learning for Cloud, Mobile, and Edge**	Anirudh Koul, Siddha Ganju & Meher Kasam	O'Reilly Media	2019
7	**Embedded Machine Learning for Cyber-Physical, IoT, and Edge Computing**	Sudeep Pasricha & Muhammad Shafique (Eds.)	Springer	2024

Embedded AI & Tiny ML

EATM 26EC3113E

L-T-P-S	3-0-6-0	Credits	4	Contact Hrs/Wk	9
Trimester	T07	Mode	PBL (Project-Based)	Category	PCC
Pre-requisite	DLRN				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze a TinyML on-device system as a layered system — end product, quantised model, on-device runtime, MCU integration, sensor preprocessing — and trace one inference from sensor sample to actuation.	BTL4 · Analyze
CO2	Apply model-compression techniques for MCU envelopes — INT8 quantisation, structured pruning, knowledge distillation, depthwise-separable convolutions — to fit useful models within KB-scale flash and RAM budgets.	BTL3 · Apply
CO3	Apply on-device inference runtimes — TensorFlow Lite for Microcontrollers, CMSIS-NN, ONNX Runtime for MCU — to deploy quantised models with predictable latency and memory footprint.	BTL3 · Apply
CO4	Apply MCU-integration patterns — DMA-driven sensor capture, ring-buffer audio streams, MFCC and other on-device preprocessing, periodic-tick model invocation under an RTOS, low-power wake-on-detection — to integrate ML into a real-time embedded application.	BTL3 · Apply
CO5	Apply edge-training practices — transfer learning from a pretrained backbone, on-device data collection, augmentation strategies for tiny datasets, federated learning at intuition level — to specialise models for a target deployment.	BTL3 · Apply
CO6	Analyze the full TinyML lifecycle — dataset curation, training, quantisation calibration, deployment, on-device measurement, iteration — and reason about energy budget, latency budget, and accuracy floor for a target product.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PSO1
CO1	H	H	H	–	–	H
CO2	H	–	H	–	–	H
CO3	H	–	H	–	H	H
CO4	H	–	H	M	–	H
CO5	H	–	H	–	–	H
CO6	H	H	–	H	–	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**On-Device AI as a System** A working keyword spotter on day one — students say "Hey KLU" near the STM32 board and watch an LED light up before any internals.; The TinyML system stack: microphone → DMA buffer → preprocessing → model (TFLite Micro) → postprocessing → action (RTOS task signalled).; The MCU envelope reality: 256 KB of flash, 192 KB of RAM, no MMU, single-precision FPU at best, sub-50mA average power; the engineering envelope this defines.; The TinyML problem statement: a model that fits AND that meets latency AND that meets accuracy AND that meets power; choose three, optimise the fourth — usually a four-way trade.; Tracing one keyword-spot inference end-to-end: 16 kHz mic samples → 25 ms frames → 13 MFCCs → INT8 CNN → softmax-over-3-classes → action.; The TinyML application landscape: keyword spotting, person detection, anomaly detection on industrial machinery, gesture recognition, predictive maintenance.; A reading of WhisperPort's architecture: the sensor task, the preprocessing module, the model wrapper, the action task; how it sits on top of ESDT's FreeRTOS.; What this course owns vs what comes downstream: TinyML envelope here; GPU-class DL in DLAC; cloud LLMs in NLPL.	CO1

	Title & Topics	CO
M2	**Model Compression for the Edge** Why compression: a 5 MB pretrained model does not fit on a 256 KB MCU; compression is non-optional, not an optimisation.; Quantisation revisited from DLAC at the embedded envelope: INT8 is the bread-and-butter, INT4 is the frontier, FP16 rarely useful on Cortex-M (no FP16 vector unit).; Post-training quantisation with TFLite: representative dataset, calibration, the per-tensor quantisation parameters.; Quantisation-aware training (QAT) for the cases where PTQ loses too much accuracy.; Pruning for MCUs: structured pruning (channel removal) is what hardware actually exploits; unstructured pruning saves model size but rarely improves runtime on MCU.; Knowledge distillation: training a tiny student to imitate a large teacher; the canonical recipe for compressing a vision model 50×.; Architecture choice for tiny: depthwise-separable convolutions (MobileNet-V2 style), shallow networks, the MicroNet family designed for sub-1MB budgets.; Decision framework: given a model and a target MCU, what compression sequence (architecture choice → distill → prune → quantise) gets you to fit?	CO2
M3	**On-Device Inference Runtimes** TensorFlow Lite for Microcontrollers (TFLite Micro): the canonical TinyML runtime; the supported-ops catalogue; the C++ inference API.; CMSIS-NN: ARM's optimised neural-network kernel library for Cortex-M; how it accelerates TFLite Micro under the hood.; ONNX Runtime for MCU: the Microsoft-backed alternative; broader ops support; the comparison.; glow / Apache TVM at preview level: ahead-of-time compilation of models into native code; the speedup over interpreted runtimes.; edgeimpulse SDK and similar vendor tools at preview level; the productivity-vs-vendor-lock trade-off.; Memory layout in TFLite Micro: arena allocator, tensor lifecycle, the static-vs-dynamic memory choice; predicting the memory footprint before flashing.; Op selection: every supported op costs flash; the discipline of including only what the model uses.; A walkthrough of a TFLite Micro inference call: the interpreter, input tensor mapping, Invoke(), output tensor extraction; the C++ idioms.	CO3
M4	**MCU Integration** Audio capture on STM32H7: I²S peripheral, MEMS microphone (e.g. INMP441), DMA-driven double-buffered capture into a ring buffer.; Audio preprocessing on-device: framing (25 ms windows, 10 ms hop), pre-emphasis, windowing (Hamming), FFT, mel filterbank, log-mel, MFCC; CMSIS-DSP as the implementation library.; Periodic model invocation under FreeRTOS: a 50 ms tick that triggers the spotter task; staying within the budget every tick.; Sensor preprocessing patterns more broadly: image (downscale + INT8 quantise from camera), IMU (windowing + feature extraction for activity recognition), vibration (FFT magnitude for anomaly detection).; Power discipline: sub-1mA average via aggressive low-power modes; wake-on-event via PDM mic with hardware VAD; the always-on / occasionally-on architecture.; The post-inference action: signalling an RTOS task, raising an interrupt, sending a CAN frame, lighting an LED; closing the loop from sensor to outcome.; On-device debugging: SWO trace for inference timing, GPIO toggles for cycle-accurate measurement, CMSIS event recorder for profiling.; A walkthrough of WhisperPort's integration: how the audio task feeds the spotter task feeds the RTOS scheduler's notification primitive.	CO4
M5	**Training for the Edge** The data-scarcity problem: most TinyML applications start with hundreds of samples, not millions; what changes.; Transfer learning: starting from a pretrained backbone and fine-tuning on the target task; the canonical TinyML productivity multiplier.; On-device data collection: turning the deployed device into a data-gathering tool; the privacy and storage discipline.; Data augmentation strategies for audio: time stretching, pitch shifting, noise overlay, room impulse response.; Data augmentation for vision on edge: standard imagenet-style augmentations PLUS device-specific lens distortions and lighting.; Calibration data for quantisation: how to pick a representative subset; the failure mode when calibration data is unrepresentative.; Federated learning at intuition level: training across many devices without centralising data; communication-efficient updates; the privacy story.; On-device personalisation at preview level: tiny on-device fine-tuning to specialise to a particular user; the engineering challenges.	CO5
M6	**The Full TinyML Lifecycle** The TinyML lifecycle laid out as a workflow: dataset curation → train → quantise → deploy → measure on-device → iterate.; Energy budget: estimating µJ-per-inference; the energy-vs-accuracy frontier; battery life calculations.; Latency budget: measuring real on-device latency (not the simulator); the gap between desktop simulation and MCU reality.; Accuracy floor: how good is "good enough" for the product? The discipline of stating the floor before training.; Iteration discipline: deploy → measure failure cases → augment dataset → retrain → redeploy; the equivalent of MLNG's ML-engineering rituals at the edge.; Telemetry from edge fleets: reporting accuracy and false-positive rates back to a central server without violating privacy.; Model versioning on the edge: multiple models on flash, A/B switching, OTA model update (revisiting ESDT's OTA pipeline).; The TinyML production checklist: does the model fit? Does it run within budget? Is power within spec? Is accuracy floor met? Is the OTA path tested? Are failure modes documented?	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**TinyML — Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers**	Pete Warden & Daniel Situnayake	O'Reilly	2019

#	Title	Authors	Publisher	Year
2	**TinyML Cookbook — Combine artificial intelligence and ultra-low-power embedded devices**	Gian Marco Iodice	Packt	2023 (2nd Ed.)
3	**AI at the Edge — Solving Real-World Problems with Embedded Machine Learning**	Daniel Situnayake & Jenny Plunkett	O'Reilly	2023
4	**Hands-On Embedded Programming with Modern C++**	Maya Posch	Packt	2024
5	**Embedded Deep Learning — Algorithms, Architectures and Circuits for Always-On Neural Network Processing**	Bert Moons, Daniel Bankman & Marian Verhelst	Springer	2018
6	**Efficient Deep Learning — Fast and Accurate Inference**	Gaurav Menghani	Manning	2024
7	**Mastering the FreeRTOS Real-Time Kernel — A Hands-On Tutorial Guide**	Richard Barry & Real Time Engineers Ltd.	AWS / Free PDF	2024 (revised)

Embedded Software Development

ESWD 26EC2310

L-T-P-S	2-0-4-0	Credits	4	Contact Hrs/Wk	6
Semester	S04	Mode	Regular	Category	PCC
Pre-requisite	ESDI				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply embedded software fundamentals — memory-mapped I/O, register access, interrupt handling, the discipline of working with peripheral datasheets — to write firmware that controls hardware reliably.	BTL3 · Apply
CO2	Apply communication peripherals — UART, SPI, I2C, CAN at intuition — to interface with sensors, actuators, and other MCUs.	BTL3 · Apply
CO3	Apply RTOS fundamentals — tasks, scheduling, semaphores, queues, FreeRTOS / Zephyr APIs — to build multi-tasking firmware.	BTL3 · Apply
CO4	Analyze real-time scheduling — rate-monotonic and earliest-deadline-first at intuition, priority inversion, deadline analysis — to reason about timing-critical systems.	BTL4 · Analyze
CO5	Apply driver development patterns — HAL APIs vs direct register, interrupt-driven I/O, DMA at intuition, the abstraction layers in production firmware.	BTL3 · Apply
CO6	Analyze contemporary embedded development — Rust on MCUs at intuition, edge AI deployment via TensorFlow Lite Micro, OTA updates, secure-boot fundamentals, the AI-augmented embedded workflow.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO6	PSO1
CO1	H	–	H	–	–	H
CO2	H	–	H	–	–	H
CO3	H	–	H	–	–	H
CO4	H	H	H	–	–	H
CO5	H	–	H	–	–	H
CO6	H	–	H	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Embedded Software Foundations** The embedded mental model — firmware running on a small MCU, no OS, direct hardware access; the contrast with application programming.; Memory-mapped I/O — peripheral registers as memory addresses, the volatile keyword, the canonical bug from forgetting it.; Cortex-M architecture at intuition — Thumb-2, NVIC interrupt controller, SysTick timer; the canonical reference platform.; Toolchain fundamentals — gcc-arm-none-eabi, the linker script, the startup file, the canonical build flow.; Modern IDE landscape — STM32CubeIDE, PlatformIO (the modern alternative), Zephyr west; the productivity gain.; Debugging fundamentals — GDB over OpenOCD, the SWD protocol; the canonical debugging workflow with breakpoints and watchpoints.; A first lab: blink an LED on STM32 / ESP32 / NXP via direct register write; observe via oscilloscope; switch to HAL API; compare code clarity.	CO1
M2	**Communication Peripherals** UART — async serial communication, baud rate, framing, the canonical commissioning protocol.; SPI — master-slave synchronous, MOSI/MISO/SCLK/CS, modes 0-3; the canonical fast-and-simple bus.; I2C — multi-master serial, START/STOP/ACK protocol, 7-bit addressing; the canonical sensor bus.; CAN at intuition — automotive-canonical multi-master broadcast bus; depth in automotive-track courses downstream.; USB at intuition — the host-vs-device distinction, USB classes (CDC, HID, MSC); the canonical "USB-to-serial" CDC class.; Modern wireless — BLE via NimBLE / softdevice, WiFi via ESP-IDF; the bridge to FOIS-deeper IoT.; A lab: interface DHT22 via I2C, BMP280 via SPI, ESP32 via UART (between two boards); demonstrate three-protocol simultaneous communication on STM32 / ESP32.	CO2

	Title & Topics	CO
M3	**RTOS Fundamentals** Why RTOS — multi-tasking, deterministic scheduling, ease of writing concurrent firmware; the contrast with bare-metal super-loop.; Tasks and the scheduler — priorities, ready/blocked/running states; the canonical context-switch mechanism.; Synchronization primitives — semaphores (counting, binary, mutex), queues, event groups.; FreeRTOS API — xTaskCreate, vTaskDelay, xQueueSend / xQueueReceive, the canonical patterns.; Zephyr RTOS at intuition — the Linux Foundation backing, the kconfig + DTS approach, why Zephyr is the modern industrial choice.; Stack management — per-task stacks, stack overflow detection, the canonical bug source in embedded multi-tasking.; A lab: in FreeRTOS on STM32 / ESP32, build a 3-task system — sensor task reads data at 10Hz, processing task computes moving average, display task updates a 16×2 LCD; use a queue for inter-task communication.	CO3
M4	**Real-Time Scheduling** Real-time vs general-purpose computing — the deadline-meeting guarantee.; Hard, firm, and soft real-time — the canonical distinction (anti-lock brakes vs MP3 decode vs UI animation).; Rate-monotonic scheduling at intuition — fixed priorities by period; Liu and Layland's schedulability test.; Earliest-deadline-first at intuition — dynamic-priority alternative; the trade-offs.; Priority inversion — the canonical Mars Pathfinder bug; the priority-inheritance and priority-ceiling protocols.; Worst-case execution time analysis at intuition — measurement-based vs static-analysis approaches.; Schedulability analysis — given a set of tasks with periods and worst-case execution times, can they all meet deadlines?; A lab: model a 5-task system in software (no real hardware), apply RMS analysis, verify schedulability; introduce a priority-inversion scenario; observe the deadline miss; fix with priority inheritance.	CO4
M5	**Driver Development Patterns** Driver abstractions — HAL APIs vs LL (low-level) APIs vs direct register; the trade-offs (portability vs performance vs clarity).; Interrupt-driven I/O — UART RX in interrupt mode, SPI completion interrupt, the canonical patterns.; Ring buffers for I/O — the canonical pattern for decoupling interrupt handlers from main code.; DMA at intuition — peripheral-to-memory and memory-to-peripheral, double-buffering, the canonical performance win for high-rate ADC sampling.; The driver-application boundary — a clean API design discipline that pays back during testing and reuse.; Testing embedded code — host-side unit tests on driver logic, hardware-in-the-loop for full validation; the modern testing practices.; A lab: write a clean UART driver with a ring-buffered interrupt-driven RX; expose a putc/getc API to the application; build a small command-line interpreter on top.	CO5
M6	**Modern Embedded — Rust, Edge AI, Security, AI-Augmented Workflow** Rust on MCUs at intuition — the embedded-rust ecosystem; cortex-m / cortex-m-rt; why Rust is the safer alternative for new safety-critical projects.; TensorFlow Lite Micro / TinyML at intuition — running a 50KB neural network on a Cortex-M4 at 80 MHz; the canonical keyword-spotting demo.; OTA (over-the-air) updates — the canonical bootloader + dual-bank pattern; the implications for production IoT deployments.; Secure boot — chain of trust from ROM bootloader to application; the canonical TLS / X.509 use cases for IoT.; Modern observability for firmware — RTT (Real-Time Transfer), Memfault, Tracealyzer; the productivity gains for production debugging.; AI coding assistants for embedded — Claude / Copilot / Cursor for HAL boilerplate, peripheral configuration; the discipline of always testing on hardware (the AI assistant cannot see the board).; When AI assistants fail in embedded — register-bit miscalculation, missed interrupt-clear, missing volatile, compiler-version-specific bugs; the eye students must develop.; A final-week lab: deploy a TFLite Micro keyword-spotting model on ESP32 / STM32 with an I2S MEMS microphone; verify it can detect "yes/no/up/down" with >85% accuracy; measure inference time; submit to GitHub.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Mastering STM32**	Carmine Noviello	Leanpub	2024
2	**Real-Time Embedded Multithreading Using ThreadX**	Edward L. Lamie	CRC Press	2020 (4th Ed.)
3	**Mastering FreeRTOS — The FreeRTOS Real Time Kernel**	Richard Barry	FreeRTOS (free PDF)	2021
4	**Making Embedded Systems — Design Patterns for Great Software**	Elecia White	O'Reilly	2024 (2nd Ed.)
5	**Discovering the STM32 Microcontroller**	Geoffrey Brown	Free PDF (Indiana University)	2016
6	**TinyML — Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers**	Pete Warden & Daniel Situnayake	O'Reilly	2019
7	**The Rust on Embedded Devices Book**	Embedded Working Group	Online (free)	2024

Embedded Software Development

ESWD 26EC2310E

L-T-P-S	3-0-6-0	Credits	4	Contact Hrs/Wk	9
Trimester	T06	Mode	PBL (Project-Based)	Category	PCC
Pre-requisite	ESDI				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze a production-style RTOS-based embedded system as a layered system — application tasks, RTOS kernel services, HAL and drivers, hardware — and explain the data flow between concurrent tasks.	BTL4 · Analyze
CO2	Analyze RTOS scheduling — preemptive priority-based scheduling, rate-monotonic and earliest-deadline-first, schedulability analysis, priority inversion and inheritance — and design a task set that meets its deadlines.	BTL4 · Analyze
CO3	Apply RTOS inter-task communication — queues, binary and counting semaphores, mutexes with priority inheritance, event groups, direct task notifications — to design correct multi-task systems.	BTL3 · Apply
CO4	Apply embedded driver architecture — the driver-HAL layering, interrupt handlers within an RTOS, deferred-procedure-call patterns, DMA-driven peripheral I/O — to integrate hardware peripherals into a multi-task application.	BTL3 · Apply
CO5	Apply embedded build and debug tooling — CMake with the ARM cross-compile toolchain, OpenOCD with GDB, SEGGER SystemView for RTOS-aware tracing, embedded CI — to operate as a professional embedded-software team.	BTL3 · Apply
CO6	Apply embedded-reliability practices — watchdog timers, fault handlers, bootloader architecture, secure firmware update over the network — to deliver a deployable production-style embedded product.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PSO1
CO1	H	H	H	–	–	–
CO2	H	H	H	–	–	H
CO3	H	–	H	–	–	H
CO4	H	–	H	–	M	–
CO5	H	–	–	–	H	H
CO6	H	–	H	H	H	–

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The RTOS-Based Servo as a System** A working RTOS-based ServoMaster v2 running on day one — students issue commands via CAN and observe three concurrent tasks running cleanly through SystemView tracing.; The architectural shift from EBSD: bare-metal super-loop → RTOS-scheduled multi-task application; what motivates the move and what it costs.; The four canonical RTOS layers: application tasks, kernel services (scheduler + IPC), HAL + drivers, hardware.; FreeRTOS as the chosen RTOS: industry ubiquity, MIT licensed, AWS-stewarded, the Cortex-M port; alternative RTOS choices (Zephyr, RTEMS, ThreadX) at preview level.; The three ServoMaster v2 tasks: control task (high priority, 1 kHz), telemetry task (medium, 50 Hz), command task (medium, event-driven); the rationale for the split.; Tracing one CAN command end-to-end: ISR → queue → command task → semaphore → control task acts on new setpoint; the cleanly-decoupled design.; A reading of the ServoMaster v2 codebase: the FreeRTOS configuration, the task entry points, the queue declarations, the driver layer.; What this course owns vs what comes downstream: RTOS + embedded-software architecture here; embedded AI in EATM; robotics in FRMS.	CO1

	Title & Topics	CO
M2	**RTOS Scheduling** The RTOS scheduler's job: deciding which task runs at each instant; the contrast with bare-metal cooperative loops.; Preemptive priority-based scheduling: every tick, the highest-ready task runs; how preemption interacts with critical sections.; Rate-monotonic scheduling (RMS) theorem: priority inversely proportional to period; the famous 0.69 utilisation bound for n→∞.; Earliest-deadline-first (EDF) scheduling: optimal but rarely used in practice; why most embedded systems use RMS.; Schedulability analysis: response-time analysis, the iterative formula; ensuring the worst-case response time is below the deadline.; The idle task and its uses (low-power sleep, statistics, the heartbeat); the tickless idle mode for power-sensitive designs.; Time slicing among equal-priority tasks: when it helps, when it hurts; usually disabled for hard-real-time work.; Common scheduling pitfalls: starvation of low-priority tasks, the ISR-priority-vs-task-priority confusion, the busy-wait-in-task disaster.	CO2
M3	**Inter-Task Communication** Why explicit IPC: in an RTOS, tasks cannot share state via globals safely; explicit IPC primitives are the only correct path.; Queues: producer-consumer pattern, blocking sends and receives, queue sets for waiting on multiple queues.; Binary semaphores: signalling a task from an ISR; the canonical "wake the worker" pattern.; Counting semaphores: resource-pool management; the producer-consumer with bounded buffer.; Mutexes: the priority-inheritance variant in FreeRTOS; protecting shared resources without inversion.; Priority inversion: the Mars Pathfinder story; why mutexes (with inheritance) are safer than semaphores for resource protection.; Event groups: setting and waiting on multiple bits; the natural fit for "wait for any of these conditions".; Direct task notifications: lightweight, fast, the modern FreeRTOS-recommended pattern; when notifications replace queues and semaphores.	CO3
M4	**Drivers and HAL Architecture** The driver-HAL layering revisited: HAL provides the hardware-abstraction surface; drivers provide the application-facing API; tasks consume drivers.; Interrupt service routines under an RTOS: the FromISR variants of FreeRTOS APIs; the deferred-handling pattern.; The deferred-procedure-call (DPC) pattern: ISR signals a queue, a high-priority task processes; the ISR stays microsecond-short.; DMA-driven peripheral I/O: ADC scan into a buffer, UART receive ring, SPI block transfer; freeing the CPU for tasks.; Driver-state isolation: every driver instance owns its mutex, queue, and config; no global state.; Building reusable drivers: the device-tree mental model (without the Linux DT machinery); a driver as a typed structure plus an init function plus an API.; Power management at intro level: low-power modes, wake sources, the trade-off between latency and power.; A walkthrough of ServoMaster v2 driver layer: the encoder driver, the motor driver, the CAN driver — all RTOS-aware.	CO4
M5	**Embedded Build and Debug Pipeline** Why CMake for embedded: portability across host platforms, integration with IDEs (CLion, VSCode), the modern alternative to Makefiles.; The ARM cross-compile toolchain: arm-none-eabi-gcc, the linker script, startup code; how a binary becomes a flashable image.; Linker scripts at intuition level: memory regions, sections, the text/data/bss layout; what changes for FreeRTOS.; OpenOCD: the on-chip debugger; configurations for ST-LINK; the GDB remote-protocol bridge.; GDB-based debugging: setting breakpoints in interrupt handlers, examining task contexts, the FreeRTOS-aware GDB scripts.; SEGGER SystemView: RTOS-aware tracing; visualising task switches, ISR enters, queue operations on a timeline.; Static analysis for embedded: clang-tidy, MISRA C compliance at preview level, cppcheck.; Embedded CI: GitHub Actions runners building firmware, QEMU-based unit testing on host, hardware-in-the-loop CI as a glimpse.	CO5
M6	**Reliability and Production** Watchdog discipline revisited: each task feeds a sub-watchdog; a missed feed indicates the task hung; the supervisor task aggregates.; Fault handlers: HardFault, MemManage, BusFault, UsageFault on Cortex-M; capturing the fault context for postmortem analysis.; Defensive programming on embedded: assert() that does not abort production but logs and recovers; the discipline of fault containment.; Bootloader architecture: dual-bank flash, A/B partitioning, the boot-pin mechanism, recovery boot.; Secure boot at intro level: cryptographic verification of the firmware image at boot; the chain of trust.; OTA firmware update: the protocol design (chunked transfer over CAN/UART), the verify-then-switch pattern, the rollback path.; Field debugging: structured logging to flash, snapshot capture on fault, telemetry-based observability for embedded fleets.; Production embedded engineering practices: code review for safety-critical code, MISRA-style coding standards, certification at preview level (ISO 26262 for automotive).	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Mastering the FreeRTOS Real-Time Kernel — A Hands-On Tutorial Guide**	Richard Barry & Real Time Engineers Ltd.	AWS / Free PDF	2024 (revised)
2	**Embedded Systems Architecture — Explore Architectural Concepts and Design Patterns**	Daniele Lacamera	Packt	2023 (2nd Ed.)

#	Title	Authors	Publisher	Year
3	**Real-Time Embedded Systems — Design Principles and Engineering Practices**	Xiaocong Fan	Newnes (Elsevier)	2015
4	**Programming Embedded Systems in C and C++ — With Embedded Linux and ARM Cortex-M**	Michael Barr & Anthony Massa	O'Reilly	2024 (3rd Ed.)
5	**Test-Driven Development for Embedded C**	James W. Grenning	Pragmatic Bookshelf	2011
6	**Making Embedded Systems — Design Patterns for Great Software**	Elecia White	O'Reilly	2024 (2nd Ed.)
7	**Modern CMake for C++**	Rafał Świdziński	Packt	2022

Embedded System Design & IoT

ESDI 26EC2106

L-T-P-S	2-0-4-0	Credits	4	Contact Hrs/Wk	6
Semester	S03	Mode	Regular	Category	ESC
Pre-requisite	FOIS				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply ARM Cortex-M architectural concepts (core pipeline, memory model, NVIC, system tick) to characterise an STM32H7-class microcontroller and its programming model.	BTL3 · Apply
CO2	Apply STM32 peripheral subsystems — GPIO, ADC, timers, PWM, USART, SPI, I²C — through STM32 HAL to interface with sensors and actuators in small embedded programs.	BTL3 · Apply
CO3	Apply interrupt-driven and timer-driven programming patterns to build periodic and event-driven embedded behaviour without an RTOS.	BTL3 · Apply
CO4	Apply discrete-time PID controller design to a small position or velocity control loop, accounting for sampling rate and saturation at an introductory level.	BTL3 · Apply
CO5	Apply PWM-based motor drive concepts and current sensing to characterise a basic brushless or stepper motor drive scheme.	BTL3 · Apply
CO6	Apply CAN bus fundamentals — frame format, arbitration, transceivers — to design a simple node-to-node telemetry protocol on STM32H7 FDCAN.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	M	–	–	–
CO2	H	–	M	M	M
CO3	H	–	M	–	–
CO4	H	M	M	–	–
CO5	H	–	M	–	–
CO6	H	–	M	M	–

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**STM32 Microcontroller Architecture** Microcontroller vs microprocessor; on-chip integration of memory, peripherals, and connectivity.; ARM Cortex-M family overview and the Cortex-M7 in the STM32H7 series.; STM32H7 memory map: flash, ITCM, DTCM, AXI SRAM, peripheral region.; Clock tree at the user level: HSE, HSI, PLL, system clock, peripheral prescalers.; NVIC interrupt model: priority levels, the system tick, vector table.; Boot sequence: from power-on to main(); the role of the startup file.; STM32CubeMX as the configuration entry point; STM32CubeIDE as the development environment.; A first program: blink an LED on a Nucleo-H743ZI board.	CO1
M2	**GPIO, ADC, Timers, PWM** GPIO at the HAL level: input mode, output mode, alternate function, pull-up/pull-down.; Reading buttons and driving LEDs through the HAL.; STM32 timer subsystems: basic, general-purpose, advanced-control timers.; Timer modes: counting, capture, compare, PWM generation.; PWM configuration through STM32CubeMX: frequency, duty cycle, polarity.; STM32H7 ADC: SAR architecture, channels, resolution, sampling time.; Reading an analog sensor (potentiometer, temperature) at a fixed sample rate.; Common GPIO/ADC/PWM pitfalls and how to debug them.	CO2

	Title & Topics	CO
M3	**Interrupts, Timers, and Periodic Tasks** Polling vs interrupts: when each is appropriate.; NVIC priority and pre-emption.; Writing interrupt service routines: do's and don'ts; keeping ISRs short.; Periodic tasks via timer interrupt: building a 1 kHz tick for a control loop.; External interrupts (EXTI): wiring a button or limit switch to a callback.; DMA fundamentals at the introductory level: offloading data movement from the CPU.; Bare-metal periodic-task pattern: state machine driven by a tick interrupt.; When you would want an RTOS — and why this course stays bare-metal (RTOS depth comes in ESDT).	CO3
M4	**Discrete-Time PID Control** Continuous PID at intuition level: P, I, D — what each term contributes.; Discrete-time PID — the implementable form on a microcontroller.; Sampling rate selection: rule of thumb relative to the closed-loop bandwidth.; Output saturation and anti-windup at introductory level.; Tuning by manual adjustment using step responses.; Implementing a PID loop on a 1 kHz timer tick.; Measuring the closed-loop response and verifying stability.; Cascaded control loops at preview level (position outside, velocity inside).	CO4
M5	**Motor Drive Basics** PWM-based motor drive: duty cycle as voltage proxy.; DC motor drive with an H-bridge; direction and speed control.; Brushless DC motor basics: three-phase windings, six-step commutation.; Stepper motor basics: full-step and half-step modes.; Motor driver ICs: integrated H-bridges and three-phase drivers.; Current sensing through a shunt resistor and op-amp.; Safe operating limits: current, temperature, dead-time.; Pairing the PID loop from M4 with a small motor drive.	CO5
M6	**CAN Bus Fundamentals** Why CAN: dominance in automotive and industrial automation; differential signalling and noise immunity.; CAN physical layer: CAN-H/CAN-L, twisted pair, 120 Ω termination, common bit rates.; CAN frame format: identifier, DLC, data field, CRC, ACK.; Priority by identifier and bus arbitration.; CAN transceivers as the interface between the MCU and the bus.; STM32H7 FDCAN peripheral at the HAL level: configuration, transmit, receive.; A simple node-to-node telemetry exchange: periodic state frames between two STM32 boards.; Higher-level protocols (CANopen, J1939) at awareness level — those are application-domain courses, not part of EBSD.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Mastering STM32**	Carmine Noviello	Leanpub	2023 (revised)
2	**Programming with STM32 — Getting Started with the Nucleo Board and C/C++**	Donald Norris	McGraw-Hill	2018
3	**The Definitive Guide to ARM Cortex-M3 and Cortex-M4 Processors**	Joseph Yiu	Newnes	2014 (3rd Ed.)
4	**Embedded Systems — Real-Time Operating Systems for ARM Cortex-M Microcontrollers**	Jonathan Valvano	CreateSpace	2017
5	**Controller Area Network Prototyping with Arduino**	Wilfried Voss	Copperhill Technologies	2014
6	**Electric Motors and Drives — Fundamentals, Types and Applications**	Austin Hughes & Bill Drury	Newnes	2019 (5th Ed.)

Embedded System Design & IoT

ESDI 26EC2106E

L-T-P-S	3-0-6-0	Credits	4	Contact Hrs/Wk	9
Trimester	T04	Mode	PBL (Project-Based)	Category	ESC
Pre-requisite	FOIS				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze a closed-loop industrial servo as a system — the role of each component (controller, driver, motor, sensor, network) and the data flow that closes the loop from setpoint to torque.	BTL4 · Analyze
CO2	Apply discrete-time PID control to design and tune a position/velocity control loop, accounting for sampling rate, jitter, integral wind-up, and saturation.	BTL3 · Apply
CO3	Apply PWM-based motor drive techniques — duty-cycle modulation, three-phase commutation basics, current sensing via shunt resistors, and motor-driver IC interfacing — to actuate a brushless motor safely.	BTL3 · Apply
CO4	Analyze sensor signal chains for industrial control — incremental and absolute encoders, ADC-based current sensing, signal conditioning, fault detection (encoder loss, overcurrent, overtemperature) — and design defensible interlocks.	BTL4 · Analyze
CO5	Analyze CAN bus communication — physical-layer signalling, frame format, identifier-based priority arbitration, transceiver behaviour — and design a small CAN telemetry protocol between a controller and a host PC.	BTL4 · Analyze
CO6	Apply STM32H7 architecture knowledge — Cortex-M7 core, memory map, NVIC interrupt model, peripheral access via STM32 HAL — to bring up the controller from a blank board and explain at the register level what every HAL call ultimately does.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PSO1
CO1	**H**	**H**	M	–	–	–
CO2	**H**	**H**	**H**	–	–	–
CO3	**H**	–	**H**	–	M	–
CO4	**H**	**H**	M	M	–	–
CO5	**H**	–	**H**	–	M	–
CO6	**H**	–	–	–	**H**	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The Smart Servo as a System** A working closed-loop servo controller running on day one: setpoint in (over CAN), torque out (to motor), telemetry back (over CAN).; Anatomy of an industrial servo system: controller, driver IC, motor, encoder, current sensor, network.; Why industrial actuators dominate the embedded jobs market: automotive ECUs, robotic joints, CNC servos, drone gimbals, industrial conveyors.; The closed-loop control problem at the system level: setpoint, error, plant, feedback; why control loops sit at the heart of every robot, car, and machine tool.; Why the network layer matters: CAN bus is THE industrial bus (automotive, factory automation, aerospace) — its presence on the device is non-negotiable in the field.; The placement landscape for this skill: Bosch, Continental, ZF, Siemens, ABB, Tata Elxsi, Tessolve, Sasken, Wipro PARI — the ₹15–30 LPA industrial-embedded corridor.; A defensible system architecture before any code: block diagram, signal types, latencies, failure modes.; What this course owns vs what comes downstream: EBSD owns the controller, the loop, the network, the actuation; ESDT (T06) goes deep on RTOS and embedded software; FRMS (T07) covers robotics theory; EATM (T07) covers embedded AI.	CO1

	Title & Topics	CO
M2	**Real-Time Control Loops** Continuous-time PID at intuition level: proportional, integral, derivative — what each term contributes and what it costs.; Discrete-time PID — the form actually implemented on a microcontroller; the role of sampling period.; Sampling rate selection: Nyquist, control-loop bandwidth, the rule of thumb (sample 10× the desired closed-loop bandwidth).; Jitter and its impact on stability; the case for putting the control loop on a hardware timer interrupt rather than a polling loop.; Integral wind-up and three standard remedies: clamping, conditional integration, back-calculation.; Output saturation, anti-windup, and bumpless transfer between manual and automatic modes.; PID tuning in practice: Ziegler-Nichols, manual tuning by step response, evidence-based tuning logs.; Cascaded control loops at preview level — position outer loop, velocity inner loop, current innermost loop — the canonical industrial pattern.	CO2
M3	**Actuation — Motor Drive and PWM** PWM fundamentals: frequency, duty cycle, average voltage; choosing a switching frequency above human hearing.; STM32 timer subsystems for PWM generation: advanced-control timers (TIM1/TIM8) with dead-time insertion for power-stage protection.; Brushless DC motor basics: three-phase windings, electrical commutation, why a brushless motor needs an electronic driver.; Six-step (trapezoidal) commutation: the simplest BLDC drive scheme, suitable for our project.; Stepper motors as a simpler alternative: full-stepping, half-stepping, micro-stepping at intuition level.; Motor driver ICs: half-bridges, full H-bridges, dedicated three-phase drivers (e.g., TI DRV8323); why integrated drivers exist.; Current sensing via shunt resistors — low-side and in-line shunt topologies; integration with the STM32H7 ADC.; Safe-operating-area considerations: current limiting, thermal limits, dead-time, brake mode; how to avoid releasing the magic smoke.	CO3
M4	**Sensing and Fault Detection** Position sensing via incremental encoders: A/B quadrature, index pulse, resolution, the STM32 timer in encoder mode.; Absolute encoders at intuition level (single-turn, multi-turn, magnetic, optical) and their place in industrial servos.; Velocity from position: backward differentiation, low-pass filtering, the trade-off between bandwidth and noise.; STM32H7 ADC: SAR architecture, 16-bit resolution, sampling time, multi-channel scan, DMA-driven sampling.; Signal conditioning: voltage dividers, RC low-pass filters, op-amp buffers; oversampling and noise reduction.; Fault detection patterns: encoder loss (no quadrature transitions for N samples), overcurrent (current above limit for K samples), overtemperature (thermistor + ADC).; Watchdog timers — independent watchdog (IWDG) and window watchdog (WWDG) — and why every industrial controller has one.; Designing graceful-degradation behaviour: what does the servo do when a fault is detected? — safe-state shutdown, fault latching, status reporting.	CO4
M5	**CAN Bus Communication** Why CAN: 35-year-old bus that still dominates automotive and industrial automation; differential signalling, robustness to noise, automatic retransmission.; Physical layer: CAN-H, CAN-L, dominant/recessive bits, 120 Ω termination, twisted pair, common bit rates (125 kbit/s, 250 kbit/s, 500 kbit/s, 1 Mbit/s).; CAN frame format: Standard ID (11-bit) and Extended ID (29-bit), DLC, data field (0–8 bytes), CRC, ACK.; Priority by identifier: lowest ID wins arbitration; designing IDs to encode message priority.; Standard frame, remote frame, error frame; error counters and bus-off recovery.; CAN transceivers (e.g., MCP2562, TCAN334) — the bridge between the MCU's CAN controller and the differential bus.; STM32H7 FDCAN peripheral: standard CAN and CAN FD; mailbox-based transmission, FIFO-based reception, filter banks.; A small application protocol on top of CAN — periodic telemetry frames, command frames, heartbeat — the foundations CANopen and J1939 build upon (those higher-level protocols are not covered here).	CO5
M6	**STM32H7 Architecture and HAL — The Foundation Underneath** ARM Cortex-M7 architecture: dual-issue in-order pipeline, FPU, DSP extensions, the AXI bus matrix, tightly-coupled memories (ITCM, DTCM).; STM32H7 memory map: flash, ITCM/DTCM/AXI SRAM, peripheral region; what lives where and why it matters for performance.; Clock tree at the engineer's level: HSE, HSI, PLL, system clock, peripheral prescalers; using STM32CubeMX to configure them.; NVIC interrupt model: 240 sources, 8-bit priority, group/sub-priority, tail-chaining, late arrival.; STM32 HAL philosophy: layered abstraction over registers; when HAL is enough and when register-level access is needed for performance.; GPIO, ADC, timers, FDCAN — the peripherals we used throughout the project — at the register level: tracing one HAL_TIM_PWM_Start call all the way to the timer registers it touches.; Boot, vector table, startup file; what happens between power-on and main().; Tooling: STM32CubeIDE, STM32CubeMX, ST-LINK debugger, the build-flash-debug cycle on a Nucleo-H743ZI board.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Mastering STM32**	Carmine Noviello	Leanpub	2023 (revised)
2	**The Definitive Guide to ARM Cortex-M3 and Cortex-M4 Processors**	Joseph Yiu	Newnes	2014 (3rd Ed.)

#	Title	Authors	Publisher	Year
3	**Embedded Systems Architecture — Explore Architectural Concepts and Design Patterns**	Daniele Lacamera	Packt	2023 (2nd Ed.)
4	**Modern Control Engineering**	Katsuhiko Ogata	Pearson	2010 (5th Ed.)
5	**CAN System Engineering — From Theory to Practical Applications**	Wolfhard Lawrenz	Springer	2013 (2nd Ed.)
6	**Electric Motors and Drives — Fundamentals, Types and Applications**	Austin Hughes & Bill Drury	Newnes	2019 (5th Ed.)
7	**Controller Area Network Prototyping with Arduino**	Wilfried Voss	Copperhill Technologies	2014

Foreign Language

FLNG FLP

L-T-P-S	0-0-4-0	Credits	2	Contact Hrs/Wk	4
Semester	S01	Mode	Regular	Category	HSS
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the script, sounds, and certification format of the chosen language.	BTL2 · Understand
CO2	Apply foundational grammar and politeness levels to form correct sentences.	BTL3 · Apply
CO3	Apply everyday and workplace vocabulary across common situations.	BTL3 · Apply
CO4	Apply listening and speaking for daily and business interactions.	BTL3 · Apply
CO5	Apply business reading and writing for emails, forms, and certification tasks.	BTL3 · Apply
CO6	Analyze the country's culture and business etiquette, and pass a basic-cert mock.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO6	PO9	PO10	PO11	PSO1	PSO2
CO1	–	–	H	H	M	–
CO2	–	–	H	H	M	–
CO3	–	–	H	M	M	–
CO4	–	H	H	–	M	–
CO5	–	–	H	H	M	–
CO6	H	–	H	–	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Script, Sounds & First Contact** Writing system — German alphabet / Japanese hiragana & katakana / Korean Hangul.; Sounds, pronunciation, and reading aloud.; Greetings, farewells, and politeness formulas.; Introducing yourself and others.; Personal details — asking and giving.; Numbers, dates, and time.; Classroom and survival language.; Basic-certification format overview (JLPT N5 / Goethe A1 / TOPIK I). **Labs:** Script + pronunciation lab with recorded self-assessment.; Self-introduction role-play in a first-day workplace setting. **Case Study:** Case Study — a graduate's first day at a German/Japanese/Korean company: analyse the greetings and self-introduction they must handle in-country.	CO1
M2	**Core Grammar & Politeness Foundations** Nouns, articles/particles, and word order.; Personal pronouns and addressing people.; Present-tense verbs and conjugation.; Essential irregular verbs (to be / to have / to do).; Negation and yes/no & wh-questions.; Politeness levels — Sie/du, -masu/desu, jondaemal.; Plurals/counters and basic particles.; Common beginner errors for Indian learners. **Labs:** Grammar-in-use lab building polite simple sentences about yourself.; Register lab: convert casual sentences to workplace-appropriate forms. **Case Study:** Case Study — formal vs informal register at work: analyse a situation where the wrong politeness level caused offence, and the correct form.	CO2

	Title & Topics	CO
M3	**Everyday & Workplace Vocabulary** Family, people, and relationships.; Food, dining, and shopping.; Time, routine, and schedules.; Directions, transport, and places.; Office, departments, and roles.; Meetings, email, and telephone basics.; Numbers, prices, and money.; Certification core vocabulary lists. **Labs:** Themed-vocabulary lab (everyday + office) with spaced repetition.; Role-play lab: ordering, directions, and a workplace introduction. **Case Study:** Case Study — settling in abroad: a relocated employee handles shopping, transport, and an office introduction — identify the vocabulary each needs.	CO3
M4	**Listening & Speaking** Understanding slow, clear speech.; Everyday conversations and small talk.; Self-introduction in a business setting.; Exchanging contact details and business cards.; Telephone and simple meeting expressions.; Making and responding to requests.; Expressing likes, needs, and preferences.; Certification listening-section practice. **Labs:** Listening lab on basic-cert dialogues with comprehension tasks.; Paired speaking lab: business self-introduction + card exchange. **Case Study:** Case Study — a first client phone/intro call: analyse a simple business-contact exchange in-country and the listening cues that matter.	CO4
M5	**Business Reading & Writing** Reading short texts, signs, and notices.; Reading simple emails and messages.; Writing short professional messages and emails.; Filling forms and profiles.; Tone and politeness in writing.; Connecting words and simple sentences.; Certification reading-section practice.; Certification writing-section practice (where applicable). **Labs:** Reading-comprehension lab on basic-cert texts.; Write-and-revise lab: a short professional email, then critique with/beyond an AI tool. **Case Study:** Case Study — a relocation form / first work email: decode a real beginner-level document and draft an appropriate short reply.	CO5
M6	**Culture, Etiquette & Certification Readiness** Cultural values and communication style of the country.; Business etiquette — greetings, bowing/handshake, business cards.; Hierarchy, seniority, and addressing colleagues.; Punctuality, meetings, and workplace norms.; Dining and after-work culture (Feierabend / nomikai / hoesik).; Do's and don'ts and avoiding offence.; Basic-certification full mock test and strategy.; Continued-learning routine with AI tools. **Labs:** Business-etiquette simulation (greetings, cards, seating) for the chosen country.; Full basic-certification mock test with scoring and review. **Case Study:** Case Study — an etiquette misstep by a foreign employee (mishandling a Japanese business card / over-familiarity in Germany / ignoring Korean seniority): analyse what went wrong and the correct etiquette.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Genki I: An Integrated Course in Elementary Japanese (Textbook), 3rd Ed.**	Eri Banno, Yoko Ikeda, Yutaka Ohno, Chikako Shinagawa & Kyoko Tokashiki	The Japan Times	2020
2	**TRY! JLPT N5: Japanese Language Proficiency Test**	ASK Publishing	ASK Corporation	2014
3	**Netzwerk neu A1: Deutsch als Fremdsprache (Kurs- und Übungsbuch)**	Stefanie Dengler, Tanja Mayr-Sieber, Paul Rusch & Helen Schmitz	Klett	2019
4	**Fit fürs Goethe-Zertifikat A1: Start Deutsch 1**	Evelyn Frey	Hueber Verlag	2019
5	**Integrated Korean: Beginning 1, 3rd Ed.**	Young-mee Cho, Hyo Sang Lee, Carol Schulz, Ho-min Sohn & Sung-Ock Sohn	University of Hawai'i Press	2019
6	**Korean Grammar in Use: Beginning**	Ahn Jean-myung, Lee Kyung-ah & Han Hoo-young	Darakwon	2010
7	**The Culture Map: Breaking Through the Invisible Boundaries of Global Business**	Erin Meyer	PublicAffairs	2014

Foundations of Robotics & Mechatronic Systems

RBFN 26EC3112E

L-T-P-S	3-0-0-6	Credits	3	Contact Hrs/Wk	9
Trimester	T07	Mode	PBL (Project-Based)	Category	PCC
Pre-requisite	EBSD				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze a mobile robot as a layered system — task / navigation goal, perception, localisation, kinematics, actuation, mechatronic hardware — and trace one waypoint-following command from publisher to wheel motion.	BTL4 · Analyze
CO2	Apply mechatronic hardware reasoning — DC and BLDC motors with encoders, motor drivers, lidar, IMU, the power-electronics chain — to integrate a robot platform from components.	BTL3 · Apply
CO3	Apply kinematic models — forward and inverse kinematics for differential drive, an introduction to manipulator forward/inverse kinematics, the rigid-body transform machinery — to relate joint or wheel motion to end-effector or robot motion.	BTL3 · Apply
CO4	Apply the ROS 2 framework — nodes, topics, services, actions, parameters, launch files, the URDF robot model, the tf2 transform tree — to build modular, debuggable robot software systems.	BTL3 · Apply
CO5	Apply the sensing-and-localisation foundations — lidar scan interpretation, IMU integration, wheel odometry, an introduction to particle filtering and adaptive Monte Carlo localisation (AMCL) — to estimate robot pose in a known map.	BTL3 · Apply
CO6	Apply closed-loop robot motion control at the foundational level — wheel-velocity PID, basic waypoint following with pure pursuit, simple bump-and-go obstacle avoidance — recognising that perception at depth is the territory of RPCS and motion planning at depth is RMNC.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	H	H	–	H
CO2	H	–	H	M	–
CO3	H	H	–	–	H
CO4	H	–	H	H	H
CO5	H	–	H	–	H
CO6	H	H	H	–	–

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The Robot as a System** A working KartBot navigating an arena on day one — students teleoperate the robot, then watch it follow a waypoint, before any internals.; The canonical mobile-robot stack: navigation goal → planner → controller → kinematics → actuator → physical wheels; sensing closes the loop back to localisation.; The categories of robots: mobile (wheeled, legged, drone), manipulators (arms), humanoids, swarm; what changes per category.; Why ROS 2: the discipline of a publish-subscribe robot framework; the alternative landscape (bare-coding-from-scratch, ROS 1, vendor SDKs) and why ROS 2 is now the standard.; Tracing one /goal_pose command end-to-end through KartBot: ROS 2 publisher → Nav2 planner → controller → wheel-velocity command → STM32 firmware over serial → motor drivers → wheels → encoder feedback → odometry → tf tree.; A reading of the KartBot codebase: the URDF, the launch files, the controller node, the firmware's ROS-2-aware serial protocol.; The course's scope vs the downstream chain: FRMS owns foundations; RPCS owns perception at depth; RMNC owns motion planning at depth; ARHR owns autonomy.; Safety as the first-class concern: every robot project starts with the e-stop, the speed limit, the supervisor.	CO1

	Title & Topics	CO
M2	**Mechatronic Hardware** DC motors revisited from EBSD at the robot scale: hub motors, gearmotors, the speed-torque curves; choosing motors for a robot platform.; BLDC at the robot scale: when you graduate from brushed to brushless; the price-performance crossover.; Encoders revisited: incremental for velocity, absolute for joint position; encoder resolution for odometry quality.; Motor drivers at the robot scale: H-bridge for brushed, three-phase driver IC for brushless; the EBSD primitives now at robot current levels.; Lidar at intuition level: 2D scanning lidar (RPLIDAR, YDLIDAR), 3D lidar at preview level, the time-of-flight principle.; IMU: 6-DoF (gyro + accelerometer), 9-DoF (+ magnetometer); the noise, bias, and drift characteristics; sensor fusion preview.; Power electronics for robots: battery chemistry (LiPo, Li-ion 18650), BMS, regulators, the wiring discipline that keeps lethal current safe.; Mechanical considerations at intuition level: the chassis, the bearings, the wheel-rubber compound, the centre-of-gravity placement; what an embedded engineer must know about the mechanical side.	CO2
M3	**Kinematics** Rigid-body transformations: rotation matrices, translations, homogeneous coordinates; the SE(3) and SE(2) groups at intuition level.; Coordinate frames: world frame, robot frame, sensor frame; the tf2 transform tree as the practical implementation.; Quaternions vs Euler angles: why quaternions; the gimbal-lock problem.; Differential-drive forward kinematics: given wheel velocities, predict robot motion; the unicycle model.; Differential-drive inverse kinematics: given a desired robot motion, compute wheel velocities; the curvature-and-speed decomposition.; Manipulator kinematics preview: links, joints, DH parameters at intuition level, forward kinematics as a chain of transforms.; Manipulator inverse kinematics preview: the analytical vs numerical approaches; why IK is hard in general.; Singularities and joint limits: the practical pitfalls that kinematics-aware engineers avoid.	CO3
M4	**The ROS 2 Framework** ROS 2 architecture: nodes, the DDS-based middleware, the discovery mechanism; the contrast with ROS 1.; Communication primitives: topics (publish-subscribe), services (request-response), actions (long-running with feedback); when each is the right fit.; Quality of Service (QoS) in ROS 2: reliability, durability, deadline; matching publishers and subscribers correctly.; Parameters: per-node configuration, dynamic reconfiguration; the ROS launch system.; URDF: describing a robot in XML; visual, collision, inertial properties; the tf tree built automatically.; Xacro: parameterising URDF for reuse and readability.; Launch files in Python: composing nodes, namespacing, conditional launching.; The ROS 2 toolchain: rviz2 for visualisation, rqt for inspection, ros2 bag for record-and-replay; debugging discipline in a multi-node system.	CO4
M5	**Sensing and Localisation Foundations** Lidar scan interpretation: range and angle arrays; the LaserScan message; filtering invalid returns.; Occupancy grid maps: the cell-based world representation; building one offline from a recorded bag.; IMU integration at intro level: gyro to angular velocity, accelerometer to attitude (with magnetometer help); why pure-IMU dead reckoning drifts.; Wheel odometry from differential drive: integrating wheel velocities to pose; the accumulating-error problem.; Sensor-fusion preview: the complementary filter, the extended Kalman filter at intuition level; the proper coverage lives in RPCS.; AMCL (Adaptive Monte Carlo Localisation) at intuition level: the particle filter, the importance-sampling step, the resampling step; using ROS 2's Nav2 AMCL implementation.; The map-frame / odom-frame / base-link-frame discipline in tf2; the standard ROS conventions that prevent confusion.; Limitations of foundational localisation: when AMCL fails (kidnapped robot, featureless environments); these limitations are what RPCS will address.	CO5
M6	**Closed-Loop Robot Control — At the Foundation** Wheel-velocity PID at the robot scale: the same PID as EBSD, now for each wheel; cascaded with a vehicle-velocity outer loop.; Pure pursuit for waypoint following: the lookahead-distance algorithm; geometric intuition; the gold-standard simple controller.; Stanley controller as the alternative for path tracking; cross-track-error feedback.; Simple obstacle avoidance: bump-and-go (turn until clear), VFH (Vector Field Histogram) at intuition level — basic only.; The foundational nav stack: a path is given (we don't plan it here), follow it; bump-and-stop on obstacle; the limitations.; Recognising the boundaries: full path planning (A*, RRT, MPC), avoidance under uncertainty, social navigation — these are RMNC territory.; Recognising the perception boundaries: SLAM, visual-inertial odometry, semantic segmentation for navigation — these are RPCS territory.; Bringing it together: KartBot following waypoints in a known map, avoiding bumps, demonstrating the layered system end-to-end. The downstream courses make this dramatically more capable.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Modern Robotics — Mechanics, Planning, and Control**	Kevin M. Lynch & Frank C. Park	Cambridge University Press	2017
2	**Probabilistic Robotics**	Sebastian Thrun, Wolfram Burgard & Dieter Fox	MIT Press	2005

#	Title	Authors	Publisher	Year
3	**Introduction to Autonomous Mobile Robots**	Roland Siegwart, Illah R. Nourbakhsh & Davide Scaramuzza	MIT Press	2011 (2nd Ed.)
4	**A Concise Introduction to Robot Programming with ROS 2**	Francisco Martín Rico	CRC Press	2024
5	**Robot Operating System (ROS) for Absolute Beginners — Robotics Programming Made Easy**	Lentin Joseph & Aleena Johny	Apress	2022 (2nd Ed.)
6	**Robotics, Vision and Control — Fundamental Algorithms in Python**	Peter Corke	Springer	2023 (3rd Ed.)
7	**Springer Handbook of Robotics**	Bruno Siciliano & Oussama Khatib (eds.)	Springer	2016 (2nd Ed.)

Full Stack Web Development

FSWD REG · 26SC1308 PBL · 26SC1308E

L-T-P-S	2-0-4-0 / 3-0-6-0	Credits	4	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S03 · T03	Mode	Regular & PBL	Category	ESC
Pre-requisite	PSPP				

Course Outcomes (COs)

	Statement	BTL
CO1	Explain how modern full-stack web applications work and how AI-assisted development fits in, and use AI builder tools (Claude Code, Lovable) to produce a first working interface.	BTL3 · Apply
CO2	Use just-enough HTML, CSS and JavaScript to read, understand, direct and customise frontend code, including responsive design and components.	BTL3 · Apply
CO3	Build a real, functional frontend using AI tools - prompting, iterating, integrating and customising generated interfaces.	BTL6 · Create
CO4	Explain how backend engineering works - APIs, data and CRUD - through FastAPI with PostgreSQL and MongoDB.	BTL3 · Apply
CO5	Build a backend with AI tools and integrate it with the frontend into a working full-stack application, including basic authentication.	BTL6 · Create
CO6	Build and deploy a functional, enterprise-level web application using contemporary frameworks and AI tools, with basic security.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO8	PO9	PO11
CO1	M	–	M	**H**	–	–	M
CO2	M	–	M	M	–	–	–
CO3	M	–	**H**	**H**	–	–	–
CO4	M	M	–	M	–	–	–
CO5	M	–	**H**	**H**	–	–	–
CO6	M	–	**H**	**H**	M	M	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - The Modern Web & AI-Assisted Development** The north star (Outside-In): the functional, enterprise-level application you will build for your own field, fast, with contemporary tools.; How the modern web works at a working level: client, server, database, and how they connect (just-enough, not exhaustive).; The anatomy of a full-stack app and the contemporary stack you will use (a framework front-end; FastAPI + PostgreSQL/MongoDB back-end).; The AI-assisted development workflow: building with AI tools as the default, not the exception.; AI builder/coding tools: Claude Code (Anthropic's agentic coding tool, used from the command line/desktop/mobile) and Lovable (AI app builder) - what each does.; Effective prompting for building software: being specific, giving context, iterating.; The essential discipline: understanding, reading and validating what the tools produce - directing, not depending.; What you can realistically build fast with these tools (and what still needs care).; Setting up the tools, an editor and version control (Git/GitHub).; Producing a first working interface with an AI tool and choosing the project domain.	CO1

	Title & Topics	CO
M2	**Module 2 - Frontend Foundations, Fast-Tracked** Just-enough HTML: structure and the elements you must recognise to read and edit a page.; Just-enough CSS: styling, layout (flex/grid) and responsive design at a practical level.; Just-enough JavaScript (building on the programming course): variables, functions, events and updating the page - enough to understand and modify.; The component idea and modern frameworks (e.g., React) at a reading level - what components and state are and why they exist.; Reading generated frontend code: making sense of what an AI tool produces.; Customising generated code: changing styling, content and behaviour with confidence.; Connecting a frontend to data: what an API call looks like (fetch) - enough to follow it.; Using component libraries / CSS frameworks for a polished look quickly.; Debugging the frontend with browser developer tools.; Applying just-enough foundations to read and customise the project's AI-built interface.	CO2
M3	**Module 3 - Building Frontends with AI Tools** Building a functional frontend with AI tools (Claude Code, Lovable): from a description to a working interface.; Iterative prompting: refining the UI by conversation - changing layout, components and behaviour.; Integrating the pieces: assembling pages, navigation and state into a coherent app.; Customising and extending generated frontends with the just-enough foundations from Module 2.; Connecting the frontend to data/APIs (calling a backend) with AI assistance.; Handling forms, user input and dynamic data display in an AI-built frontend.; Reviewing and validating generated frontend code: does it do what you need, and is it sound?; When the AI gets it wrong: debugging and correcting generated code.; Good practice: keeping the project organised and version-controlled while building with AI.; Building the project's functional frontend with AI tools.	CO3
M4	**Module 4 - Understanding Backend Engineering (FastAPI, PostgreSQL, MongoDB)** Why a backend: logic, data and security that cannot live in the browser - what the backend is for.; FastAPI (Python): how a modern API framework works - endpoints, requests/responses, JSON, automatic docs.; REST APIs at a working level: the endpoints a frontend calls and what they return.; Relational data with PostgreSQL: tables, keys and basic SQL - how structured data is stored and queried.; Document data with MongoDB: collections and documents - how flexible data is stored.; When to use relational vs document storage (at a practical, decision level).; CRUD: how create/read/update/delete flows from the API to the database.; Connecting a backend to a database (the idea, with AI assistance).; Reading and understanding backend code - enough to direct and validate it.; Understanding the backend the project's frontend will need.	CO4
M5	**Module 5 - AI-Assisted Backend & Full-Stack Integration** Building a FastAPI backend with AI tools: scaffolding endpoints and data models by prompting.; Setting up the database (PostgreSQL or MongoDB) and connecting the backend with AI assistance.; Implementing CRUD for the app's data with AI help, then validating it works.; Connecting the AI-built frontend to the AI-built backend: the full-stack flow.; Handling cross-origin requests (CORS) so the pieces work together.; Basic authentication: a simple login, and why passwords must be hashed (basic, defensive).; Reviewing and validating the generated backend: testing the endpoints and checking the data.; Debugging across the stack when something does not work.; Keeping the integrated app organised and in version control.; Integrating the project into a working full-stack application.	CO5
M6	**Module 6 - Building & Deploying a Functional Enterprise-Level Application** What makes an application 'enterprise-level' at a functional level: real data, multiple features, users, reliability.; Assembling the full application from contemporary frameworks and AI tools into a coherent product.; Adding the features that make it functional and useful for the student's domain.; Basic security and good-practice habits (validating input, not exposing secrets) - defensive.; Configuration for production vs development.; Deployment: getting the full application online on a modern hosting platform (with AI assistance).; Testing that the application works end to end.; Validating and polishing the AI-built application - making it genuinely usable.; Documenting the application and the repository.; Where this leads: building functional applications fast is a durable, cross-domain skill for the student's field and beyond.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**The Road to React**	Robin Wieruch	Leanpub	2024
2	**FastAPI (Modern Python Web Development)**	Bill Lubanovic	O'Reilly	2023
3	**Learning Web Design**	Jennifer Robbins	O'Reilly	2018 (5th ed.)
4	**Full Stack open (course book, free)**	University of Helsinki	fullstackopen.com	2024
5	**AI-assisted development guides (Anthropic Claude Code documentation)**	Anthropic	Anthropic (online)	2025

Fundamentals of IoT & Sensors

FOIS REG · 26EC1101 PBL · 26EC1101E

L-T-P-S	2-0-4-0 / 3-3-0-0	Credits	4	Contact Hrs/Wk	6
Term (Reg · PBL)	S02 · T01	Mode	Regular & PBL	Category	ESC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Explain what embedded systems and the IoT are, describe the sense-think-act-connect architecture, and read component specifications/datasheets well enough to select and plan a build - treating all hardware as plug-and-play black boxes.	BTL3 · Apply
CO2	Identify and select sensors for a sensing need, interpret their outputs at a usage level (digital vs analog, range, resolution, calibration), and reason about placement and reliability - without circuit analysis.	BTL4 · Analyze
CO3	Use a microcontroller development board as the 'brain': identify its pins and interfaces from the pinout, read inputs and drive outputs, use PWM and a display, and connect a module over a standard interface (I2C/SPI/UART) at a usage level.	BTL4 · Analyze
CO4	Select and integrate actuators and small mechanical components to make a system move and act - motors, servos, drivers, wheels, gears, and simple mechanisms - reasoning about motion, mounting and power at a usage level, as the first step toward robotics.	BTL6 · Create
CO5	Connect an embedded device to the internet and exchange data: use wired/wireless links and IoT protocols at a usage level, publish sensor data to a cloud dashboard, and receive remote commands - turning a device into an IoT product.	BTL4 · Analyze
CO6	Integrate sensing, control logic, actuation, mechanics and connectivity into a complete, reliable sense-think-act system; apply simple decision/feedback logic, debug an embedded build, and package it - establishing the foundation for robotics.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO8	PO9	PO11
CO1	M	M	M	L	M	M	–	–	M
CO2	M	M	M	L	**H**	–	–	–	M
CO3	M	M	**H**	M	**H**	–	–	–	M
CO4	M	M	M	M	**H**	–	–	–	M
CO5	M	M	**H**	M	**H**	–	–	–	M
CO6	M	M	**H**	M	**H**	M	M	M	**H**

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - The Embedded & IoT Landscape and the Art of Reading Specs** What an embedded system is (a computer built into a product) and how it differs from a general-purpose computer - conceptually, no internals.; The IoT 'big picture': the sense → think → act → connect → cloud architecture and where each part lives in a real product.; Anatomy of an IoT/embedded product as building blocks: sensors, controller board, actuators, mechanical parts, power, connectivity, cloud/app - each as a black box defined by its spec.; The plug-and-play philosophy: modules, breakout boards, and 'shields/HATs' as pre-built blocks you connect rather than design.; How to read a datasheet/specification WITHOUT electronics theory: identifying operating voltage, the kind of signal a part needs/produces (digital vs analog at a usage level), current/power notes, pin labels, and the communication 'language' a part speaks (e.g., I2C/SPI/UART named, not analysed).; Interpreting connection diagrams, pinout cards and 'how to wire' guides; matching a sensor's voltage/logic level to a board (compatibility checking by specification).; Units and quantities a builder must recognise (volts, milliamps, the meaning of a sampling rate or range on a sensor spec) - read, not derived.; Where the chip internals live (and that they are intentionally out of scope here - the partner DDCA course covers them); the black-box contract a datasheet represents.; From idea to block diagram: turning a product idea into a labelled block diagram and a bill of materials chosen from specs.; Safety, static handling, and good build practices for beginners (do/don't list) - practical hygiene, not theory.	CO1

	Title & Topics	CO
M2	**Module 2 - Sensors: Types, Specifications & Sensing Concepts** What a sensor does conceptually: turning a physical quantity into a signal a controller can read.; Families of sensors a builder meets: environmental (temperature, humidity, light, gas/air-quality), distance/proximity (ultrasonic, IR), motion/presence (PIR), orientation/motion (accelerometer/gyro module), touch/input - described by what they measure and output, not by their physics.; Digital vs analog sensor outputs at a usage level: a digital sensor gives a ready value/level; an analog sensor gives a varying level the board converts to a number (the idea of reading an analog value, not the electronics of conversion).; Reading a sensor spec that matters to a build: measurement range, resolution/precision, response time/sampling rate, operating voltage, and the interface it uses (direct pin vs I2C/SPI module).; Modules vs raw sensors: why beginners use breakout modules (regulation and connections pre-done) - plug-and-play sensing.; Calibration and the difference between a raw reading and a meaningful value (offset/scaling at a conceptual level; using a known reference).; Noise, drift and unreliable readings - what they look like and beginner strategies to handle them (averaging readings, placement, debouncing an input) conceptually.; Sensor selection reasoning: matching a sensing requirement (what, how far, how fast, how precise, indoor/outdoor) to an appropriate sensor from specs.; Placement and environment: how mounting, orientation and surroundings affect a sensor (e.g., ultrasonic and soft surfaces, light sensors and shadows).; Multiple sensors together: the idea of combining readings (sensor fusion at an intuitive level) toward a robot that perceives its surroundings.	CO2
M3	**Module 3 - The Controller Board: I/O, Interfaces & Talking to Modules** The development board as the 'brain' of a build (controller + USB + pins + power), introduced as a black box with a pinout - no architecture internals (those are DDCA's).; Reading a board pinout/cheat-sheet: power pins, ground, general-purpose I/O pins, which pins are analog-capable, which support PWM, and which are the I2C/SPI/UART pins.; Digital input and output at a usage level: a pin can read a HIGH/LOW (button, digital sensor) or drive a HIGH/LOW (LED, buzzer, relay module).; Analog input at a usage level: reading a varying value from an analog sensor into a number the program can use (range/levels conceptually).; PWM as 'how to dim, set brightness, set a speed or a servo angle' - the builder's mental model of a proportional output (no signal theory).; Output devices as plug-and-play blocks: LED, buzzer, relay module, and a small display (OLED/LCD) for showing readings.; Communication interfaces as 'languages modules speak': I2C, SPI, UART named and used at the level of 'connect these labelled pins and use the library' - addresses/buses conceptually, not electrically.; Libraries and example sketches: the plug-and-play software side - installing a device library, opening its example, and adapting it to your pins/values.; Powering a build sensibly: USB vs battery, the idea that motors need their own power, and not exceeding a board's limits (from spec, as a rule of thumb).; The build loop: connect by pinout → load/adapt example → read the serial output → verify - the beginner's debugging cycle.	CO3
M4	**Module 4 - Making Things Move: Actuators & Small Mechanical Components** Actuators as the 'act' in sense-think-act: turning a controller decision into physical action.; Types of small motors a builder meets, by behaviour and use: the hobby servo (precise angle, e.g., a sensor sweep or gripper), the DC gear motor (continuous drive for wheels), and the stepper (precise steps) - chosen by what motion you need, from spec.; Why a motor needs a driver module (e.g., an H-bridge/motor-driver board) and its own power - introduced purely as a plug-and-play block with a wiring card, not as electronics.; Controlling motion at a usage level: setting a servo angle, setting motor direction and speed (via PWM), and starting/stopping - through library calls.; Small mechanical components and what they do: wheels and a chassis, gears and gear-ratio intuition (trading speed for turning force), levers/linkages, a simple gripper, mounts and brackets, fasteners.; Motion concepts for a beginner robot: driving straight, turning (differential drive intuition - two wheels at different speeds), and sweeping a sensor to 'look around'.; Mechanical assembly practicalities: mounting motors and sensors firmly, alignment, balance, and why loose mechanics ruin a build.; Power for motion: separate motor power vs controller power, common-ground idea (as a connection rule), and battery choice from spec/run-time needs.; Matching actuator to job: torque vs speed intuition, weight the device can carry/move, and reading the few spec numbers that matter (stall/torque/speed/voltage) as selection guidance.; From mechanism to robot: how sensing + a controller + actuators + structure compose into a machine that moves with purpose - the bridge to the robotics courses.	CO4

	Title & Topics	CO
M5	**Module 5 - Connectivity & the 'Internet' of Things** Why and when a device needs connectivity: local vs remote monitoring and control; the 'Things' becoming 'Internet of Things'.; Wired vs wireless links at a usage level: USB/serial, and wireless Wi-Fi and Bluetooth/BLE - what each is good for (range, power, setup), chosen by need not by radio theory.; The connected-board idea: boards with built-in Wi-Fi/BLE (e.g., ESP32-class) so connectivity is plug-and-play in software.; IoT messaging at a usage level: the request/response idea (HTTP) and the publish/subscribe idea (MQTT) - explained as 'how devices send and receive messages', with a broker/topic mental model.; Cloud dashboards and IoT platforms: sending sensor readings to a dashboard to chart them, and defining a control (button/slider) that sends a command back - using a free IoT platform's library and tokens.; Data formats a builder sees: simple key-value/JSON-style messages (read and adapt, not parse formally).; Telemetry and remote control end-to-end: device → cloud (publish readings) and cloud/app → device (receive command) - the full IoT round trip.; Connectivity reliability for beginners: handling a dropped Wi-Fi connection, reconnecting, and why timing/credentials matter (practical, from examples).; Security hygiene at a beginner level: don't hard-code secrets publicly, use device tokens, change default passwords - good practice, not cryptography (the maths of which lives in Mathematics for Computation).; Privacy and responsible IoT: what data a device collects and why minimal, consented collection matters - conceptual awareness.	CO5
M6	**Module 6 - Integration, Control Logic & the Path to Robotics** The sense-think-act loop as the heart of every embedded/robotic system: read sensors → decide → drive actuators → repeat.; Simple decision logic for autonomy: threshold rules ('if distance < X, stop/turn'), states (idle/driving/avoiding), and combining several sensor inputs into one decision (no control theory - intuition only).; The idea of feedback: using a sensor reading to adjust an action (e.g., slow down as an obstacle nears) - conceptual closed-loop intuition that later robotics control courses formalise.; System integration: bringing sensors, controller I/O, motors/mechanism and connectivity together so they cooperate (timing, sharing the controller's attention, avoiding conflicts).; Power and reliability of a complete build: budgeting battery life, separating motor and logic power, brown-out symptoms and beginner fixes (from spec/rules of thumb).; Enclosure, mounting and 'productising': protecting the electronics, cable management, and making a build robust enough to move without falling apart.; Debugging an embedded system methodically: isolate (sensor? wiring-by-pinout? library? power? mechanics?), use serial prints to observe, change one thing at a time.; Testing a build against its requirements: defining what 'working' means and checking each behaviour (acceptance checklist).; Documenting a build so others can reproduce it: block diagram, connection card, bill of materials, and a short build guide.; From this device to a robot: how the integrated sense-think-act node maps directly onto a mobile robot, and what the robotics courses add next (perception, motion planning, control).	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Getting Started with Arduino**	Massimo Banzi & Michael Shiloh	Make: Community (Maker Media)	2022 (4th ed.)
2	**Programming Arduino: Getting Started with Sketches**	Simon Monk	McGraw-Hill / TAB	2022 (3rd ed.)
3	**Exploring Arduino: Tools and Techniques for Engineering Wizardry**	Jeremy Blum	Wiley	2019 (2nd ed.)
4	**Make: Sensors**	Tero Karvinen, Kimmo Karvinen & Ville Valtokari	Make: Community	2014
5	**Designing the Internet of Things**	Adrian McEwen & Hakim Cassimally	Wiley	2013
6	**Make: Getting Started with the ESP32 / Internet of Things projects (Espressif/ESP32 maker guides)**	Various (e.g., Marco Schwartz; Espressif/Random Nerd Tutorials community guides)	Make: Community / community resources	2021-2023

Generative AI & AI Platform Engineering

GNAI REG · 26AD2202 PBL · 26AD2202E

L-T-P-S	2-0-4-0 / 3-0-6-0	Credits	4	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S06 · T05	Mode	Regular & PBL	Category	PCC
Pre-requisite	MLRN				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze a generative-AI platform as a layered system — user interface, orchestration, retrieval, model serving, foundation model, observability, safety — and trace one user query through every layer.	BTL4 · Analyze
CO2	Apply foundational generative-model concepts at intuition level — autoregressive generation, attention mechanism, the transformer block, decoder-only vs encoder-only vs encoder-decoder architectures, scaling laws — to reason about model capabilities and limits.	BTL3 · Apply
CO3	Apply embedding models and vector stores to design retrieval-augmented generation (RAG) systems — chunking strategies, dense and hybrid retrieval, approximate nearest-neighbour indexes (HNSW, IVF), reranking — and reason about retrieval quality.	BTL3 · Apply
CO4	Analyze prompt engineering and in-context learning — zero-shot vs few-shot, chain-of-thought, structured output via JSON-mode and function-calling, prompt-template design, prompt evaluation — and apply systematic improvement workflows.	BTL4 · Analyze
CO5	Apply parameter-efficient fine-tuning techniques (LoRA, QLoRA), instruction-tuning datasets, and the RLHF / DPO post-training pipeline at intuition level — and execute a small fine-tuning run on a domain-specific dataset.	BTL3 · Apply
CO6	Apply AI-platform engineering practices — orchestration frameworks (LangChain, LlamaIndex), observability for LLM systems (LangSmith, Helicone, OpenTelemetry), cost management, latency optimisation, safety guardrails — to deploy a production-style generative-AI service.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PSO1
CO1	H	H	H	–	–	H
CO2	H	H	–	–	–	H
CO3	H	–	H	–	H	H
CO4	H	H	–	–	–	H
CO5	H	–	H	–	–	H
CO6	H	–	H	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The Generative AI Platform as a System** A working document-aware assistant on day one — students chat with AskKLU and observe RAG behaviour, tool calls, structured output before any internals.; The canonical generative-AI platform stack: UI → orchestrator (LangChain/LlamaIndex) → prompt template → vector retrieval → LLM API → response post-processing → observability/cost tracking.; The four common generative-AI product patterns: chat with RAG, structured-output extraction, function-calling tool use, agentic multi-step workflows.; The foundation-model landscape: closed-source (OpenAI GPT, Anthropic Claude, Google Gemini), open-source (Llama, Mistral, Qwen, Gemma); India-origin models (Sarvam, Krutrim) at preview level.; Tokens as the unit of cost and rate-limit: the tokeniser, input vs output tokens, the practical cost arithmetic.; Tracing one /ask query end-to-end: query → embedding → vector search → context assembly → prompt fill → API call → response → safety check → return to user.; A reading of the AskKLU codebase: the UI, the FastAPI handlers, the prompt templates, the retriever, the LLM client, the observability hooks.; What GALE owns vs what comes downstream: platform engineering here; transformer math at depth in NLPL; GPU training in DLAC; agentic architecture in AAAE.	CO1

	Title & Topics	CO
M2	**Foundations of Generative Models** Autoregressive generation: predicting the next token given the prefix; sampling strategies (temperature, top-k, top-p/nucleus, beam search).; Tokenisation at intuition level: byte-pair encoding (BPE), WordPiece, SentencePiece; the practical implications for non-English text.; The attention mechanism at intuition level: query-key-value, softmax-over-dot-products, why attention is the central innovation. (Full math depth lives in NLPL.); The transformer block: self-attention + feed-forward + residuals + layer-norm; stacking blocks to build a model.; Decoder-only vs encoder-only vs encoder-decoder: GPT-family, BERT-family, T5-family; which architecture for which job.; Scaling laws at intuition level (Chinchilla): model size, data size, compute — the proportional relationship.; Capabilities and limits: where foundation models excel (language fluency, recall of training data, zero-shot generalisation), where they fail (faithful arithmetic, knowledge cutoffs, hallucination).; The pre-training / fine-tuning / RLHF / RAG pyramid: how a useful product is composed from these layers.	CO2
M3	**Embeddings and Vector Stores — The RAG Pattern** Dense embeddings: turning text into a vector that preserves meaning; the contrast with sparse (TF-IDF, BM25) representations.; Embedding models in practice: OpenAI text-embedding-3, Cohere embed, sentence-transformers (open source); dimension and quality trade-offs.; Document chunking strategies: fixed-size with overlap, semantic chunking, document-structure-aware chunking; the chunk-size-vs-context-size trade-off.; Vector stores: in-memory (FAISS), local (ChromaDB), production (Pinecone, Weaviate, Qdrant, pgvector on PostgreSQL).; Approximate nearest-neighbour indexes: HNSW (the modern default), IVF, the recall-vs-latency trade-off.; Hybrid retrieval: dense + sparse (BM25); reciprocal rank fusion as the simple combiner.; Reranking: cross-encoder rerankers (Cohere Rerank, BGE-reranker), the two-stage retrieve-then-rerank pattern.; Retrieval evaluation: hit-rate, MRR, NDCG; building a small ground-truth retrieval test set is non-negotiable.	CO3
M4	**Prompt Engineering and In-Context Learning** The prompt as the new programming primitive: system prompt, user prompt, few-shot examples, output format specification.; Zero-shot vs few-shot prompting: when each is appropriate; the marginal value of examples.; Chain-of-thought (CoT) prompting: encouraging step-by-step reasoning; the gain on math and reasoning tasks.; Structured output: JSON mode, function calling, tool use; pydantic schemas as the contract.; Prompt-template design discipline: separation of system prompt from user-supplied data; prompt injection as the SQL-injection of LLMs.; In-context learning at intuition level: how a model "learns" from few-shot examples without weight updates.; Prompt evaluation: building a small evaluation set, using LLM-as-judge with caveats, the discipline of measuring before iterating.; Common pitfalls: prompt leakage, instruction injection, over-fitting to a particular model's style, the silent-quality-regression trap when models are upgraded.	CO4
M5	**Fine-Tuning and Adaptation** When fine-tuning helps and when it doesn't: domain adaptation, style adaptation, narrow-task specialisation; vs RAG and prompting.; Full fine-tuning at intuition level: updating every parameter; the cost and infrastructure requirements.; Parameter-efficient fine-tuning (PEFT): the family that includes LoRA, QLoRA, prefix tuning; why these are practical.; LoRA in detail at intuition level: low-rank decomposition of weight updates, the rank hyperparameter, why it works.; QLoRA: 4-bit quantisation + LoRA; running 7B-parameter fine-tuning on a single consumer GPU.; Instruction tuning: dataset curation, the format conventions (Alpaca, ShareGPT), supervised fine-tuning (SFT).; Post-training alignment: RLHF (PPO) at intuition level; DPO as the modern simpler alternative; RLAIF at preview level.; The build-vs-buy decision tree: when to use a closed-source API, when to fine-tune a small open-source model, when to combine.	CO5
M6	**AI Platform Engineering** LLM orchestration frameworks: LangChain (broad, churn-heavy), LlamaIndex (RAG-focused), Semantic Kernel (.NET-focused); the choice criteria.; Observability for LLM systems: trace each call (LangSmith, Helicone, OpenLLMetry); the unit of observation is the chain run, not the individual API call.; Cost management: per-tenant budgets, response caching for identical queries, prompt-caching at provider level (Anthropic), routing to cheaper models for easier queries.; Latency optimisation: streaming responses to reduce perceived latency, prompt-prefix caching, parallelising independent tool calls.; Safety guardrails: input filtering (prompt-injection detection), output filtering (harmful content classifiers), structured-output validation, the human-in-the-loop layer for high-stakes decisions.; Evaluation harnesses: regression test suites for prompt changes; the ML-engineering rigour applied to LLM systems.; Privacy and compliance: PII detection, data retention, GDPR/DPDP applicability; the discipline of never training on customer data without explicit opt-in.; A glance at the operational reality: SLOs and SLIs for LLM services, on-call playbooks for "the model started hallucinating again", capacity planning.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**AI Engineering — Building Applications with Foundation Models**	Chip Huyen	O'Reilly	2024

#	Title	Authors	Publisher	Year
2	**Building LLMs for Production — Enhancing LLM Abilities and Reliability with Prompting, Fine-Tuning, and RAG**	Louis-François Bouchard & Louie Peters	Towards AI	2024
3	**Hands-On Large Language Models — Language Understanding and Generation**	Jay Alammar & Maarten Grootendorst	O'Reilly	2024
4	**Natural Language Processing with Transformers — Building Language Applications with Hugging Face**	Lewis Tunstall, Leandro von Werra & Thomas Wolf	O'Reilly	2022 (Revised Ed.)
5	**Designing Machine Learning Systems**	Chip Huyen	O'Reilly	2022
6	**LangChain in Action**	Ben Auffarth	Manning	2024
7	**Prompt Engineering for LLMs — The Art and Science of Building Production-Ready Applications**	John Berryman & Albert Ziegler	O'Reilly	2024

Human Values, Gender Equality & Professional Ethics

HVEG REG · 26UC0026 PBL · 26UC0026E

L-T-P-S	2-0-0-0 / 3-0-0-0	**Credits**	2	**Contact Hrs/Wk**	2 / 3
Term (Reg · PBL)	S08 · T12	**Mode**	Regular & PBL	**Category**	HSS
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand human values, self-exploration, and harmony at individual and family levels.	BTL2 · Understand
CO2	Understand harmony in society and nature, and holistic, sustainable living.	BTL2 · Understand
CO3	Apply professional ethics and codes of conduct to engineering practice.	BTL3 · Apply
CO4	Analyze gender equality and inclusion issues in education and the workplace.	BTL4 · Analyze
CO5	Analyze ethical dilemmas using moral-reasoning frameworks and real cases.	BTL4 · Analyze
CO6	Analyze contemporary ethics — AI, data, privacy, and sustainability responsibilities.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO6	PO7	PO8	PO9	PSO1	PSO2
CO1	M	–	**H**	–	M	–
CO2	**H**	**H**	–	–	M	–
CO3	**H**	–	**H**	–	M	–
CO4	**H**	–	**H**	–	–	M
CO5	–	–	**H**	M	–	M
CO6	–	M	**H**	–	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Human Values & Self-Exploration** Need and process of value education.; Self-exploration and natural acceptance.; Basic human aspirations — happiness and prosperity.; Harmony in the self (I and the body).; Self-regulation and health.; Values in the family and relationships.; Trust, respect, and gratitude.; From values to conduct. **Labs:** Self-exploration reflection journal + structured peer discussion.; Values-clarification exercise mapping personal values to decisions. **Case Study:** Case Study — values in a real career choice: analyse a professional who faced a values-vs-gain decision (e.g., a whistleblower) and what their values cost and protected.	CO1
M2	**Harmony in Society & Nature** Harmony from family to society.; Universal human order.; Values in social institutions.; Harmony in nature — the four orders.; Interconnectedness and co-existence.; Holistic perception of reality.; Sustainable and ethical living.; Responsibility to future generations. **Labs:** Field-observation lab on a community/ecological system and its values.; Group exercise designing a harmony-based response to a social problem. **Case Study:** Case Study — a community/ecological harmony case (e.g., a successful commons or a sustainability initiative): analyse how shared values sustained collective well-being.	CO2

	Title & Topics	CO
M3	**Professional Ethics** What professional ethics means.; Engineering as a profession and social responsibility.; Codes of ethics (IEEE, Institution of Engineers India).; Safety, risk, and the engineer's duty.; Honesty, integrity, and accountability.; Conflicts of interest.; Whistleblowing.; Rights and responsibilities of engineers. **Labs:** Map a real engineering decision against a professional code of ethics.; Role-play an ethics-committee review of a workplace dilemma. **Case Study:** Case Study — the Bhopal gas tragedy / a major engineering disaster: analyse the ethical failures, responsibilities, and what a code of ethics demanded.	CO3
M4	**Gender Equality & Inclusion** Concepts of sex, gender, and equality.; Gender roles, stereotypes, and bias.; Women in STEM and the workplace.; Gender equality and the law in India.; Sexual harassment and the PoSH Act / Vishakha guidelines.; Inclusive workplaces and allyship.; Diversity beyond gender.; Equity vs equality. **Labs:** Analyse a gender-bias scenario and design an inclusive-workplace policy.; Bystander/allyship role-play with structured reflection. **Case Study:** Case Study — a workplace gender-inclusion or harassment case (and the PoSH framework): analyse the issue, the legal/ethical response, and how inclusion was (or wasn't) achieved.	CO4
M5	**Ethical Reasoning & Decision-Making** Moral theories — consequentialism, deontology, virtue ethics.; Rights and justice.; Ethical decision-making frameworks.; Resolving moral dilemmas.; Cross-cultural and global ethics.; Corporate social responsibility.; Ethics of teamwork and leadership.; Building moral courage. **Labs:** Structured ethical-dilemma debate using competing frameworks.; Apply a decision-making framework to a real professional dilemma. **Case Study:** Case Study — a classic ethics dilemma (e.g., the Challenger launch decision): analyse it through multiple moral frameworks and reach a reasoned position.	CO5
M6	**Contemporary Ethics — AI, Data & Sustainability** Ethics of artificial intelligence.; Algorithmic bias and fairness.; Data privacy and the DPDP Act 2023.; Surveillance and consent.; Ethics of automation and jobs.; Environmental ethics and sustainability.; Responsible innovation.; The engineer's role in a tech-shaped society. **Labs:** AI-ethics audit of a real system for bias, privacy, and consent.; Design an ethical-and-sustainable guideline for a technology project. **Case Study:** Case Study — an AI-ethics controversy (e.g., a biased hiring/credit algorithm): analyse the harm, responsibility, and the principles that should have governed it.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**A Foundation Course in Human Values and Professional Ethics, 2nd Revised Ed.**	R. R. Gaur, R. Asthana & G. P. Bagaria	Excel Books	2019
2	**Professional Ethics and Human Values**	R. S. Naagarazan	New Age International	2006
3	**Ethics in Engineering, 4th Ed.**	Mike W. Martin & Roland Schinzinger	McGraw-Hill	2005
4	**Engineering Ethics: Concepts and Cases, 5th Ed.**	Charles E. Harris, Michael S. Pritchard & Michael J. Rabins	Cengage Learning	2013
5	**Justice: What's the Right Thing to Do?**	Michael J. Sandel	Farrar, Straus and Giroux	2009
6	**Seeing Like a Feminist**	Nivedita Menon	Penguin/Zubaan	2012
7	**Ethics: Theory and Contemporary Issues, 9th Ed.**	Barbara MacKinnon & Andrew Fiala	Cengage Learning	2017

Indian Constitution

INCT REG · 26UC0008 PBL · 26UC0008E

L-T-P-S	2-0-0-0 / 3-0-0-0	Credits	0	Contact Hrs/Wk	2 / 3
Term (Reg · PBL)	S07 · T11	Mode	Regular & PBL	Category	AUC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the Constitution as the practical foundation of the laws governing professional life.	BTL2 · Understand
CO2	Apply fundamental rights to real personal, civic, and workplace situations.	BTL3 · Apply
CO3	Analyze equality, anti-discrimination, and anti-harassment protections at work.	BTL4 · Analyze
CO4	Analyze employee rights and obligations under constitutionally-grounded labour and contract law.	BTL4 · Analyze
CO5	Analyze digital-age rights — privacy, data protection, IP, and speech — for a working engineer.	BTL4 · Analyze
CO6	Apply mechanisms to access justice and use the law for personal and collective benefit.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO6	PO8	PO10	PO11	PSO2
CO1	H	M	–	–	M
CO2	H	H	–	–	M
CO3	H	H	–	–	M
CO4	H	–	–	M	M
CO5	H	H	–	–	M
CO6	H	–	M	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The Constitution as Your Foundation** What the Constitution is and why it matters to you personally.; Preamble values — justice, liberty, equality, fraternity — in everyday and work life.; The Constitution as a living, enforceable document (not trivia).; Rule of law and constitutional supremacy.; How constitutional law flows down into the laws that govern your job.; Fundamental Rights, Directive Principles, and Duties — the practical map.; How to find, read, and make sense of a law in plain language.; Your identity and responsibilities as a citizen-professional. **Labs:** Map Preamble values to real personal and workplace decisions.; 'Find the law' lab: locate and plain-language summarise the law behind a given right. **Case Study:** Case Study — an ordinary person who invoked a constitutional right to fix a real problem (e.g., a worker securing dignity or livelihood): trace how a Preamble value became an enforceable outcome.	CO1
M2	**Your Fundamental Rights in Practice** Right to equality (Art 14–18) — fairness and non-arbitrariness in daily life.; Right to freedom (Art 19) — speech, profession, movement — and reasonable restrictions.; Right to life and personal liberty (Art 21) — dignity, livelihood, health, privacy.; Right against exploitation (Art 23–24) — forced and child labour.; Freedom of religion and cultural/educational rights (in brief).; Rights against the State vs against private employers.; Right to constitutional remedies (Art 32) — the writs, in practice.; How and where to actually seek a remedy. **Labs:** Scenario lab: match real situations to the fundamental right that applies.; Outline a writ/grievance for a rights-violation scenario. **Case Study:** Case Study — the expansion of Art 21 (e.g., right to livelihood / privacy / dignity): analyse how a broad reading of one right now protects working people.	CO2

	Title & Topics	CO
M3	**Equality, Anti-Discrimination & Anti-Harassment at Work** Equality applied to hiring, pay, and promotion.; Anti-discrimination — gender, caste, religion, disability.; Equal pay for equal work.; Sexual harassment and the PoSH Act 2013 (from Vishakha to statute).; The Internal Committee — your rights and the complaint process.; Rights of persons with disabilities (RPwD Act 2016).; Maternity and related benefits and their constitutional basis.; Demanding and building an inclusive workplace. **Labs:** PoSH complaint walkthrough: draft and route a complaint correctly.; Review a workplace anti-discrimination/inclusion policy for gaps. **Case Study:** Case Study — a real workplace harassment/discrimination matter and its resolution: analyse the rights invoked, the PoSH/IC process, and the outcome.	CO3
M4	**Your Rights & Duties as an Employee** Constitutional basis of labour welfare (DPSP — Art 38, 39, 41, 42, 43).; The new Labour Codes — Wages, Industrial Relations, Social Security, OSH.; Employment contracts — what to check before you sign.; Wages, hours, leave, PF, ESI, and gratuity.; NDAs, non-compete, and employment bonds — what is enforceable vs not (Art 19 freedom of profession).; Termination, notice, and recourse against wrongful dismissal.; Whistleblower protection.; Trade unions and collective rights. **Labs:** Review a sample employment contract and flag the red-flag clauses.; Draft a professional grievance/representation letter for a workplace issue. **Case Study:** Case Study — an employment-bond / non-compete dispute (common for fresh engineers): analyse what the law actually enforces and how an employee protected themselves.	CO4
M5	**The Engineer-Citizen in the Digital Age** Right to privacy (Puttaswamy 2017) — what it means for you and your data.; The DPDP Act 2023 — your rights as a data principal and duties as a professional handling data.; Intellectual property at work — who owns what you create (work-for-hire), patents, copyright.; Freedom of speech online — limits, defamation, and employer social-media policies.; IT Act essentials — cyber offences, your protection and your liability.; Consumer rights (Consumer Protection Act 2019) as professional and consumer.; Contract law essentials (Indian Contract Act) for everyday agreements.; Where professional ethics meets legal duty. **Labs:** Draft a data-rights (DPDP) request and a basic privacy-notice check.; Analyse an IP-ownership / NDA clause and explain its real effect. **Case Study:** Case Study — a data-privacy or IP-ownership dispute relevant to a tech employee: analyse who held the rights, what the law said, and how it was resolved.	CO5
M6	**Accessing Justice & Using the Law for Your Benefit** The court and tribunal hierarchy — which forum for which dispute.; Consumer, labour, civil, and writ forums in practice.; Alternative dispute resolution — arbitration, mediation, Lok Adalat.; Right to Information (RTI) — how to actually use it.; Legal aid and your right to it.; Filing complaints — police, and commissions (NHRC, women's, SC/ST, consumer).; Public Interest Litigation for collective benefit.; Practical legal literacy — notices, FIRs, affidavits, and staying law-aware. **Labs:** File a mock RTI application for a real information need.; Choose-the-right-forum simulation: route given disputes to the correct remedy. **Case Study:** Case Study — someone who used RTI / a consumer forum / a tribunal to secure their rights: analyse the path they took and why it worked.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Introduction to the Constitution of India, 25th Ed.**	Durga Das Basu	LexisNexis	2018
2	**The Indian Constitution (Oxford India Short Introductions)**	Madhav Khosla	Oxford University Press	2012
3	**Labour and Industrial Laws**	S. N. Misra	Central Law Publications	2021
4	**Information Technology Law and Practice, 5th Ed.**	Vakul Sharma	LexisNexis	2021
5	**Intellectual Property Rights, 2nd Ed.**	V. K. Ahuja	LexisNexis	2017
6	**Law of Consumer Protection: Principles and Practice**	Avtar Singh	Eastern Book Company	2020
7	**Privacy 3.0: Unlocking Our Data-Driven Future**	Rahul Matthan	HarperCollins India	2018

Indian Knowledge Systems

IKSL 26UC0017

L-T-P-S	2-0-0-0	Credits	0	Contact Hrs/Wk	2
Semester	2S	Mode	Regular	Category	AUC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand the scope of Indian Knowledge Systems and their contributions.	BTL2 · Understand
CO2	Understand the foundations and philosophy of Vedic Mathematics and its sutras.	BTL2 · Understand
CO3	Apply Vedic arithmetic techniques for fast, accurate computation.	BTL3 · Apply
CO4	Apply Vedic methods to algebra and problem-solving.	BTL3 · Apply
CO5	Understand India's broader contributions to mathematics and science.	BTL2 · Understand
CO6	Analyze the relevance of IKS and Vedic Mathematics in the modern AI era.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO5	PO6	PO11	PSO1	PSO2
CO1	–	–	–	**H**	**H**	M	–
CO2	**H**	–	–	–	M	M	–
CO3	**H**	M	–	–	–	M	–
CO4	**H**	**H**	–	–	–	M	–
CO5	–	–	–	**H**	**H**	M	–
CO6	–	–	M	–	**H**	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Introduction to Indian Knowledge Systems** What Indian Knowledge Systems are.; Scope and classification of IKS.; Oral and written knowledge traditions.; The Vedic corpus and Vedangas.; Knowledge domains — science, math, arts, medicine.; Gurukula and learning traditions.; Why IKS matters today.; IKS in the national curriculum (NEP 2020). **Labs:** Survey and map domains of IKS with examples for each.; Research-and-present lab on one Indian knowledge contribution. **Case Study:** Case Study — a rediscovered Indian contribution (e.g., the concept of zero / ancient metallurgy of the Iron Pillar): analyse the contribution and how it shaped the world.	CO1
M2	**Vedic Mathematics Foundations** Origin — Jagadguru Bharati Krishna Tirthaji.; The 16 Sutras and 13 sub-sutras.; Philosophy and coherence of the system.; Place value and number sense.; Complements and the 'all from 9, last from 10'.; Mental-math principles.; Advantages — speed and accuracy.; When Vedic methods help most. **Labs:** Drill lab: apply 'all from 9, last from 10' to subtraction problems.; Speed-vs-accuracy lab comparing conventional and Vedic methods. **Case Study:** Case Study — mental-math champions / competitive-exam aspirants: analyse how Vedic techniques give a real edge in speed and accuracy under exam conditions.	CO2

	Title & Topics	CO
M3	**Vedic Arithmetic Techniques** Multiplication by Nikhilam.; Multiplication by Urdhva-Tiryagbhyam (vertically and crosswise).; Multiplication special cases.; Division techniques.; Squares and square roots.; Cubes and cube roots.; Divisibility tests.; Digit sums and checking answers. **Labs:** Multiplication lab using Nikhilam and Urdhva-Tiryagbhyam.; Squares/cubes and roots lab with self-timed problems. **Case Study:** Case Study — large-number multiplication under time pressure: students solve a set conventionally and via sutras, and analyse the time/accuracy difference.	CO3
M4	**Vedic Algebra & Applications** Vedic approach to algebra.; Linear equations via sutras.; Simultaneous equations.; Factorisation techniques.; Quadratic insights.; Applications in aptitude and competitive exams.; Cross-checking with digit sums.; Building a personal technique toolkit. **Labs:** Equation-solving lab applying Vedic algebra sutras.; Aptitude lab: solve a timed competitive-style set using Vedic methods. **Case Study:** Case Study — a competitive-exam quantitative section (CAT/GATE/bank): analyse how a Vedic toolkit can shave time on real problem types.	CO4
M5	**India's Contributions to Mathematics & Science** The decimal system and zero.; Aryabhata, Brahmagupta, Bhaskara.; The Kerala school and infinite series.; Sulba Sutras and geometry.; Indian astronomy.; Ancient metallurgy and chemistry.; Ayurveda and life sciences (overview).; Linguistics and Panini's grammar. **Labs:** Timeline lab mapping Indian mathematical/scientific milestones.; Deep-dive presentation on one Indian mathematician/scientist. **Case Study:** Case Study — the Kerala school's pre-Newtonian series: analyse how Indian mathematicians anticipated calculus concepts centuries earlier.	CO5
M6	**IKS & Vedic Mathematics in the AI Era** Algorithmic thinking in the sutras.; Vedic methods and computing/digital logic.; IKS and modern STEM education.; Preservation and digitisation of texts.; AI for ancient-text analysis and translation.; Heritage, ethics, and authenticity.; Evidence-based vs exaggerated claims.; Carrying IKS forward responsibly. **Labs:** Express a Vedic sutra as an algorithm/flowchart and discuss its logic.; Critically evaluate a popular IKS claim against evidence. **Case Study:** Case Study — AI applied to ancient Indian texts (digitisation/translation of manuscripts): analyse how modern AI helps preserve and study IKS, and the pitfalls.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Vedic Mathematics**	Jagadguru Swami Sri Bharati Krishna Tirthaji Maharaja	Motilal Banarsidass	2015
2	**Introduction to Indian Knowledge System: Concepts and Applications**	B. Mahadevan, Vinayak Rajat Bhat & Nagendra Pavana R. N.	PHI Learning	2022
3	**Vedic Mathematics Made Easy**	Dhaval Bathia	Jaico Publishing House	2005
4	**The Power of Vedic Maths, 2nd Ed.**	Atul Gupta	Jaico Publishing House	2004
5	**Mathematics in India: 500 BCE–1800 CE**	Kim Plofker	Princeton University Press	2009
6	**Pride of India: A Glimpse into India's Scientific Heritage**	Samskrita Bharati	Samskrita Bharati	2006
7	**A Concise History of Science in India, 2nd Ed.**	D. M. Bose, S. N. Sen & B. V. Subbarayappa (Eds.)	Universities Press	2009

Indian Knowledge Systems

IKSL 26UC0017E

L-T-P-S	3-0-0-0	Credits	0	Contact Hrs/Wk	3
Trimester	2S	Mode	PBL (Project-Based)	Category	AUC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand Indian Knowledge Systems, their epistemology, and the ancient Indian engineering mindset.	BTL2 · Understand
CO2	Understand computational, logical, and linguistic foundations in IKS relevant to computing and AI.	BTL2 · Understand
CO3	Understand Indian contributions to mathematics, astronomy, and precision instrumentation.	BTL2 · Understand
CO4	Analyze ancient Indian civil engineering, architecture, and town planning.	BTL4 · Analyze
CO5	Analyze ancient Indian materials, metallurgy, and mechanical systems.	BTL4 · Analyze
CO6	Analyze ancient Indian life sciences and chemistry, and the responsible, ethical revival of IKS.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO6	PO8	PO11	PSO1	PSO2
CO1	–	–	**H**	–	**H**	M	–
CO2	M	–	**H**	–	–	M	–
CO3	M	–	**H**	–	–	M	–
CO4	–	M	**H**	–	–	–	M
CO5	–	M	**H**	–	–	–	M
CO6	–	–	**H**	**H**	–	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Foundations of IKS & the Ancient Engineering Mind** What Indian Knowledge Systems are; the vidyasthanas and classification of knowledge.; Knowledge traditions — oral transmission, guru-shishya, and preservation.; Epistemology — pramanas (means of valid knowledge), observation and inference.; Systems thinking and first-principles reasoning in ancient India.; Sanskrit as a precise knowledge-encoding language and technical terminology.; Interdisciplinarity — how math, science, and craft connected.; Where ancient lineage is strong, and where modern domains have no direct antecedent (honest mapping).; Relevance of IKS to modern engineering and NEP 2020. **Labs:** Map the vidyasthanas / IKS domains onto our nine modern branches — and mark where there is genuinely no ancient counterpart.; Epistemology lab: classify how a given ancient claim was 'known' (perception, inference, testimony). **Case Study:** Case Study — how a complex engineering tradition (e.g., temple-building or metallurgy) was preserved and transmitted for centuries without modern documentation: analyse the knowledge-system that made it possible.	CO1
M2	**Computation, Logic & Language — CSE / CS&IT / AI&DS** Pingala's Chandahshastra — binary numbers and combinatorics.; Meru Prastara (Pascal's triangle) and the Hemachandra (Fibonacci) sequence.; Panini's Ashtadhyayi — a formal, generative rule-system for language.; Panini and modern formal-language / compiler theory.; Nyaya and Navya-Nyaya — formal logic, inference, and structured debate.; Algorithms in Indian mathematics — kuttaka and chakravala.; Encoding systems — Katapayadi and Bhutasankhya.; Connections to NLP, knowledge representation, and AI reasoning. **Labs:** Express a Panini-style rule as a production rule / BNF and discuss the parallel.; Generate a combinatorial pattern using Pingala's method and relate it to binary. **Case Study:** Case Study — Panini's grammar and its influence on formal-language theory and computational linguistics: analyse why a 2,500-year-old grammar reads like a programming-language specification.	CO2

	Title & Topics	CO
M3	**Mathematics, Astronomy & Precision Instruments** The decimal place-value system and the concept of zero.; Aryabhata, Brahmagupta, and Bhaskara II — algebra and trigonometry.; The Kerala school — infinite series and precursors to calculus.; Jyotisha — planetary models, eclipse and time prediction.; Calendars, timekeeping, and systems of measurement.; Astronomical instruments and the Jantar Mantar observatories.; The science of sound — Shiksha/phonetics and acoustics (a cross-cutting thread for ECE/communication).; Applications in modern computation, measurement, and signal foundations. **Labs:** Apply an ancient algorithm (e.g., for square roots or sine values) and compare with the modern method.; Analyse the geometry and accuracy of one Jantar Mantar instrument. **Case Study:** Case Study — the Jantar Mantar observatories as precision instrumentation: analyse how stone instruments achieved remarkable measurement accuracy without electronics.	CO3
M4	**Civil Engineering, Architecture & Town Planning — CIVIL** Indus Valley — grid town planning, standardized bricks, and the world's earliest urban drainage.; Sulba Sutras — altar geometry, the Pythagorean relation, and area transformations.; Vastu Shastra and Shilpa Shastra — design and layout principles.; Temple architecture and engineering (Mayamata, Manasara; Nagara and Dravida styles).; Structural feats — the Kailasa rock-cut temple and the Brihadeeswara temple.; Water engineering — stepwells, tanks, canals, and the Kallanai (Grand Anicut) dam.; Materials — lime mortar, fired brick, and stone construction.; Lessons for modern sustainable and climate-responsive civil engineering. **Labs:** Analyse the grid-planning and drainage logic of an Indus city.; Reconstruct a Sulba-Sutra geometric construction and verify it. **Case Study:** Case Study — the Kallanai dam (Grand Anicut) or Indus drainage: analyse the hydraulic and structural engineering that has functioned for centuries to millennia.	CO4
M5	**Materials, Metallurgy & Mechanical Systems — MECH / Materials** Wootz / Damascus steel — microstructure and superior properties.; The Iron Pillar of Delhi — corrosion resistance and the protective film.; Zinc distillation at Zawar — early industrial metallurgy.; Rasashastra — mineral processing and proto-chemistry.; Yantras and machines — the Samarangana Sutradhara and mechanical devices.; Water-lifting and agricultural machinery.; Shipbuilding and maritime engineering.; Modern materials science learning from ancient metallurgy. **Labs:** Analyse the metallurgy of wootz steel and why its properties were hard to replicate.; Reverse-engineer the working principle of an ancient yantra / water-lifting device. **Case Study:** Case Study — the Iron Pillar of Delhi: analyse the materials engineering behind its 1,600-year corrosion resistance.	CO5
M6	**Life Sciences, Chemistry & Sustainable Bio-Systems — BIOTECH** Ayurveda — a systems approach to physiology, diagnosis, and health.; Sushruta Samhita — surgery, surgical instruments, and anatomy.; Charaka Samhita — pharmacology and clinical method.; Rasashastra and pharmaceutics — drug preparation and standardization.; Vrikshayurveda — plant science, soil, and agriculture.; Fermentation and traditional food technology.; Biomimicry, frugality, and sustainability embedded in IKS.; Ethics and IP — biopiracy, the Traditional Knowledge Digital Library, and AI for preserving IKS. **Labs:** Map an Ayurvedic systems-diagnosis onto a modern systems-thinking framework.; Explore the Traditional Knowledge Digital Library and a real biopiracy case, and propose a protection approach. **Case Study:** Case Study — Sushruta's surgical techniques (e.g., reconstructive rhinoplasty) and their documented influence on modern surgery: analyse the method and the evidence.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Introduction to Indian Knowledge System: Concepts and Applications**	B. Mahadevan, Vinayak Rajat Bhat & Nagendra Pavana R. N.	PHI Learning	2022
2	**A Concise History of Science in India, 2nd Ed.**	D. M. Bose, S. N. Sen & B. V. Subbarayappa (Eds.)	Universities Press	2009
3	**Indian Science and Technology in the Eighteenth Century**	Dharampal	Other India Press	2000
4	**Mathematics in India: 500 BCE–1800 CE**	Kim Plofker	Princeton University Press	2009
5	**Pride of India: A Glimpse into India's Scientific Heritage**	Samskrita Bharati	Samskrita Bharati	2006
6	**The Wonder That Was India**	A. L. Basham	Picador India / Rupa	2004
7	**Marvels of Indian Iron Through the Ages**	R. Balasubramaniam	Rupa & Infinity Foundation	2008

Industrial Automation, PLC & SCADA Systems IAPS REG · 26EE3208E PBL · 26EE3208E

L-T-P-S	2-0-4-0 / 3-0-6-0	Credits	4	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S06 · T08	Mode	Regular & PBL	Category	PCC
Pre-requisite	CSEE				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze the industrial-automation pyramid — field, control, supervisory and enterprise levels — and the role of each in a modern automated plant.	BTL4 · Analyze
CO2	Apply sensor and actuator interfacing — discrete and analog field devices, signal conditioning, and standard 4–20 mA / industrial signalling — to acquire and act on plant data.	BTL3 · Apply
CO3	Apply PLC programming using IEC 61131-3 languages (ladder logic, function block, structured text) to implement discrete and sequential control.	BTL3 · Apply
CO4	Apply HMI and SCADA design — operator screens, alarms, trends and data logging — to supervise and operate an automated process.	BTL3 · Apply
CO5	Analyze industrial communication networks and protocols (Modbus, PROFIBUS/PROFINET, EtherNet/IP, OPC UA) and select appropriate connectivity for an automation system.	BTL4 · Analyze
CO6	Analyze Industry 4.0 and OT/IT convergence — IIoT, digital twins, MES, and OT cybersecurity — and their application to smart manufacturing.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO7	PSO1
CO1	H	H	–	–	–	H
CO2	H	–	H	–	–	H
CO3	H	–	H	–	–	H
CO4	–	–	H	H	–	H
CO5	H	–	–	H	–	H
CO6	–	H	–	–	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The Automation Pyramid** What industrial automation is and why it matters (productivity, safety, quality).; The automation pyramid: field, control (PLC/DCS), supervisory (SCADA), MES, ERP.; PLC vs DCS vs PAC — when each is used.; Discrete vs process vs batch automation.; A tour of a real automated process (conveyor/bottling/water-treatment example).; The role of the control engineer / system integrator.; Standards landscape: IEC 61131, ISA-88/95, IEC 62443.; Framing the course project process to automate.	CO1
M2	**Sensors, Actuators and Field Interfacing** Discrete sensors: proximity, limit, photoelectric.; Analog sensors: temperature, pressure, flow, level.; The 4–20 mA current loop and industrial signal standards.; Actuators: solenoids, relays, contactors, motor starters, valves.; Variable-frequency drives as automation actuators (drive internals owned by EMDE).; Signal conditioning and isolation.; Wiring, grounding and noise in the plant environment.; Interfacing field devices to PLC I/O.	CO2
M3	**PLC Programming (IEC 61131-3)** PLC architecture: CPU, I/O, scan cycle, memory.; Ladder logic: contacts, coils, latches, the scan-based execution model.; Timers and counters.; Function block diagram and structured text.; Sequential function charts for sequential processes.; PID function blocks in the PLC (using CSEA tuning knowledge).; Program structure, modularity and good practice.; Programming the project process control logic.	CO3

	Title & Topics	CO
M4	**HMI and SCADA** The HMI: operator screens, graphics, navigation.; SCADA architecture: RTUs, master station, historian.; Alarms and events; alarm management best practice.; Trends, data logging and the historian.; Recipes and batch control (ISA-88 concept).; Redundancy and reliability in SCADA.; Security of HMI/SCADA (intro; deep OT security in M6).; Building the HMI/SCADA for the project process.	CO4
M5	**Industrial Communication Networks** Why fieldbuses: deterministic, robust plant communication.; Serial fieldbuses: Modbus RTU, PROFIBUS.; Industrial Ethernet: PROFINET, EtherNet/IP, Modbus TCP.; OPC and OPC UA — the interoperability and IT/OT bridge standard.; Time-sensitive networking (concept).; Network topology, determinism and redundancy.; Wireless in industrial settings (concept).; Networking the project PLC, HMI and field devices.	CO5
M6	**Industry 4.0 and OT/IT Convergence** The Industry 4.0 vision and India's SAMARTH Udyog initiative.; Industrial IoT (IIoT): edge gateways, cloud connectivity.; Digital twins of processes and machines.; Manufacturing Execution Systems (MES) and ISA-95.; Predictive maintenance and analytics on plant data.; OT cybersecurity: IEC 62443, the Purdue model, segmentation.; The smart factory and the role of AI.; Adding an IIoT/analytics layer to the project for the demo.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Programmable Logic Controllers**	Frank D. Petruzella	McGraw-Hill	2016 (5th Ed.)
2	**Programmable Logic Controllers**	W. Bolton	Newnes	2015 (6th Ed.)
3	**Industrial Automation: Hands-On**	Frank Lamb	McGraw-Hill	2013
4	**SCADA: Supervisory Control and Data Acquisition**	Stuart A. Boyer	ISA	2009 (4th Ed.)
5	**Automating Manufacturing Systems with PLCs**	Hugh Jack	Open-source (Jack)	2010

Insights to Sustainable Development Goals

SDGL REG · 26UC0032 PBL · 26UC0032E

L-T-P-S	2-0-0-0 / 3-0-0-0	Credits	0	Contact Hrs/Wk	2 / 3
Term (Reg · PBL)	1S · 1S	Mode	Regular & PBL	Category	AUC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand sustainable development — concept, pillars, and the road to the SDGs.	BTL2 · Understand
CO2	Understand the 17 SDGs, their targets, and their interlinkages.	BTL2 · Understand
CO3	Analyze the people-and-prosperity SDGs in the Indian and global context.	BTL4 · Analyze
CO4	Analyze the planet SDGs — climate, energy, water, and ecosystems.	BTL4 · Analyze
CO5	Apply SDG measurement, governance, and localisation.	BTL3 · Apply
CO6	Analyze how technology, AI, and innovation can accelerate the SDGs.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO5	PO6	PO7	PO11	PSO1	PSO2
CO1	–	M	**H**	–	M	–
CO2	–	**H**	**H**	–	M	–
CO3	–	**H**	M	–	–	M
CO4	–	M	**H**	–	–	M
CO5	–	–	**H**	M	–	M
CO6	**H**	–	**H**	–	–	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Foundations of Sustainable Development** What sustainable development means.; The Brundtland definition.; The three pillars — economic, social, environmental.; Planetary boundaries.; From MDGs to SDGs.; The 2030 Agenda and 'leave no one behind'.; Sustainability vs growth debates.; India's sustainability context. **Labs:** Map a real project against the three pillars of sustainability.; Footprint lab: estimate and reduce a sustainability footprint. **Case Study:** Case Study — a development-vs-environment trade-off (e.g., a dam/infrastructure project): analyse how the three pillars of sustainability were balanced or violated.	CO1
M2	**The 17 SDGs Overview** Overview of all 17 goals.; Targets and indicators.; The SDG 'wedding cake' model.; Interlinkages and trade-offs among goals.; Synergies across goals.; 'Leave no one behind'.; The global indicator framework.; Tracking progress. **Labs:** Build an SDG interlinkage map for a chosen development problem.; Indicator lab: interpret SDG progress data for a country/state. **Case Study:** Case Study — SDG interlinkages in practice (e.g., clean energy enabling health and education): analyse how progress on one goal advances others.	CO2

	Title & Topics	CO
M3	**People & Prosperity SDGs** No poverty and zero hunger.; Good health and well-being.; Quality education.; Gender equality.; Decent work and economic growth.; Reduced inequalities.; Industry, innovation, and infrastructure.; India's progress on people/prosperity goals. **Labs:** Analyse a region's people/prosperity indicators and gaps.; Design an intervention targeting a people/prosperity SDG. **Case Study:** Case Study — an Indian social initiative (e.g., a health/education/livelihood program): analyse its SDG contribution and outcomes.	CO3
M4	**Planet SDGs** Clean water and sanitation.; Affordable and clean energy.; Climate action.; Life below water.; Life on land.; Sustainable cities and communities.; Responsible consumption and production.; Circular economy. **Labs:** Audit a setting for responsible consumption and propose circular fixes.; Design a planet-SDG (energy/water/waste) solution for a community. **Case Study:** Case Study — a clean-energy or water initiative in India (e.g., solar mission / river rejuvenation): analyse its planet-SDG impact and scalability.	CO4
M5	**Partnership, Governance & Measurement** Peace, justice, and strong institutions (SDG16).; Partnerships for the goals (SDG17).; Financing the SDGs.; Indicators and the SDG India Index.; Localising the SDGs.; Voluntary National Reviews.; Role of governments, business, and civil society.; Accountability and reporting. **Labs:** Build an SDG scorecard for an organisation/community.; Design a multi-stakeholder partnership for an SDG target. **Case Study:** Case Study — the SDG India Index / a state's SDG localisation: analyse how SDGs are measured and localised, and what drives improvement.	CO5
M6	**Technology, AI & Innovation for the SDGs** Technology as an SDG accelerator.; AI for good — use cases across goals.; Social entrepreneurship and impact ventures.; Frugal and inclusive innovation.; Digital public infrastructure for development.; Measuring impact (theory of change).; The engineer's and youth's role.; Designing an SDG-aligned solution. **Labs:** Design a technology/AI-enabled solution mapped to an SDG with a theory of change.; Pitch a social-impact venture targeting one or more SDGs. **Case Study:** Case Study — AI/technology for an SDG (e.g., AI in healthcare access or precision agriculture): analyse how technology accelerates a specific goal, with its risks.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**The Age of Sustainable Development**	Jeffrey D. Sachs	Columbia University Press	2015
2	**Doughnut Economics: Seven Ways to Think Like a 21st-Century Economist**	Kate Raworth	Random House Business	2017
3	**An Introduction to Sustainable Development, 4th Ed.**	Jennifer A. Elliott	Routledge	2012
4	**Sustainability: Principles and Practice, 2nd Ed.**	Margaret Robertson	Routledge	2017
5	**Prosperity without Growth: Foundations for the Economy of Tomorrow, 2nd Ed.**	Tim Jackson	Routledge	2017
6	**Our Common Future**	World Commission on Environment and Development (Brundtland Commission)	Oxford University Press	1987
7	**Factfulness: Ten Reasons We're Wrong About the World**	Hans Rosling with Ola Rosling & Anna Rosling Rönnlund	Flatiron Books	2018

IoT Protocols & Cloud Integration

IOTP REG · 26IN2201 PBL · 26IN2201E

L-T-P-S	2-0-2-4 / 3-0-6-0	Credits	4	Contact Hrs/Wk	8 / 9
Term (Reg · PBL)	S04 · T06	Mode	Regular & PBL	Category	PCC
Pre-requisite	FOIS				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand IoT-protocol stacks — physical/MAC layers (LoRa, BLE, Wi-Fi, Zigbee), network and transport (IPv6, 6LoWPAN, UDP), application (MQTT, CoAP, HTTP) — and place modern IoT systems in this canonical structure.	BTL2 · Understand
CO2	Apply LPWAN protocols — LoRaWAN at depth, NB-IoT and LTE-M at intuition, Sigfox at preview — to long-range low-power IoT applications.	BTL3 · Apply
CO3	Apply MQTT at production depth — broker / client architecture, topics, QoS levels, retained messages, last-will-and-testament, security via TLS + X.509.	BTL3 · Apply
CO4	Apply cloud-IoT integration — AWS IoT Core / Azure IoT Hub / ThingsBoard, device provisioning, fleet management, OTA updates, time-series ingestion (TimescaleDB / InfluxDB) — to ship complete pipelines.	BTL3 · Apply
CO5	Analyze IoT security — device identity (X.509), TLS, secure provisioning, OTA security, the Mirai-class threat models, the modern zero-trust patterns.	BTL4 · Analyze
CO6	Analyze edge AI integration, 5G + IoT convergence, digital twins at intuition, the AI-augmented IoT design workflow — to architect 2026-current IoT systems.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO6	PO8	PSO1
CO1	H	H	–	–	–	–	H
CO2	H	–	H	–	–	–	H
CO3	H	–	H	–	–	–	H
CO4	H	–	H	H	–	–	H
CO5	H	–	H	–	H	H	H
CO6	H	–	H	H	–	–	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**IoT Protocol Stacks** The IoT-protocol stack — analogous to OSI but with the IoT-specific layer choices; the canonical reference architecture.; PHY/MAC layers — LoRa (long-range), BLE (short-range), Wi-Fi (medium), Zigbee/Thread (mesh); range and power trade-offs.; 6LoWPAN — IPv6 over Low-power Wireless PAN; how IPv6 addresses fit in IEEE 802.15.4 frames.; Transport — UDP (default for IoT), TCP (when reliability matters), QUIC at preview.; Application protocols — MQTT (the dominant IoT pub-sub), CoAP (RESTful for constrained devices), HTTP (familiar but heavy).; Comparing protocol choices — the canonical decision matrix (range, power, throughput, latency, cost).; A first lab: in Wireshark, capture and identify MQTT, CoAP, and HTTP packets from a sample IoT device; observe overhead differences.	CO1
M2	**LPWAN — LoRaWAN at Depth** LPWAN motivation — battery-powered devices, 5-15 km range, no Wi-Fi assumption; the use cases (agricultural, asset tracking, smart meters).; LoRa modulation at intuition — chirp spread spectrum; the canonical SF7-SF12 spreading factors and the range-vs-data-rate trade-off.; LoRaWAN architecture — devices → gateway → network server → application server; the asymmetric topology.; LoRaWAN device classes — Class A (asynchronous, lowest power), Class B (scheduled), Class C (always-listening); when each fits.; LoRaWAN security — AppSKey + NwkSKey, OTAA vs ABP activation; the canonical X.509-style key management.; NB-IoT and LTE-M at intuition — cellular alternatives via existing networks; Indian operators (Jio, Airtel) deployments.; A lab: deploy two ESP32 LoRa boards in a private LoRaWAN network via ChirpStack on a Raspberry Pi; demonstrate uplink data at SF7 (close range) and SF12 (long range, slow).	CO2

	Title & Topics	CO
M3	**MQTT at Production Depth** MQTT recap — broker / client pub-sub architecture, the canonical Mosquitto broker.; Topics — hierarchical naming (sensors/floor1/temp), wildcards (+, #) at intuition, the canonical naming-convention discipline.; QoS levels — 0 (at-most-once), 1 (at-least-once), 2 (exactly-once); the overhead trade-offs.; Retained messages — last-known-good values for new subscribers; the canonical "device status" pattern.; Last-will-and-testament — broker publishes message on client unexpected disconnect; the canonical "device offline" detection.; MQTT security — TLS for transport, X.509 client certificates for identity, ACL for topic-level authorization.; Production MQTT brokers — HiveMQ, EMQX, AWS IoT Core (MQTT-compatible); the multi-tenant scale concerns.; A lab: deploy Mosquitto with TLS + client certs on a free-tier cloud VM; provision 3 ESP32 clients with their own X.509 identities; demonstrate publish-subscribe with QoS 1; demonstrate that without the cert, connection is rejected.	CO3
M4	**Cloud-IoT Integration** AWS IoT Core at depth — device registry, X.509-based auth, IoT Rules engine for routing data, IoT Jobs for fleet management.; Azure IoT Hub at intuition — the Microsoft alternative; device twin pattern.; GCP IoT Core deprecated 2023 — the canonical "lock-in risk" cautionary tale; the migration to ClearBlade.; ThingsBoard at depth — the open-source alternative; deployable on free-tier cloud VM; canonical for Indian-context deployments where data residency matters.; Time-series databases — TimescaleDB (Postgres extension), InfluxDB (the canonical OSS option), AWS Timestream as managed alternative.; Dashboards — Grafana with InfluxDB or TimescaleDB; the canonical OSS combination.; Device fleet management — provisioning, OTA updates, remote configuration, health monitoring.; A lab: provision 5 ESP32 devices on a free-tier ThingsBoard instance; ingest live sensor data; build dashboard with current readings + 24-hour history; demonstrate OTA firmware update of one device.	CO4
M5	**IoT Security in Practice** IoT threat landscape — Mirai botnet (2016) as the canonical wakeup; the Indian IoT security context (mass-deployed cheap devices, weak defaults).; Common attack categories — default-password device hijacking, unencrypted communications (MITM), supply-chain compromises, denial-of-service amplification.; Device identity — X.509 client certificates as the modern standard; secure-element chips (ATECC608A, NXP SE050) for hardware-rooted identity.; Secure boot and trusted firmware — chain of trust from ROM bootloader to application; the canonical implementation patterns.; OTA security — signed firmware images, anti-rollback, the canonical "what if the OTA channel is compromised?" attack scenarios.; TLS for IoT at depth — TLS 1.3, the implementation footprint on Cortex-M4 (mbedTLS, wolfSSL), the cipher-suite choices for constrained devices.; Zero-trust IoT — never trust the network, always verify the device; the post-perimeter security model.; A lab: secure the M4 ThingsBoard pipeline — enable TLS, rotate device credentials, demonstrate a man-in-the-middle attack succeeds without TLS and fails with TLS enabled; write a 1-page security review.	CO5
M6	**Edge AI, 5G+IoT, Digital Twins** Edge AI motivation — latency, privacy, bandwidth, offline operation; the trade-offs vs cloud AI.; TFLite Micro at depth — the canonical 50KB-class model for keyword spotting, anomaly detection, basic image classification on Cortex-M4.; TinyML deployment workflow — train in Python with TensorFlow, convert to TFLite, quantize, deploy to MCU.; 5G + IoT — network slicing for IoT, ultra-reliable-low-latency for industrial; mMTC (massive machine-type) slice.; Digital twins at intuition — the canonical mirror-physical-asset-in-cloud pattern; Azure Digital Twins, AWS IoT TwinMaker.; AIoT (AI + IoT) — the convergence; sensor-data → ML inference → actuation as the canonical industrial pattern.; AI-augmented IoT design — Claude / ChatGPT for protocol-decision brainstorming, schema design, dashboard SQL; the discipline of always testing on real devices.; A final-week lab: deploy a TFLite Micro anomaly-detection model on ESP32 reading vibration data from accelerometer; transmit via LoRaWAN to ThingsBoard; observe normal-vs-anomaly classification on dashboard; document end-to-end latency.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Internet of Things — A Hands-On Approach**	Arshdeep Bahga & Vijay Madisetti	Universities Press	2015
2	**Hands-On Internet of Things with MQTT**	Tim Pulver	Packt	2018
3	**LoRa and LoRaWAN — A Technical Overview**	Andreas Spiess	LoRa Alliance / Online (free)	2024
4	**Practical Internet of Things Security**	Brian Russell & Drew Van Duren	Packt	2018 (2nd Ed.)
5	**TinyML — Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers**	Pete Warden & Daniel Situnayake	O'Reilly	2019
6	**Industrial Internet of Things — Cybermanufacturing Systems**	Sabina Jeschke et al.	Springer	2017

#	Title	Authors	Publisher	Year
7	**Architecting the Industrial Internet — The Architect's Guide to Designing Industrial Internet Solutions**	Shyam Nath & Robert Stackowiak	Packt	2017

Language Skills for Engineers

LSFE REG · 26UC1102 PBL · 26UC1102E

L-T-P-S	0-0-4-0 / 3-0-0-0	Credits	2	Contact Hrs/Wk	4 / 3
Term (Reg · PBL)	S01 · T01	Mode	Regular & PBL	Category	HSS
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Use English with grammatical accuracy and clarity appropriate to a professional engineering context, recognising and correcting common errors and choosing the right register and tone.	BTL3 · Apply
CO2	Build and deploy a precise professional and technical vocabulary, choosing words for accuracy, tone and audience and avoiding jargon, wordiness and misuse.	BTL3 · Apply
CO3	Apply the principles of professional writing to draft clear, well-structured documents, using a disciplined draft-revise-edit process supported by AI feedback.	BTL6 · Create
CO4	Compose effective, etiquette-correct corporate emails and written correspondence for a wide range of real workplace purposes, tailoring tone and structure to audience and intent.	BTL6 · Create
CO5	Hold clear, confident professional conversations - introductions, discussions, telephone and video calls - using intelligible speech, active listening and appropriate spoken-English conventions.	BTL4 · Analyze
CO6	Apply corporate communication etiquette and professional presence across cultures, channels and difficult situations, and curate a professional communication identity.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO7	PO8	PO9	PO11
CO1	–	–	**H**	M
CO2	–	–	**H**	M
CO3	–	–	**H**	M
CO4	–	M	**H**	M
CO5	L	M	**H**	M
CO6	L	M	**H**	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - Foundations of English for the Engineer: Accuracy, Clarity & Register** Why language accuracy matters professionally: credibility, clarity and the cost of miscommunication in engineering work.; Sentence fundamentals: parts of speech in use, subject-verb agreement, tense consistency, and sentence types (simple, compound, complex).; Common error patterns of Indian/engineering English: articles, prepositions, agreement, run-ons and fragments, dangling modifiers, and direct translation from the mother tongue.; Punctuation for clarity: comma, semicolon, colon, apostrophe and the meaning they carry; punctuation that changes meaning.; Clarity principles: one idea per sentence, active vs passive voice and when each is appropriate, parallel structure, and avoiding ambiguity.; Register and tone: formal, neutral and informal English; matching register to audience and situation (a manager vs a peer vs a client).; Concision: removing redundancy, wordiness and filler; saying more with fewer words.; Standard varieties and intelligibility: British/American/Indian English conventions and writing/speaking to be understood across them.; Proofreading and self-editing fundamentals: a method for catching one's own errors.; Using AI as an accuracy tutor: how to ask an AI tool to explain (not just fix) an error, and how to learn from its corrections rather than copy them.	CO1

	Title & Topics	CO
M2	**Module 2 - Vocabulary & Word Precision** How professional vocabulary is built: roots, prefixes and suffixes; learning words in context rather than in isolation.; Precision in word choice: denotation vs connotation; commonly confused words (affect/effect, principal/principle, etc.); near-synonyms with different shades.; Professional and corporate vocabulary: the everyday lexicon of the workplace (meetings, deadlines, deliverables, stakeholders) used correctly.; Technical vs general vocabulary: when technical terms aid precision and when they become jargon that excludes the audience.; Explaining technical ideas to non-technical audiences: plain-English equivalents and analogies.; Collocations and natural usage: word partnerships that make English sound fluent (make/do, strong/heavy, etc.).; Idioms, phrasal verbs and professional expressions used appropriately (and the risk of misusing them).; Tone-carrying words: hedging and softening (could/might/perhaps), assertive vs tentative language, and positive vs negative framing.; Avoiding clichés, buzzwords and empty corporate-speak; writing with substance.; Using AI for vocabulary growth: generating contextual examples, checking nuance/collocation, and building a personal word bank - while verifying suggestions.	CO2
M3	**Module 3 - Professional Writing & Drafting** The qualities of effective professional writing: clarity, conciseness, completeness, correctness, coherence, courtesy and concreteness (the 7 Cs).; Audience and purpose analysis: writing for the reader, and the question 'what do I want the reader to know/feel/do?'.; Structuring writing: the paragraph (topic sentence, support, transition), logical flow, and the top-down 'bottom line up front' principle.; Coherence and cohesion: transitions, signposting, and given-to-new information flow.; Common short professional documents (principles of): notes, memos, messages, short reports and summaries - their purpose and conventions (detailed report/technical-document craft is developed further in later courses).; Plain-language writing and readability: short sentences, familiar words, and designing text to be scanned (headings, lists, white space).; The writing process: planning/outlining, drafting, revising for content and structure, then editing for language - and why these are separate passes.; Summarising and paraphrasing accurately without distortion or plagiarism; quoting and attributing correctly.; Tone in writing: professional courtesy, the 'you-attitude', and positive phrasing of negative messages.; Using AI as a drafting and revision partner: generating a first draft to react to, getting structural and clarity feedback, and critically editing AI output - with academic-integrity boundaries on when AI help is and isn't appropriate.	CO3
M4	**Module 4 - Corporate Email & Written Correspondence** Anatomy of a professional email: subject line, salutation, opening, body, call to action, closing and signature - and the job each part does.; Subject lines that work: specific, scannable, action-oriented.; Email for many purposes (conventions for each): requests, follow-ups and reminders, status/progress updates, scheduling and confirmations, declining or saying no politely, apologising, complaining effectively, thanking, introductions, and escalations.; Tone and register in email: formality levels, warmth vs directness, and reading the relationship with the recipient.; Email etiquette: To/Cc/Bcc use, reply-all discipline, response-time norms, threading, attachments, and forwarding etiquette.; Clarity and brevity in email: getting to the point, one purpose per email, and making the ask explicit.; Handling difficult messages in writing: delivering bad news, disagreeing, and de-escalating - politely and professionally.; Cross-cultural and global email: directness vs indirectness across cultures, honorifics, and writing for non-native readers.; Other correspondence conventions (overview): formal letters, official requests and professional messaging-app etiquette.; Using AI for correspondence: drafting and tone-adjusting an email with AI, then verifying facts, personalising, and protecting confidential information - the ethics of AI-assisted correspondence.	CO4
M5	**Module 5 - Spoken English & Professional Conversation** Spoken vs written English: how professional speech differs from writing; spontaneity, fillers and recovery.; Intelligibility over accent: clear articulation, word stress, sentence stress and intonation for being understood (not accent elimination).; Professional introductions and self-introduction: making a strong, concise first impression; introducing others.; Conversation skills: starting, sustaining and closing a conversation; turn-taking; small talk and professional rapport-building.; Active listening: signalling attention, paraphrasing to confirm, clarifying questions, and not interrupting.; Asking and answering professionally: open vs closed questions, answering concisely, and saying 'I don't know' gracefully.; Expressing opinions, agreeing and disagreeing politely; suggesting, requesting and refusing in speech.; Telephone and video-call etiquette: opening/closing a call, spelling/numbers clearly, managing audio issues, and on-camera presence.; Speaking in everyday work situations: stand-ups, one-on-ones, asking for help, giving a quick update (extended presentations/public speaking are developed in the T02 course).; Using AI as a spoken-English partner: practising conversations and pronunciation with AI voice/chat tools, getting feedback, and building speaking confidence through low-stakes repetition.	CO5

	Title & Topics	CO
M6	**Module 6 - Corporate Etiquette & Professional Presence** What professional etiquette is and why it signals competence and respect; first impressions and credibility.; Workplace communication norms: punctuality in responses, reliability of word, acknowledging messages, and respecting others' time and attention.; Meeting etiquette: preparing, contributing without dominating, not interrupting, note-taking, and following up - the communication side of meetings.; Channel etiquette: choosing the right channel (email vs chat vs call vs in-person), and the norms of each; messaging-app professionalism.; Cross-cultural communication: high- vs low-context cultures, directness, formality, names/titles, and avoiding culturally loaded expressions in a global team.; Digital professionalism and netiquette: tone in text, emoji/abbreviation appropriateness, and the permanence of written messages.; Handling difficult interactions with courtesy: disagreement, criticism (giving and receiving), saying no, and de-escalating tension.; Inclusive and respectful language: gender-neutral and bias-free communication; professional courtesy to all.; Professional presence and personal brand in communication: a polished LinkedIn presence, a professional self-introduction, and consistency between how one writes, speaks and behaves.; Using AI responsibly in professional communication: confidentiality, disclosure norms, authenticity, and not outsourcing one's professional voice entirely to AI.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Technical Communication: Principles and Practice**	Meenakshi Raman & Sangeeta Sharma	Oxford University Press (India)	2015 (3rd ed.)
2	**On Writing Well: The Classic Guide to Writing Nonfiction**	William Zinsser	Harper Perennial	2016 (anniversary ed.)
3	**The Elements of Style**	William Strunk Jr. & E. B. White	Pearson/Longman	2019 (4th ed.)
4	**Practical English Usage**	Michael Swan	Oxford University Press	2016 (4th ed.)
5	**Word Power Made Easy**	Norman Lewis	Penguin / Goyal	2014
6	**HBR Guide to Better Business Writing**	Bryan A. Garner	Harvard Business Review Press	2013

Machine Learning

MLRN REG · 26SC2110 PBL · 26SC2108E

L-T-P-S	2-0-4-0 / 3-0-6-0	Credits	4	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S03 · T04	Mode	Regular & PBL	Category	ESC
Pre-requisite	CFAI				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze the end-to-end ML system lifecycle — data → features → training → evaluation → deployment → monitoring — and trace one prediction request from a deployed model back through every preceding lifecycle stage.	BTL4 · Analyze
CO2	Apply linear supervised-learning models — linear regression, ridge, lasso, elastic net, logistic regression, multinomial logistic regression — and reason about regularisation, the bias-variance trade-off, and feature scaling.	BTL3 · Apply
CO3	Apply tree-based supervised-learning models — decision trees, random forests, gradient-boosted trees (XGBoost, LightGBM) — and reason about feature importance, splitting criteria, and ensemble variance reduction.	BTL3 · Apply
CO4	Apply unsupervised-learning techniques — k-means and hierarchical clustering, density-based clustering (DBSCAN), dimensionality reduction (PCA, t-SNE, UMAP), and anomaly detection — to discover structure in unlabelled data.	BTL3 · Apply
CO5	Analyze model performance using cross-validation, train/validation/test splits, classification metrics (accuracy, precision, recall, F1, ROC-AUC, PR-AUC), regression metrics (RMSE, MAE, R^2), calibration, and hyperparameter search (grid, random, Bayesian).	BTL4 · Analyze
CO6	Apply ML-engineering practices — feature stores, training-serving skew avoidance, model packaging, REST-based serving, basic monitoring for drift and performance degradation — to deploy a production-style ML system.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PSO1
CO1	H	H	H	–	–	H
CO2	H	H	H	–	–	H
CO3	H	–	H	–	–	H
CO4	H	M	–	–	–	H
CO5	H	H	–	H	–	H
CO6	H	–	H	–	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**PlacementPredict as a System — The ML Lifecycle** A working prediction service running on day one — students issue requests against PlacementPredict and see predictions before any internals.; The ML system lifecycle: raw data → cleaned dataset → engineered features → training set → trained model → packaged artifact → served endpoint → monitored production → retraining loop.; Where each lifecycle stage actually lives in code: data pipeline scripts, feature engineering modules, training notebooks, model registry, serving framework, monitoring dashboard.; The supervised vs unsupervised vs reinforcement learning split; what PlacementPredict is (supervised classification) and what it is not.; Tracing one prediction end-to-end: HTTP request → feature lookup → model inference → response; what changes for a batch prediction.; The training-serving boundary: the split that causes the most production bugs (training-serving skew); why feature stores exist.; A reading of the PlacementPredict codebase: data ingestion, feature pipeline, training script, serving FastAPI app, monitoring hooks.; What this course owns vs what comes downstream: classical ML and ML engineering here; deep learning in DLAC; generative models in GALE; LLMs in NLPL.	CO1

	Title & Topics	CO
M2	**Supervised Learning — Linear Models at Depth** Linear regression revisited from CFAI: closed-form normal equations, gradient-descent training, the comparison.; Regularisation: ridge (L2), lasso (L1), elastic net; the geometric picture of why L1 induces sparsity.; Logistic regression for binary classification: the sigmoid, the cross-entropy loss, the decision boundary as a hyperplane.; Multinomial logistic regression (softmax regression) for multi-class problems.; Feature scaling and standardisation: when it matters (gradient descent), when it doesn't (tree models in M3); the StandardScaler and MinMaxScaler.; Categorical features: one-hot encoding, ordinal encoding, target encoding, embedding-based encoding at preview level.; Handling missing values: deletion, mean/median imputation, model-based imputation, MissingIndicator features.; A worked example end-to-end: predict student placement outcome from academic features using ridge logistic regression; train, validate, interpret coefficients.	CO2
M3	**Supervised Learning — Tree-Based Models** Decision trees: greedy top-down construction, splitting criteria (Gini impurity, entropy, MSE for regression), stopping rules.; Tree pruning: pre-pruning by depth/leaf-size, post-pruning by cost-complexity (CCP); the bias-variance angle.; Random forests: bagging, feature subsampling, out-of-bag error, the variance-reduction argument.; Boosting: AdaBoost at intuition level, gradient boosting as functional gradient descent.; XGBoost and LightGBM as the modern industrial workhorses: histogram-based splitting, leaf-wise growth, regularisation.; Feature importance: gini importance, permutation importance, SHAP values at preview level.; Tree models on tabular data: why they routinely beat neural networks on structured datasets (the Kaggle reality).; A worked example end-to-end: same placement-prediction task with a gradient-boosted tree; compare against linear baseline; interpret feature importance.	CO3
M4	**Unsupervised Learning** K-means revisited from CFAI with the modern Lloyd-algorithm view; choosing K via the elbow method and the silhouette score.; K-means++ initialisation revisited; mini-batch k-means for large datasets.; Hierarchical clustering: agglomerative bottom-up, the linkage choices (single, complete, average, Ward), reading dendrograms.; Density-based clustering with DBSCAN: ε and minPts, identifying outliers naturally, why DBSCAN handles non-convex clusters.; Dimensionality reduction revisited: PCA from CFAI in production-style usage; pitfalls when applied without standardisation.; t-SNE and UMAP as the visualisation workhorses; their hyperparameters; why distances in t-SNE/UMAP space are not meaningful.; Anomaly detection: isolation forest, one-class SVM at intuition level, autoencoder-based anomaly detection at preview level (deep dive in DLAC).; Practical pipeline: when to cluster vs when to use a supervised model; cluster labels as features for a downstream supervised model.	CO4
M5	**Model Evaluation, Selection, and Calibration** Train/validation/test split discipline: the three-way split, why a test set must be held out from every modelling decision.; k-fold and stratified k-fold cross-validation; nested CV when both hyperparameter selection and unbiased evaluation are needed.; Classification metrics: accuracy (and when it lies), precision/recall/F1, the confusion matrix, ROC-AUC, PR-AUC; class-imbalance impact.; Regression metrics: RMSE, MAE, MAPE, R^2; the choice between RMSE and MAE depending on outlier sensitivity.; Probability calibration: why a 0.7 prediction should mean "70% likely"; Platt scaling, isotonic regression; reliability diagrams.; Hyperparameter search: grid search, random search (Bergstra-Bengio), Bayesian optimisation with Gaussian processes at preview level.; Learning curves and validation curves: diagnosing under-fitting, over-fitting, data scarcity from a single plot.; Statistical significance for model comparison at intro level: McNemar's test for classifiers, paired bootstrap for regressors.	CO5
M6	**ML Engineering — From Notebook to Production** The notebook-to-production gap: why a Jupyter notebook is not a deployable system; the ML-engineering rituals that close the gap.; Feature engineering at scale: feature stores (Feast at conceptual level), the offline-online split, point-in-time correctness.; Training-serving skew: the canonical bug class in production ML; how to detect and prevent it.; Model packaging: pickle pitfalls, ONNX as the portable format, container-based serving.; Serving frameworks at intro level: FastAPI for REST, gRPC for low-latency, batch vs online prediction.; Monitoring in production: predictive performance, latency, throughput, data drift, concept drift, label drift.; Retraining strategies: scheduled retraining, drift-triggered retraining, online learning at preview level.; A glance at MLOps platforms: MLflow for experiment tracking, DVC for data versioning, the basic engineer's toolkit. (Deep MLOps and platform engineering belong to GALE and SDSR.)	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow**	Aurélien Géron	O'Reilly	2022 (3rd Ed.)
2	**The Elements of Statistical Learning — Data Mining, Inference, and Prediction**	Trevor Hastie, Robert Tibshirani & Jerome Friedman	Springer	2017 (2nd Ed., corrected 12th printing)

#	Title	Authors	Publisher	Year
3	**An Introduction to Statistical Learning — with Applications in Python**	Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani & Jonathan Taylor	Springer	2023
4	**Pattern Recognition and Machine Learning**	Christopher M. Bishop	Springer	2006
5	**Designing Machine Learning Systems — An Iterative Process for Production-Ready Applications**	Chip Huyen	O'Reilly	2022
6	**Building Machine Learning Powered Applications**	Emmanuel Ameisen	O'Reilly	2020
7	**Machine Learning Engineering**	Andriy Burkov	True Positive Inc.	2020

Mathematics for AI

MTAI REG · 26MT1202 PBL · 26MT1202E

L-T-P-S	2-2-0-0 / 3-3-0-0	Credits	4	Contact Hrs/Wk	4 / 6
Term (Reg · PBL)	S02 · T02	Mode	Regular & PBL	Category	BSC
Pre-requisite	MTCP				

Course Outcomes (COs)

	Statement	BTL
CO1	Represent and manipulate data and transformations with vectors and matrices, and reason geometrically about vector spaces - the language in which AI represents data and models.	BTL3 · Apply
CO2	Apply eigen-analysis and matrix decompositions to understand variance, dimensionality and structure in data - the mathematics behind techniques like PCA.	BTL4 · Analyze
CO3	Use single- and multivariable calculus - derivatives, partial derivatives, gradients and the chain rule - to quantify how a model's output and error change, the basis of learning.	BTL3 · Apply
CO4	Formulate and solve optimization problems - especially gradient-based minimisation of a loss - which is how AI models are trained.	BTL6 · Create
CO5	Reason quantitatively under uncertainty using probability - random variables, distributions, expectation and Bayes' theorem - the basis of probabilistic AI.	BTL3 · Apply
CO6	Integrate linear algebra, calculus, optimization and probability to follow and explain the end-to-end mathematics of a simple learning algorithm.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO11
CO1	**H**	**H**	M	M	M	M
CO2	**H**	**H**	M	M	M	M
CO3	**H**	**H**	M	M	M	M
CO4	**H**	**H**	M	**H**	M	M
CO5	**H**	**H**	M	M	M	M
CO6	**H**	**H**	M	M	M	**H**

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - Linear Algebra I: Vectors, Matrices & Linear Systems** Vectors as data: vectors in R^n, vector operations, linear combinations, and the geometric meaning (a data point / feature vector as a vector).; Dot product, norms and distance: length, angle, cosine similarity - and why these measure similarity between data points in AI.; Matrices as data and as transformations: a dataset as a matrix; matrix-vector and matrix-matrix multiplication and their meaning.; Linear transformations: how a matrix maps vectors (rotation, scaling, projection); composition of transformations.; Systems of linear equations: representation Ax=b, Gaussian elimination, and solution existence/uniqueness.; Matrix inverse and determinant: meaning, computation for small matrices, and when an inverse exists.; Vector spaces, subspaces, span, linear independence and basis (foundational, applied not abstract-heavy).; Rank and the column/row space; the idea of dimensionality of data.; Orthogonality and orthonormal bases; projection onto a subspace (the basis of least-squares, used in Module 3).; AI/ML meaning throughout: features as vectors, datasets as matrices, similarity, and why linear algebra is the language of AI.	CO1

	Title & Topics	CO
M2	**Module 2 - Linear Algebra II: Eigenvalues, Decompositions & Their AI Meaning** Eigenvalues and eigenvectors: definition, computation for small matrices, and geometric meaning (directions a transformation preserves).; Diagonalisation and its use; symmetric matrices and their special properties.; Quadratic forms and positive-definiteness (foundational) - and why they appear in optimization (Module 4).; Singular Value Decomposition (SVD): concept, what the singular values/vectors mean, and its central role in data.; Principal Component Analysis (PCA) - the mathematics: variance, covariance (link to Module 5), and how eigen/SVD finds the directions of greatest variance.; Dimensionality reduction conceptually: why high-dimensional data can often be compressed, and what is preserved/lost.; Matrix factorisation as a recurring AI idea (recommendation, embeddings) - at a mathematical-foundation level.; Numerical reality (awareness): why exact hand-computation gives way to algorithms for large matrices (verified with NumPy).; Norms of matrices and conditioning (awareness): why some computations are sensitive.; AI/ML meaning throughout: PCA, embeddings and why eigen/SVD underlie how AI finds structure in data.	CO2
M3	**Module 3 - Calculus for Learning: Derivatives, Gradients & the Chain Rule** Limits, continuity and the derivative (consolidation): the derivative as a rate of change and as a slope.; Differentiation rules and the single-variable chain rule, reframed as the engine of sensitivity analysis.; Functions of several variables: surfaces, level sets, and the idea of a multivariable function (a model's error as a function of its parameters).; Partial derivatives: how output changes with one variable at a time.; The gradient vector: direction of steepest ascent/descent, and its central role in training.; The multivariable chain rule: composing derivatives - the mathematical heart of backpropagation (concept, not the algorithm).; Jacobian and Hessian (foundational): first- and second-order multivariable derivative information and what they tell us.; Taylor approximation (single and intro multivariable): approximating functions locally - the basis of many optimization methods.; Integration essentials and the area-under-a-curve idea, with the probability link (expected values, densities in Module 5).; AI/ML meaning throughout: the loss function, why gradients drive learning, and a first glimpse of backpropagation as the chain rule.	CO3
M4	**Module 4 - Optimization: Minimising Loss & Training Models** Optimization as the core of learning: 'training a model' = minimising a loss function; framing problems as optimization.; Unconstrained optimization: critical points, the role of the gradient (=0) and the Hessian (convexity/curvature) in finding minima.; Convexity: convex sets and functions, why convex problems are 'nice', and recognising convexity (foundational).; Gradient descent: the algorithm by hand - update rule, learning rate, and convergence intuition.; Variants conceptually: stochastic gradient descent, momentum (awareness) - why and when, without deep algorithmics.; Least squares as optimization: deriving the normal equations (linking to projection from Module 1) - a fully worked AI-relevant example.; Constrained optimization (foundational): the idea of constraints and Lagrange multipliers at an introductory level.; Regularisation as an optimization idea (foundational): penalising complexity (L1/L2) - the mathematics, link to ML later.; Common pitfalls: local vs global minima, learning-rate effects, and what can go wrong (intuition for debugging models).; AI/ML meaning throughout: gradient descent IS how neural networks and many models learn; this module is the mathematical core of training.	CO4
M5	**Module 5 - Probability: Reasoning Under Uncertainty** Probability foundations: sample spaces, events, axioms, and conditional probability (a fresh, AI-oriented treatment).; Independence and the multiplication rule; the law of total probability.; Bayes' theorem: the mathematics of updating belief with evidence - the foundation of Bayesian/probabilistic AI and the naive-Bayes classifier (concept).; Random variables: discrete and continuous; probability mass and density functions; cumulative distributions.; Expectation, variance and standard deviation: the meaning and computation - and their role as model summaries.; Key distributions and where AI meets them: Bernoulli/Binomial, Uniform, Gaussian/Normal (and why the Normal is everywhere), Poisson (awareness).; The Gaussian in depth (foundational): its parameters, shape and why it dominates ML; standardisation.; Joint, marginal and conditional distributions; covariance and correlation (linking to PCA in Module 2).; The Law of Large Numbers and the Central Limit Theorem (conceptual): why sampling and averages behave predictably.; AI/ML meaning throughout: classification as probability, likelihood, uncertainty estimates - and the boundary to statistics/inference, which is Mathematics for Data Science (T03).	CO5

	Title & Topics	CO
M6	**Module 6 - Putting It Together: The Mathematics of a Learning Algorithm** The data-to-model pipeline mathematically: data as matrices/vectors → a model as a function → a loss → optimization → a trained model.; Linear regression end-to-end: model, squared-error loss, solving by normal equations (linear algebra) AND by gradient descent (calculus+optimization) - one problem, two mathematical routes.; Logistic regression (foundational): the sigmoid, a probabilistic interpretation (probability), and training by gradient descent - integrating all four strands.; A single neuron and a tiny neural network as mathematics: weights as a matrix, a forward pass as linear algebra + a nonlinearity, the loss, and gradients via the chain rule (the mathematical skeleton of deep learning).; Why these pieces fit: seeing linear algebra (representation), calculus (sensitivity), optimization (learning) and probability (uncertainty) working together.; Reading the mathematics in ML documentation/papers: decoding the notation now that the pieces are known (a transferable skill).; Numerical and practical awareness: why libraries (NumPy and ML frameworks) implement this, and what the maths tells you when a model misbehaves.; The bridge to the next courses: how Computational Foundations for AI (T03) implements these algorithms and Mathematics for Data Science (T03) adds statistical depth.; Limits of this course: what is foundational here vs developed later (statistics, advanced optimization, full deep-learning theory).; AI/ML meaning throughout: this module proves the course's promise - AI is mathematics the student can now follow end-to-end.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Mathematics for Machine Learning**	Marc Peter Deisenroth, A. Aldo Faisal & Cheng Soon Ong	Cambridge University Press	2020
2	**Introduction to Linear Algebra**	Gilbert Strang	Wellesley-Cambridge Press	2023 (6th ed.)
3	**Linear Algebra and Its Applications**	David C. Lay, Steven Lay & Judi McDonald	Pearson	2021 (6th ed.)
4	**Calculus: Early Transcendentals**	James Stewart, Daniel Clegg & Saleem Watson	Cengage	2020 (9th ed.)
5	**A First Course in Probability**	Sheldon Ross	Pearson	2019 (10th ed.)
6	**Convex Optimization (selected intro chapters)**	Stephen Boyd & Lieven Vandenberghe	Cambridge University Press (free online)	2004

Mathematics for Computation

MTCP REG · 26MT1101 PBL · 26MT1101E

L-T-P-S	2-2-0-0 / 3-3-0-0	Credits	4	Contact Hrs/Wk	4 / 6
Term (Reg · PBL)	S01 · T01	Mode	Regular & PBL	Category	BSC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply propositional and predicate logic and construct rigorous mathematical proofs using standard proof techniques, building the computational-logic foundation for reasoning about correctness.	BTL4 · Analyze
CO2	Use set theory, relations and functions to model and reason about structured collections, equivalences, orderings and mappings that underpin data, types and access control.	BTL4 · Analyze
CO3	Apply combinatorial principles to count, arrange and bound discrete configurations, reasoning about key-spaces, complexity of choices, and collision probabilities at a structural level.	BTL4 · Analyze
CO4	Apply number theory and modular arithmetic to reason about divisibility, primes, congruences and the mathematical machinery underlying modern public-key cryptography.	BTL6 · Create
CO5	Model systems and networks with graphs and trees and reason about connectivity, traversal structure, colouring and spanning properties that underpin networks, dependencies and routing.	BTL4 · Analyze
CO6	Reason with algebraic and discrete structures - Boolean algebra, recurrence relations, and elementary groups/fields - that underpin digital logic, error-correcting codes, pseudo-randomness and cryptography.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO11
CO1	H	H	M	L	L	M
CO2	H	H	M	M	L	M
CO3	H	H	M	M	L	M
CO4	H	H	M	M	M	M
CO5	H	H	M	M	M	M
CO6	H	M	M	M	M	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - Mathematical Logic & Proof Techniques** Propositions, logical connectives (negation, conjunction, disjunction, implication, biconditional) and truth tables.; Tautologies, contradictions, contingencies; logical equivalence and the principal laws (De Morgan, distributive, contrapositive).; Conditional and biconditional reasoning; converse, inverse, contrapositive; necessary vs sufficient conditions.; Rules of inference and validity of arguments (modus ponens, modus tollens, syllogism, resolution) - the logical core of automated reasoning.; Predicates and quantifiers (universal, existential); nested quantifiers and negation of quantified statements.; Normal forms (CNF, DNF) and their role in logic-based computation (conceptual link to SAT, circuits, query logic).; Methods of proof: direct proof, proof by contraposition, proof by contradiction, proof by cases, and disproof by counterexample.; Mathematical induction and strong induction; the well-ordering principle.; Reasoning about program/protocol correctness informally via logic (pre/post-conditions and invariants stated as logical assertions - no code).; Common proof pitfalls and the standards of a rigorous mathematical argument.	CO1

	Title & Topics	CO
M2	**Module 2 - Set Theory, Relations & Functions** Sets, subsets, power sets, set operations (union, intersection, difference, complement) and Venn reasoning; set identities and their proofs.; Cartesian products; cardinality of finite sets; the inclusion-exclusion principle (foundational form).; Countable vs uncountable sets; Cantor's diagonalisation argument (conceptual link to limits of computation).; Relations: definition, representation (matrix and digraph), and properties (reflexive, symmetric, antisymmetric, transitive).; Equivalence relations and partitions; equivalence classes (link to hashing buckets, data partitioning).; Partial orders, total orders, Hasse diagrams; least/greatest elements and lattices (foundation for type hierarchies and dependency ordering).; Closure of relations (reflexive, symmetric, transitive closure) and their meaning.; Functions: domain, codomain, range; injective, surjective, bijective functions; composition and inverse.; Special functions relevant to computing: floor, ceiling, modulo as a function, and hashing functions as set-to-set mappings (modelled mathematically).; Pigeonhole principle and its surprising consequences (collision arguments foundational to hashing and cryptography).	CO2
M3	**Module 3 - Combinatorics & Counting** The fundamental counting principles: sum rule and product rule; counting by cases.; Permutations and combinations; permutations with repetition; circular permutations.; Combinations with repetition; counting selections and distributions (stars-and-bars).; The binomial theorem and binomial coefficients; Pascal's identity and combinatorial identities.; The inclusion-exclusion principle (general form) and derangements.; The pigeonhole principle (generalised) and its counting consequences.; Counting and key-space sizing: how counting bounds the size of password/key spaces (mathematical sizing, link to cryptographic strength).; The birthday problem and collision counting (combinatorial reasoning foundational to hash collisions and cryptographic attacks - argued by counting, not probability theory).; Recurrence relations as counting tools: formulating recurrences for combinatorial quantities.; Generating-function viewpoint (introductory) as a counting device; Catalan numbers as a structural counting example.	CO3
M4	**Module 4 - Number Theory & its Cryptographic Foundations** Divisibility, the division algorithm, GCD and LCM; the Euclidean algorithm and the extended Euclidean algorithm (as mathematical procedures, hand-executed).; Primes, the fundamental theorem of arithmetic, and the infinitude of primes (Euclid's proof).; Modular arithmetic: congruences, residue classes, and arithmetic in Z_n.; Properties of congruences; modular addition, multiplication and exponentiation (fast exponentiation by squaring as a mathematical method).; Modular inverses and solving linear congruences; the Chinese Remainder Theorem.; Fermat's Little Theorem and Euler's theorem; the Euler totient function.; Primality and the mathematical basis of primality reasoning (Fermat test idea - mathematically, not as code).; The mathematics of public-key cryptography: derivation and correctness proof of RSA (key generation, encryption, decryption) from Euler's theorem.; Discrete logarithm problem and the Diffie-Hellman key-exchange mathematics (why hardness gives security).; Hash-function and one-way-function mathematics conceptually; how number-theoretic hardness assumptions underpin contemporary security.	CO4
M5	**Module 5 - Graph Theory & Trees** Graphs: definitions, types (simple, multigraph, directed, weighted), degree, and the handshaking lemma.; Graph representations: adjacency matrix and adjacency list (as mathematical representations) and isomorphism of graphs.; Walks, paths, cycles, connectivity, and components; reasoning about reachability.; Euler paths and circuits (with the existence theorem and proof) and Hamiltonian paths and cycles (existence reasoning) - link to routing and the structure of hard problems.; Shortest-path and minimum-spanning-tree as mathematical structures and properties (existence/optimality reasoning; not algorithm complexity, which is DSA-1's).; Planar graphs, Euler's formula, and graph colouring; the chromatic number (link to register allocation, scheduling, frequency assignment - structurally).; Trees: definitions, properties, and the count of edges; rooted trees, binary trees, and tree terminology (as mathematical objects).; Spanning trees and their counting (Cayley's formula, conceptually) and the cut/cycle properties.; Bipartite graphs and matchings (Hall's theorem, conceptual) - link to assignment problems.; Graphs in technology, reasoned mathematically: key-distribution topologies, social/network structure, and dependency graphs.	CO5
M6	**Module 6 - Algebraic & Discrete Structures** Boolean algebra: axioms, Boolean functions, and identities; duality.; Representation and simplification of Boolean functions (algebraic simplification and Karnaugh-map reasoning) - the mathematics behind digital logic and circuit minimisation.; Recurrence relations: formulation, and solving linear homogeneous recurrences with constant coefficients (characteristic-equation method).; Solving non-homogeneous recurrences; using recurrences to model discrete processes (link to algorithm cost equations - mathematically, not Big-O).; Algebraic structures: binary operations, semigroups, monoids and groups; properties (closure, associativity, identity, inverse).; Subgroups, cyclic groups, and the order of an element; modular groups (Z_n, +) and (Z_p*, x) as worked examples linking back to number theory.; Introduction to rings and fields; finite fields GF(p) and the idea of GF(2^n) (conceptual) - the algebra behind AES and coding.; Mathematics of error-detecting and error-correcting codes: parity, Hamming distance, and the Hamming bound (linear-code reasoning, structurally).; Generating pseudo-randomness mathematically: linear congruential and linear-feedback structures (as algebraic recurrences).; Synthesis: how Boolean algebra, finite fields and recurrences jointly underpin contemporary cryptography, coding and digital design.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Discrete Mathematics and Its Applications**	Kenneth H. Rosen	McGraw-Hill	2019 (8th ed.)
2	**Discrete Mathematics with Applications**	Susanna S. Epp	Cengage	2019 (5th ed.)
3	**Concrete Mathematics: A Foundation for Computer Science**	Ronald Graham, Donald Knuth & Oren Patashnik	Addison-Wesley	1994 (2nd ed.)
4	**Elements of Discrete Mathematics**	C. L. Liu & D. P. Mohapatra	McGraw-Hill	2017 (4th ed.)
5	**An Introduction to the Theory of Numbers**	G. H. Hardy & E. M. Wright	Oxford University Press	2008 (6th ed.)
6	**Introduction to Graph Theory**	Douglas B. West	Pearson	2017 (2nd ed.)

Mathematics for Electrical Systems

MTES 26MT1305

L-T-P-S	2-2-0-0	Credits	4	Contact Hrs/Wk	4
Semester	S03	Mode	Regular	Category	BSC
Pre-requisite	MTCP				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply complex analysis - complex numbers, functions and the phasor representation - to AC circuit and system analysis, and verify computationally.	BTL4 · Analyze
CO2	Solve differential equations that model electrical circuits and systems, and verify computationally.	BTL6 · Create
CO3	Apply the Laplace transform and s-domain methods to network analysis and transfer functions, and verify computationally.	BTL6 · Create
CO4	Apply Fourier analysis - Fourier series and the Fourier transform - to signals and frequency-domain analysis, and verify computationally.	BTL4 · Analyze
CO5	Apply vector calculus to electromagnetic fields - gradient, divergence, curl and the structure behind Maxwell's relations - and verify computationally.	BTL4 · Analyze
CO6	Apply numerical methods and probabilistic mathematics - numerical solving, and probability/random signals for noise and reliability - in electrical engineering, and verify computationally.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO11
CO1	**H**	**H**	–	–	M	–
CO2	**H**	**H**	M	–	M	–
CO3	**H**	**H**	M	–	M	–
CO4	**H**	**H**	–	–	M	–
CO5	**H**	**H**	–	–	M	–
CO6	**H**	**H**	–	M	**H**	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - Complex Analysis & Phasors** Complex numbers: rectangular and polar/exponential forms; Euler's formula.; Operations, powers and roots of complex numbers (De Moivre).; Functions of a complex variable; the complex exponential.; Phasors: representing sinusoids as complex quantities - the key to AC analysis.; Impedance and admittance as complex quantities; phasor circuit analysis.; Analyticity and the Cauchy-Riemann equations (introduction).; Complex frequency and its meaning (a bridge to the s-domain).; Applications: AC steady-state circuit analysis with phasors.; The geometry of complex numbers (Argand plane) and its circuit interpretation.; Computational verification (does not replace the mathematics): solve a phasor AC problem by hand, then confirm with SymPy/NumPy complex arithmetic and plot the phasor diagram - the derive-then-verify habit, introduced.	CO1
M2	**Module 2 - Differential Equations for Circuits & Systems** First-order ODEs: RC and RL circuit transients; time constants.; Second-order ODEs: RLC circuits; the characteristic equation.; Natural and forced response; damping regimes (under/critical/over).; Initial conditions and the complete response of a circuit.; Sinusoidal steady-state from the differential-equation viewpoint (linking to phasors).; Systems of ODEs for coupled circuits/state variables (intro).; Stability of a linear electrical system (intro, via roots/eigenvalues).; Modelling an electrical system from its governing equations.; Resonance in RLC circuits and its mathematics.; Computational verification: solve a circuit ODE by hand, then confirm the closed form with SymPy and plot the transient/response - confirmation, not substitution.	CO2

	Title & Topics	CO
M3	**Module 3 - Laplace Transforms & Network Analysis** The Laplace transform: definition, existence and transforms of common functions.; Properties: linearity, shifting, derivatives/integrals - turning ODEs into algebra.; The inverse Laplace transform and partial fractions.; Solving circuit ODEs with the Laplace transform; initial conditions in the s-domain.; Impedance in the s-domain; transformed circuit analysis.; Transfer functions: definition, poles and zeros, and their meaning.; Network analysis: the s-domain view of networks and two-port parameters (intro).; Step and impulse response of a network.; The link between pole locations and stability/transient behaviour.; Computational verification: compute a transform/transfer function by hand, then confirm with SymPy and plot the pole-zero map and response - confirmation and intuition.	CO3
M4	**Module 4 - Fourier Analysis & Signals** Fourier series: representing a periodic signal; trigonometric and exponential forms.; Computing Fourier coefficients; convergence and the Gibbs phenomenon.; The frequency spectrum of a periodic signal.; The Fourier transform: from series to transform; the continuous spectrum.; Properties of the Fourier transform and common transform pairs.; The frequency response of a system (linking to transfer functions).; Filtering and bandwidth in the frequency domain (mathematical view).; Sampling and an introduction to the discrete/Fast Fourier Transform (DFT/FFT).; Applications: signal content, harmonics in power systems, communication signals.; Computational verification: compute a Fourier series/transform by hand, then confirm with SymPy and compute an FFT spectrum with NumPy - seeing the frequency content, not replacing the analysis.	CO4
M5	**Module 5 - Vector Calculus & Electromagnetic Fields** Vector fields; the gradient, divergence and curl and their physical meaning.; Line, surface and volume integrals in field problems.; The divergence (Gauss) and Stokes theorems and their use.; Electrostatic fields: Gauss's law and potential (mathematical treatment).; Magnetostatic fields: Ampere's law (mathematical treatment).; The structure behind Maxwell's equations (the four relations and what each says).; Coordinate systems (cylindrical, spherical) for field geometry.; Applications: fields in machines, transmission lines and antennas (mathematical view).; The wave equation for electromagnetic waves (introduction).; Computational verification: compute a field operator/integral by hand, then confirm symbolically with SymPy and visualise the field - the mathematics first, computation as check and picture.	CO5
M6	**Module 6 - Numerical & Probabilistic Methods for EE** Why numerical methods: circuit/field equations that have no closed form.; Root-finding (Newton-Raphson) for nonlinear circuit equations.; Numerical solution of circuit ODEs (Euler, Runge-Kutta).; Numerical methods for power-flow/network equations (intro).; Probability essentials for EE: random variables, expectation, variance, distributions.; Random signals and noise: the mathematics of noise in electrical systems.; Power spectral density (introduction) and the frequency view of noise.; Reliability mathematics: failure distributions and basic system reliability.; Monte-Carlo estimation for an electrical/reliability problem.; Computational verification: implement a numerical method by hand for a few steps, then run it (NumPy/SciPy); simulate noise/reliability and confirm - and the boundary to later power/control/communication courses.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Advanced Engineering Mathematics**	Erwin Kreyszig	Wiley	2011 (10th ed.)
2	**Signals and Systems**	Oppenheim, Willsky & Nawab	Pearson	1996 (2nd ed.)
3	**Engineering Circuit Analysis**	Hayt, Kemmerly & Durbin	McGraw-Hill	2018
4	**Engineering Electromagnetics**	William H. Hayt & John A. Buck	McGraw-Hill	2018
5	**Probability, Random Variables and Stochastic Processes**	Papoulis & Pillai	McGraw-Hill	2002

Mathematics for Electrical Systems

MTEE 26MT1305E

L-T-P-S	3-3-0-0	Credits	4	Contact Hrs/Wk	6
Trimester	T03	Mode	PBL (Project-Based)	Category	BSC
Pre-requisite	MTCP				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply complex analysis - complex numbers, functions and the phasor representation - to AC circuit and system analysis, and verify computationally.	BTL4 · Analyze
CO2	Solve differential equations that model electrical circuits and systems, and verify computationally.	BTL6 · Create
CO3	Apply the Laplace transform and s-domain methods to network analysis and transfer functions, and verify computationally.	BTL6 · Create
CO4	Apply Fourier analysis - Fourier series and the Fourier transform - to signals and frequency-domain analysis, and verify computationally.	BTL4 · Analyze
CO5	Apply vector calculus to electromagnetic fields - gradient, divergence, curl and the structure behind Maxwell's relations - and verify computationally.	BTL4 · Analyze
CO6	Apply numerical methods and probabilistic mathematics - numerical solving, and probability/random signals for noise and reliability - in electrical engineering, and verify computationally.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO11
CO1	**H**	**H**	–	–	M	–
CO2	**H**	**H**	M	–	M	–
CO3	**H**	**H**	M	–	M	–
CO4	**H**	**H**	–	–	M	–
CO5	**H**	**H**	–	–	M	–
CO6	**H**	**H**	–	M	**H**	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - Complex Analysis & Phasors** Complex numbers: rectangular and polar/exponential forms; Euler's formula.; Operations, powers and roots of complex numbers (De Moivre).; Functions of a complex variable; the complex exponential.; Phasors: representing sinusoids as complex quantities - the key to AC analysis.; Impedance and admittance as complex quantities; phasor circuit analysis.; Analyticity and the Cauchy-Riemann equations (introduction).; Complex frequency and its meaning (a bridge to the s-domain).; Applications: AC steady-state circuit analysis with phasors.; The geometry of complex numbers (Argand plane) and its circuit interpretation.; Computational verification (does not replace the mathematics): solve a phasor AC problem by hand, then confirm with SymPy/NumPy complex arithmetic and plot the phasor diagram - the derive-then-verify habit, introduced.	CO1
M2	**Module 2 - Differential Equations for Circuits & Systems** First-order ODEs: RC and RL circuit transients; time constants.; Second-order ODEs: RLC circuits; the characteristic equation.; Natural and forced response; damping regimes (under/critical/over).; Initial conditions and the complete response of a circuit.; Sinusoidal steady-state from the differential-equation viewpoint (linking to phasors).; Systems of ODEs for coupled circuits/state variables (intro).; Stability of a linear electrical system (intro, via roots/eigenvalues).; Modelling an electrical system from its governing equations.; Resonance in RLC circuits and its mathematics.; Computational verification: solve a circuit ODE by hand, then confirm the closed form with SymPy and plot the transient/response - confirmation, not substitution.	CO2

	Title & Topics	CO
M3	**Module 3 - Laplace Transforms & Network Analysis** The Laplace transform: definition, existence and transforms of common functions.; Properties: linearity, shifting, derivatives/integrals - turning ODEs into algebra.; The inverse Laplace transform and partial fractions.; Solving circuit ODEs with the Laplace transform; initial conditions in the s-domain.; Impedance in the s-domain; transformed circuit analysis.; Transfer functions: definition, poles and zeros, and their meaning.; Network analysis: the s-domain view of networks and two-port parameters (intro).; Step and impulse response of a network.; The link between pole locations and stability/transient behaviour.; Computational verification: compute a transform/transfer function by hand, then confirm with SymPy and plot the pole-zero map and response - confirmation and intuition.	CO3
M4	**Module 4 - Fourier Analysis & Signals** Fourier series: representing a periodic signal; trigonometric and exponential forms.; Computing Fourier coefficients; convergence and the Gibbs phenomenon.; The frequency spectrum of a periodic signal.; The Fourier transform: from series to transform; the continuous spectrum.; Properties of the Fourier transform and common transform pairs.; The frequency response of a system (linking to transfer functions).; Filtering and bandwidth in the frequency domain (mathematical view).; Sampling and an introduction to the discrete/Fast Fourier Transform (DFT/FFT).; Applications: signal content, harmonics in power systems, communication signals.; Computational verification: compute a Fourier series/transform by hand, then confirm with SymPy and compute an FFT spectrum with NumPy - seeing the frequency content, not replacing the analysis.	CO4
M5	**Module 5 - Vector Calculus & Electromagnetic Fields** Vector fields; the gradient, divergence and curl and their physical meaning.; Line, surface and volume integrals in field problems.; The divergence (Gauss) and Stokes theorems and their use.; Electrostatic fields: Gauss's law and potential (mathematical treatment).; Magnetostatic fields: Ampere's law (mathematical treatment).; The structure behind Maxwell's equations (the four relations and what each says).; Coordinate systems (cylindrical, spherical) for field geometry.; Applications: fields in machines, transmission lines and antennas (mathematical view).; The wave equation for electromagnetic waves (introduction).; Computational verification: compute a field operator/integral by hand, then confirm symbolically with SymPy and visualise the field - the mathematics first, computation as check and picture.	CO5
M6	**Module 6 - Numerical & Probabilistic Methods for EE** Why numerical methods: circuit/field equations that have no closed form.; Root-finding (Newton-Raphson) for nonlinear circuit equations.; Numerical solution of circuit ODEs (Euler, Runge-Kutta).; Numerical methods for power-flow/network equations (intro).; Probability essentials for EE: random variables, expectation, variance, distributions.; Random signals and noise: the mathematics of noise in electrical systems.; Power spectral density (introduction) and the frequency view of noise.; Reliability mathematics: failure distributions and basic system reliability.; Monte-Carlo estimation for an electrical/reliability problem.; Computational verification: implement a numerical method by hand for a few steps, then run it (NumPy/SciPy); simulate noise/reliability and confirm - and the boundary to later power/control/communication courses.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Advanced Engineering Mathematics**	Erwin Kreyszig	Wiley	2011 (10th ed.)
2	**Signals and Systems**	Oppenheim, Willsky & Nawab	Pearson	1996 (2nd ed.)
3	**Engineering Circuit Analysis**	Hayt, Kemmerly & Durbin	McGraw-Hill	2018
4	**Engineering Electromagnetics**	William H. Hayt & John A. Buck	McGraw-Hill	2018
5	**Probability, Random Variables and Stochastic Processes**	Papoulis & Pillai	McGraw-Hill	2002

Modern Power Systems & Smart Grid Technologies

MPSG REG · 26EE2307 PBL · 26EE2307E

L-T-P-S	2-0-4-0 / 3-0-6-0	Credits	4	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S05 · T06	Mode	Regular & PBL	Category	PCC
Pre-requisite	EGTD				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze power-system representation — the per-unit system, single-line diagrams and the bus admittance/impedance matrices — to model an interconnected network.	BTL4 · Analyze
CO2	Apply load-flow methods — Gauss-Seidel, Newton-Raphson and fast-decoupled — to compute the operating state of a transmission and distribution network.	BTL3 · Apply
CO3	Analyze symmetrical and unsymmetrical faults and rotor-angle stability (the swing equation and the equal-area criterion) for secure grid operation.	BTL4 · Analyze
CO4	Apply smart-grid sensing and communication — PMUs, WAMS, SCADA/EMS and grid communication standards — to build observable, controllable grids.	BTL3 · Apply
CO5	Analyze the integration of distributed energy resources, microgrids and energy storage into a modern distribution system.	BTL4 · Analyze
CO6	Apply grid operation and analytics — economic dispatch, demand-side management and AI-assisted forecasting — to a smart-grid case study.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO7	PSO1
CO1	H	H	–	–	–	–	H
CO2	H	H	–	–	M	–	H
CO3	H	H	–	M	–	–	H
CO4	H	–	M	–	H	–	H
CO5	H	H	–	–	–	H	H
CO6	H	–	H	–	H	M	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Power-System Representation & Per-Unit System** Recap of the EGTD end-to-end system; what "modern" adds; The per-unit system — base quantities, changing base and its advantages; Single-line and impedance diagrams — symbols for generator, transformer, bus and line; The bus admittance matrix (Y-bus) — formation by inspection and by singular transformation; The bus impedance matrix (Z-bus) and network reduction; Modelling generators, transformers, lines and loads for system studies; Symmetrical components — introduction (full fault use in Module 3); Setting up a small test-network model	CO1
M2	**Load Flow / Power Flow Analysis** The power-flow problem — bus types (slack, PV, PQ); Power-flow equations and their nonlinearity; Gauss-Seidel method — acceleration and convergence; Newton-Raphson method and the Jacobian; Fast-decoupled load flow (B′ and B″ matrices); Interpreting results — voltage profile, line flows, losses and reactive margins; Contingency analysis (N-1); ill-conditioned systems and flat start; Load-flow software (PowerWorld, MATPOWER) on the test network	CO2

	Title & Topics	CO
M3	**Fault Analysis & System Stability** Symmetrical three-phase fault — pre-fault conditions, Thevenin equivalent and the Z-bus method; Symmetrical components — positive, negative and zero sequence; the operator a; Sequence networks for generators, transformers and lines; Unsymmetrical faults — single-line-to-ground, line-to-line and double-line-to-ground; Fault levels, short-circuit MVA and circuit-breaker rating (ANSI/IEEE C37); The swing equation and the inertia constant H; Steady-state and transient stability — the equal-area criterion and critical clearing time; Methods to improve stability — fast fault clearing, excitation control and governor action	CO3
M4	**Smart-Grid Sensing & Communication** The smart-grid vision — a two-way, digital, observable grid; Phasor Measurement Units (PMUs) and synchrophasors; Wide-Area Monitoring Systems (WAMS); SCADA and Energy Management Systems (EMS); Grid communication standards — IEC 61850 and DNP3; Cyber-physical security of the grid (overview); State estimation and bad-data detection; Designing a monitoring scheme for the test feeder	CO4
M5	**Distributed Energy Resources, Microgrids & Storage** Distributed generation — rooftop solar and small wind and their grid impact; Energy storage in the grid — batteries and the duck curve; Microgrids — grid-connected and islanded operation; Distribution challenges with high DER penetration — voltage rise and reverse flow; Hosting capacity and interconnection standards (IEEE 1547); Electric vehicles as grid loads and resources (V2G — depth in EVBT); India's 500-GW-by-2030 renewable context; Analysing DER impact on the test distribution system	CO5
M6	**Grid Operation, Analytics & Demand-Side Management** Economic dispatch and unit-commitment basics; Frequency control — governor action, load-frequency control and AGC; Voltage and reactive-power control; FACTS at concept level; Smart metering and Advanced Metering Infrastructure (AMI); Demand response and demand-side management; time-of-use tariffs and prosumers; Load and renewable forecasting (AI-assisted; ML tools used as given); Outage management, self-healing grids and grid digital twins; A smart-grid analytics case study for the demo	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Power System Analysis and Design**	J. Duncan Glover, Thomas Overbye & Mulukutla Sarma	Cengage	2017 (6th Ed.)
2	**Modern Power System Analysis**	I. J. Nagrath & D. P. Kothari	McGraw-Hill	2011 (4th Ed.)
3	**Power System Analysis**	Hadi Saadat	McGraw-Hill	2010 (3rd Ed.)
4	**Power System Analysis**	John J. Grainger & William D. Stevenson	McGraw-Hill	1994
5	**Power System Stability and Control**	Prabha Kundur	McGraw-Hill	1994
6	**Smart Grid: Technology and Applications**	Janaka Ekanayake, Kithsiri Liyanage, Jianzhong Wu, Akihiko Yokoyama & Nick Jenkins	Wiley	2012

Power Electronics & Wide-Bandgap Converters PEWB REG · 26EE2205 PBL · 26EE2205E

L-T-P-S	2-0-0-4 / 3-0-0-6	Credits	3	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S04 · T05	Mode	Regular & PBL	Category	PCC
Pre-requisite	ECES				

Course Outcomes (COs)

	Statement	BTL
CO1	Apply power-semiconductor-device fundamentals — power diode, BJT, MOSFET, IGBT, thyristor (SCR), GTO and IGCT — their V-I characteristics, switching dynamics, gate-drive requirements and canonical applications to select an appropriate device for a given converter.	BTL3 · Apply
CO2	Apply AC-DC converter analysis — single-phase and three-phase half-wave / full-wave / bridge rectifiers, controlled rectifiers, 12-pulse converters — to design rectifier-based DC supplies including power-factor and harmonic-mitigation requirements per IEEE 519 / IS 16636.	BTL3 · Apply
CO3	Apply DC-DC converter design — buck, boost, buck-boost, Cuk, SEPIC, and isolated topologies (flyback, forward, push-pull, half-bridge, full-bridge) — in CCM and DCM regimes for canonical SMPS, solar-MPPT, EV-DC-DC and telecom-power applications.	BTL3 · Apply
CO4	Apply DC-AC inverter design — single-phase and three-phase voltage-source inverters, square-wave and PWM modulation strategies (sine-PWM, space-vector PWM), multilevel inverters at intuition — to canonical EV-traction-inverter and grid-tied solar-inverter applications.	BTL3 · Apply
CO5	Analyse AC-AC converters (cycloconverters, matrix converters), snubber-circuit design, gate-driver requirements, EMI / EMC considerations per CISPR 22 / FCC Part 15, and PCB-layout discipline for high-frequency power-electronics designs.	BTL4 · Analyze
CO6	Analyse wide-bandgap devices (SiC, GaN), modern simulation tools (PSIM, LTspice, MATLAB Simulink), the AI-augmented power-electronics-design workflow and the Indian power-electronics ecosystem in the context of EV traction, solar-string inverter, and emerging high-frequency power-supply applications.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PO7	PSO1
CO1	H	H	H	M	–	H
CO2	H	H	H	M	–	H
CO3	H	H	H	H	–	H
CO4	H	H	H	H	–	H
CO5	H	H	H	H	–	H
CO6	H	H	H	H	M	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Power Semiconductor Devices** Why power semiconductors — efficient energy conversion via switching; the canonical "switch on and off rapidly with low loss" principle; comparison with linear regulators.; Power diode — V-I characteristics, reverse recovery time t_rr, the canonical PIN-structure; soft-recovery and Schottky-diode variants.; Power BJT — saturated-switch operation, the canonical safe-operating-area (SOA), second breakdown; mostly legacy in 2026 (replaced by MOSFET / IGBT).; Power MOSFET — the canonical switching-power-supply workhorse below 200 V; R_DS(on), gate charge Q_g, body diode, Miller plateau; canonical Infineon CoolMOS, ON-Semi, ST Micro families.; IGBT (Insulated Gate Bipolar Transistor) — the canonical 600 V – 6500 V industrial / EV-traction switch; gate-drive simplicity meets bipolar conduction loss; canonical Mitsubishi, Infineon, ABB modules.; Thyristor (SCR) — the canonical line-commutated high-power switch; latched-on operation requiring natural or forced commutation; canonical HVDC / industrial-rectifier applications.; GTO and IGCT at intuition — high-power gate-turn-off devices; the canonical Indian-Railways traction and HVDC-valve applications (LCC-HVDC stations).; First lab — characterise a small MOSFET; measure R_DS(on), threshold voltage V_GS(th), switching waveforms with an oscilloscope; observe the canonical Miller plateau in the gate waveform.	CO1

	Title & Topics	CO
M2	**AC-DC Converters (Rectifiers)** Diode-based half-wave rectifier — the canonical simplest topology; the canonical 0.318 V_p average output; output ripple analysis.; Diode-based full-wave rectifier — centre-tapped and bridge configurations; the canonical 0.636 V_p average output.; Filtering — capacitive (most common), LC and π-filter; the canonical ripple-voltage equation V_r ≈ I_dc / (f C); design trade-offs.; Three-phase rectifiers — half-wave (3-pulse), full-wave (6-pulse bridge); the canonical voltage and current waveforms and ripple analysis.; 12-pulse rectifiers — the canonical high-power industrial / HVDC / large-motor-drive application; harmonic-cancellation benefit.; Controlled rectifiers (thyristor-based) — phase-control; the canonical α-control of average DC output (V_dc = V_dc0 cosα); single-phase and three-phase variants.; Power-factor and harmonics in rectifiers — the canonical IEEE 519 / IS 16636 limits; modern power-factor-correction (PFC) techniques (boost-PFC, bridgeless-PFC).; Lab — build and characterise a 1-phase bridge rectifier with capacitive filter; measure ripple, regulation, power factor; verify against textbook predictions.	CO2
M3	**DC-DC Converters** Buck converter — canonical step-down topology; the canonical V_o = D · V_in relationship in CCM; inductor and capacitor selection.; Boost converter — canonical step-up topology; the canonical V_o = V_in / (1 − D) relationship; the canonical solar-MPPT / EV-DC-DC-stage application.; Buck-boost converter — bidirectional voltage transformation; the canonical inverting non-isolated topology; V_o = − V_in · D / (1 − D).; Cuk and SEPIC converters at intuition — the canonical advanced non-isolated topologies with continuous input current.; Continuous Conduction Mode (CCM) vs Discontinuous Conduction Mode (DCM) — the canonical regimes; design criteria; transition load current.; Isolated DC-DC converters — flyback (low power), forward, push-pull, half-bridge, full-bridge (high power); the canonical "transformer-isolated SMPS" workhorses.; Modern SMPS controller ICs — TI UCC2802x family, ADI LT8000-series, Infineon ICE family; the canonical "controller-plus-MOSFET" design pattern.; Lab — design and build a small 12 V → 5 V buck converter at 1 A; measure efficiency at full load; observe switching waveforms; verify against design calculations.	CO3
M4	**DC-AC Inverters** Single-phase voltage-source inverter — half-bridge and full-bridge; the canonical four switching states and dead-time requirements.; Square-wave vs PWM modulation — the canonical motivation for PWM (lower harmonics, cleaner output); modulation index.; Sine-PWM (SPWM) — the canonical compare-with-sine-reference technique; carrier frequency, modulation index, over-modulation.; Three-phase voltage-source inverter — the canonical 6-switch topology; eight switching states (six active + two zero); 120° and 180° conduction modes.; Space-Vector PWM (SVPWM) at intuition — the canonical higher-modulation-index technique; the canonical use in modern motor drives and grid-tied inverters.; Multilevel inverters at intuition — neutral-point-clamped (NPC), flying-capacitor, modular multilevel (MMC); the canonical HVDC-VSC, traction and large-PV-inverter applications.; EV traction inverter — the canonical three-phase IGBT or SiC-MOSFET inverter with FOC control; canonical Tesla / Ola Electric / Tata EV / Mahindra Last Mile Mobility designs.; Lab — simulate a three-phase SPWM inverter feeding an RL load in MATLAB Simulink; observe line voltages, line currents, and FFT showing dominant harmonics; vary modulation index.	CO4
M5	**AC-AC Converters & Protection** AC voltage controllers — phase-controlled and integral-cycle control; the canonical light-dimmer / fan-speed-control / industrial-heater applications.; Cycloconverters — the canonical high-power low-frequency-output applications (cement-kiln drives, large rolling-mill drives); mostly legacy due to VFD prevalence.; Matrix converters at intuition — direct AC-AC conversion without DC-link; canonical research and specialty applications (aircraft, aerospace, sustainability research).; Snubber circuits — RC, RCD; the canonical switching-spike suppression and EMI control.; Gate drivers — the canonical isolated gate-drive ICs (TI UCC family, Infineon EiceDriver, Silicon Labs Si827x); Miller-clamp, desat-detection, fault-handling features.; EMI / EMC in power electronics — conducted and radiated emissions; the canonical CISPR 22 / FCC Part 15 / IS 13252 standards; mitigation via input / output LC filtering and PCB layout.; PCB layout for power electronics — the canonical loop-area minimisation, ground-plane discipline, thermal-pad practice; canonical industry-standard layout patterns.; Lab — characterise the EMI emissions of the M3 buck converter using a near-field probe and spectrum analyser; identify dominant frequencies; apply LC filtering; measure the improvement.	CO5

	Title & Topics	CO
M6	**Wide-Bandgap Devices, Modern Tooling & Indian Ecosystem** Silicon carbide (SiC) — the canonical wide-bandgap material for high-voltage (650 – 1700 V), high-frequency (50 kHz – 200 kHz) applications; canonical EV-traction-inverter and solar-string-inverter applications.; Gallium nitride (GaN) — the canonical wide-bandgap material for low-voltage (40 – 650 V), very-high-frequency (1 – 10 MHz) applications; canonical mobile-phone-charger (GaNFast) and compact-power-supply applications.; SiC vs Si trade-offs — switching-loss reduction, conduction-loss trade-off, thermal-management implications; the canonical efficiency-vs-cost engineering decision.; GaN vs Si trade-offs — size and weight reduction at high frequency; gate-drive challenges (GaN HEMT lower V_GS_max); canonical applications.; Modern simulation tools — LTspice (free, ubiquitous), PSIM (specialised for power electronics + control), MATLAB Simulink with Simscape Electrical, PLECS (precision-converter modelling); the canonical "verify before building" workflow.; AI-augmented power-electronics design — LLM-assisted component selection (Claude / GPT-4 for sanity-checking power-stage choices), control-loop auto-tuning starting points; the discipline of always validating with simulator and bench measurement.; India's power-electronics ecosystem — Tata Power Solar inverters (Bengaluru), Delta Electronics India (Gurgaon, Chennai), ABB India, Schneider Electric India, Bharat Electronics Limited (BEL) defence-electronics; the upcoming Tata-PSMC 28 nm fab implications for power-semiconductor manufacturing.; Final-week lab — redesign the M3 buck converter using a GaN-based controller (e.g., TI LMG3422) at 1 MHz switching frequency; compare efficiency and size against the silicon-based version; observe the canonical wide-bandgap advantages.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Power Electronics — Converters, Applications, and Design**	Ned Mohan, Tore M. Undeland & William P. Robbins	Wiley	2003 (3rd Ed.)
2	**Power Electronics — Circuits, Devices, and Applications**	Muhammad H. Rashid	Pearson Education	2017 (4th Ed.)
3	**Fundamentals of Power Electronics**	Robert W. Erickson & Dragan Maksimović	Springer	2020 (3rd Ed.)
4	**Power Electronics**	P. S. Bimbhra	Khanna Publishers	2017 (6th Ed.)
5	**Power Electronics — A First Course**	Ned Mohan	Wiley	2012
6	**Wide Bandgap Semiconductor Power Devices — Materials, Physics, Design, and Applications**	B. Jayant Baliga	Woodhead Publishing	2018
7	**GaN Transistors for Efficient Power Conversion**	Alex Lidow, Michael de Rooij, Johan Strydom, David Reusch & John Glaser	Wiley	2019 (3rd Ed.)

Power System Protection and Control

PSPC REG · 26EE3309 PBL · 26EE3309E

L-T-P-S	2-0-2-0 / 3-0-0-6	Credits	3	Contact Hrs/Wk	4 / 9
Term (Reg · PBL)	S06 · T09	Mode	Regular & PBL	Category	PCC
Pre-requisite	EGTD				

Course Outcomes (COs)

	Statement	BTL
CO1	Understand power-system protection philosophy, relay types and protection-zone concepts.	BTL2 · Understand
CO2	Apply overcurrent protection — relay characteristics, coordination and grading — for radial feeders.	BTL3 · Apply
CO3	Apply differential protection for transformers, generators and busbars.	BTL3 · Apply
CO4	Analyze distance (impedance) protection for transmission lines — zones, characteristics, pilot schemes and power swings.	BTL4 · Analyze
CO5	Apply economic operation of power systems — economic dispatch and hydrothermal scheduling.	BTL3 · Apply
CO6	Analyze load-frequency control and voltage stability of a power system — the single-area LFC loop, automatic voltage regulation, and P-V/Q-V behaviour with reactive-power support.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PSO1	PSO2
CO1	H	H	M	M	M	L	L	–	L	L	–	H	M
CO2	H	H	H	H	H	L	–	–	M	M	L	H	M
CO3	H	H	H	H	H	L	–	–	M	M	L	H	M
CO4	H	H	H	H	H	L	–	–	L	L	–	H	M
CO5	H	H	H	H	H	L	–	–	L	L	–	H	M
CO6	H	H	H	H	H	L	–	L	M	M	M	H	M

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Protection Philosophy and Relay Fundamentals** Protection zones — overlapping zones; primary and backup protection; speed vs selectivity; Instrument transformers — CT accuracy class, burden and saturation; PT accuracy class; Relay operating principles — electromagnetic attraction; induction disc; static; digital; numerical; Relay characteristics — definite time; inverse time; extremely inverse; Circuit breakers — SF6 and vacuum; arc interruption; operating time; auto-reclosing; SCADA interface; Protection requirements — reliability (dependability + security), sensitivity, selectivity, speed, simplicity	CO1
M2	**Overcurrent Protection** Overcurrent relay (OCR) — plug-setting multiplier (PSM) and time-multiplier setting (TMS); IDMT relay — t = TMS × 0.14/(PSM^0.02 − 1); standard, very and extremely inverse curves; Grading of overcurrent relays — time-current grading; radial-feeder coordination; grading margin (0.3–0.4 s); Directional overcurrent relay — operating vs polarizing quantity; forward/reverse; Earth-fault relay — residual current I_0 = (I_a + I_b + I_c)/3; high-impedance earthing; restricted earth fault; Ring-main and interconnected systems — directional relays; pilot-wire protection; check-zone coordination	CO2
M3	**Differential Protection** Differential principle — I_d = I_1 − I_2; operate if I_d > k·I_r; Transformer differential protection — CT-ratio matching; zero-sequence elimination; percentage-bias characteristic; Inrush blocking — second-harmonic restraint; dual-slope bias characteristic; Transformer protection zones — differential for internal faults; overcurrent for through faults; Buchholz relay; Generator differential protection — stator-winding short-circuit; split-phase protection; Busbar differential protection — high-impedance (Solkor); low-impedance (Merz-Price); check relay	CO3

	Title & Topics	CO
M4	**Distance Protection** Distance-relay principle — Z_app = V/I; apparent impedance on the R-X diagram; Zone settings — Zone 1 (80–85%, instantaneous); Zone 2 (120%, time-delayed); Zone 3 (backup); Relay characteristics — impedance, reactance, mho, quadrilateral; load encroachment; Power-swing blocking — distinguishing a fault from a stable/unstable power swing; Phase and ground distance — separate pickup and settings; the mutual-impedance effect; Pilot protection schemes — POTT, PUTT, blocking schemes and DUTT	CO4
M5	**Economic Operation and Control** Economic dispatch — minimise total generation cost subject to load balance; Lagrange multiplier λ; Incremental cost (λ) — λ = dC_i/dP_i; the equal-incremental-cost criterion; Generator cost curve — input-output, heat-rate and incremental-heat-rate curves; fuel cost; Loss formula (B-coefficients) — P_L = ΣΣ P_i B_ij P_j; penalty factor; optimal dispatch with losses; Unit commitment — priority-list method; dynamic programming; MIP formulation; Hydrothermal scheduling — water value; the coordination equation; weekly hydro-energy target	CO5
M6	**LFC and Voltage Stability** Load-frequency control (single area) — governor, turbine and generation-load model; primary and secondary loops; Automatic voltage regulator (AVR) — block diagram; Voltage stability — maintaining acceptable voltages under steady-state and disturbance conditions; P-V (nose) curve — maximum loadability; the voltage-collapse point; voltage-stability margin; Q-V curve — reactive demand vs voltage; the stability boundary; Q reserve; Reactive-power sources — capacitor banks, synchronous condenser, STATCOM, SVC and generator VAr capability	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Protective Relaying: Principles and Applications**	J. Lewis Blackburn & Thomas J. Domin	CRC Press	2014 (4th Ed.)
2	**Power System Protection**	A. G. Phadke & J. S. Thorp	Wiley-IEEE Press	2009 (2nd Ed.)
3	**Fundamentals of Power System Protection**	Y. G. Paithankar & S. R. Bhide	PHI	2011 (2nd Ed.)
4	**Digital Protection for Power Systems**	A. G. Phadke & J. S. Thorp	IET	2009
5	**Power System Analysis**	Hadi Saadat	McGraw-Hill	2010 (3rd Ed.)

Problem Solving Using Programming (Java)

PSPJ 26SC1101

L-T-P-S	2-0-0-8	Credits	4	Contact Hrs/Wk	10
Semester	S01	Mode	Regular	Category	ESC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Translate a real-world problem into a precise algorithm and a working Java program using primitive types, expressions, and sequential console I/O.	BTL3 · Apply
CO2	Design control flow using selection and iteration to solve problems requiring decisions and repetition, choosing the appropriate construct and reasoning about termination.	BTL4 · Analyze
CO3	Modularise solutions using methods and recursion, and store/process homogeneous data with one- and two-dimensional arrays, applying standard processing patterns.	BTL4 · Analyze
CO4	Model a problem domain with classes and objects applying encapsulation and relationships, and make programs robust using Java's exception-handling mechanism.	BTL6 · Create
CO5	Process textual data with the String/StringBuilder APIs, apply standard library algorithms (searching, sorting, comparison), and persist data using file input/output.	BTL4 · Analyze
CO6	Manage collections of data and express data processing declaratively using the Collections framework, lambdas, functional interfaces, and Stream pipelines.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO8	PO9	PO11
CO1	M	**H**	M	L	**H**	–	–	M
CO2	M	**H**	M	L	**H**	–	–	M
CO3	M	**H**	**H**	M	**H**	–	–	M
CO4	M	**H**	**H**	M	**H**	–	–	M
CO5	M	**H**	M	M	**H**	–	–	M
CO6	M	**H**	**H**	M	**H**	M	M	**H**

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - From Problem to Program: Computational Thinking & Java Foundations** What computation is: input → process → output; effective procedures and determinism.; Computational-thinking pillars: decomposition, pattern recognition, abstraction, algorithm design.; Problem specification: turning ambiguous statements into precise inputs, outputs, assumptions, constraints, and edge cases.; Algorithms vs programs: pseudocode, flowcharts, stepwise refinement; one problem, many correct algorithms.; Desk-checking and trace tables; informal pre-/post-conditions and the idea of an invariant.; The Java toolchain: source → compiler (javac) → bytecode → JVM; 'write once, run anywhere' (conceptual).; Structure of a minimal Java program: class, main, statements, blocks.; Primitive types (byte, short, int, long, float, double, char, boolean): sizes, ranges, selection.; Variables, identifiers, literals, constants (final); operators (arithmetic, relational, logical, assignment, ternary).; Expression evaluation: precedence, associativity, mixed-type arithmetic; type conversion and casting (widening/narrowing, integer division, overflow, floating-point imprecision).; Console I/O: formatted printing and reading typed input with Scanner; readability discipline (naming, comments).	CO1

	Title & Topics	CO
M2	**Module 2 - Control Flow: Selection & Iteration** Selection: if, if-else, nested if-else, the if-else-if ladder; choosing between them.; The switch statement (including switch expressions): structure, fall-through, when switch beats a ladder.; Boolean conditions and short-circuit evaluation of && and ||.; Iteration: while, do-while, the for loop, and the enhanced for-each; equivalences and idiomatic uses.; Loop anatomy: initialisation, condition, update, body; definite vs indefinite iteration.; Counter-, sentinel-, and flag-controlled loops; nested loops and two-dimensional reasoning.; Loop control: break and continue and their readability trade-offs.; Reusable iteration patterns: counting, summing, max/min, search-for-condition, accumulation.; Reasoning about termination and the loop-invariant idea (informally) to avoid infinite loops and off-by-one errors.; Decision tables for mapping complex multi-condition logic before coding.	CO2
M3	**Module 3 - Decomposition with Methods, Recursion & Arrays** Methods as named, reusable abstractions: declaration, signature, invocation; a method as a contract.; Parameters and arguments: formal vs actual; pass-by-value semantics for primitives and references.; Return values and void; variable scope and lifetime; the principle of least scope.; Method overloading and compiler resolution; single-responsibility decomposition into a call hierarchy.; Recursion: base case and recursive case; the call stack as a mental model; recursion vs iteration trade-offs (qualitative).; Classic recursive formulations (factorial, sum of digits, power, simple sequences) as control technique only.; One-dimensional arrays: declaration, instantiation, initialisation, indexing, length; bounds and defensive iteration.; Array-processing patterns: traversal, accumulation, linear search, finding extremes, counting, reversing (as applied patterns - formal complexity is DSA-1's).; Two-dimensional arrays: rows/columns, row-major intuition, nested-loop processing (tables, grids, matrices as data).; Arrays as method arguments/returns; arrays as reference types and the aliasing pitfall; the Math class and standard methods as black-box abstractions.	CO3
M4	**Module 4 - Object-Oriented Modelling, Design & Exception Handling** From procedures to objects: bundling state and behaviour to model the real world.; Classes as blueprints, objects as instances; fields and methods; object creation with new; the reference model and null.; Constructors: default vs parameterised; constructor overloading; the this reference.; Encapsulation: private fields, accessors/mutators, information hiding and its rationale.; Instance vs static members; static utility methods and constants; the toString idea for self-description.; Modelling relationships: association and composition ('has-a'); designing a class from a specification (nouns→classes, verbs→methods).; Introduction to inheritance and interfaces at a working level: 'is-a', method overriding, and programming to an interface (kept practical - deep polymorphism/generics design is revisited in later OOP-heavy courses).; Errors vs exceptions; the exception hierarchy (checked vs unchecked) conceptually.; try-catch-finally; multi-catch; throwing exceptions with throw and declaring with throws.; Designing for robustness: validating inputs, failing safely, and writing a simple custom exception; the role of exceptions in clean error reporting.	CO4
M5	**Module 5 - Strings, Library Algorithms & File I/O** The String abstraction: immutability, the character model, and core operations (length, indexing, substring, concatenation, comparison, case conversion, searching).; equals vs == for Strings (value vs reference) and why it matters; StringBuilder for efficient mutation.; Building and parsing text: split/join, trimming, validation; simple tokenisation of records.; Applying library algorithms (not analysing them): Arrays.sort / Collections.sort, Arrays.binarySearch, and ordering with Comparable/Comparator at a usage level.; Standard searching and sorting as ready-made tools - choosing and calling them correctly (internals and Big-O are deferred to DSA-1).; Files and persistence: why programs need to remember data across runs.; Reading and writing text files with modern APIs (java.nio.file Files/Path) and buffered readers/writers; reading a whole file, line-by-line processing, appending.; Parsing structured text files (delimited records) into program data and writing reports back out.; Exceptions in file work: handling missing files and malformed lines gracefully; try-with-resources for automatic closing.; Putting it together: a file-backed data workflow (load → process → persist) as the backbone of a small persistent application.	CO5
M6	**Module 6 - Functional Programming with Collections & Streams (as practical APIs)** Why collections beyond arrays: dynamic size and richer operations; the Collections framework map (List, Set, Map, Queue) as practical APIs.; Using ArrayList, HashMap, HashSet, and an ordered map/set at a usage level: adding, removing, looking up, iterating (when to pick which - internals/complexity belong to DSA-1).; Generics at a usage level: typed collections (List<Parcel>) and why type safety helps; autoboxing.; Iterating collections: for-each, iterators, and the move toward declarative processing.; Functional interfaces and lambda expressions: the idea of behaviour-as-data; method references.; Common functional interfaces in practice (Predicate, Function, Consumer, Supplier, Comparator) used to parameterise behaviour.; Stream pipelines: creating streams from collections/files; intermediate operations (filter, map, sorted, distinct, limit) and terminal operations (forEach, collect, count, reduce).; Collectors in practice: toList, groupingBy, counting, joining - for building reports and summaries.; Optional as a null-safe result and its basic use; reading file lines as a stream.; Imperative vs functional style: rewriting a loop-based aggregation as a stream pipeline and reasoning about readability and intent (no performance/complexity analysis - that is DSA-1's).	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Java: How to Program (Early Objects)**	Paul J. Deitel & Harvey Deitel	Pearson	2017 (11th ed.)
2	**Core Java, Volume I - Fundamentals**	Cay S. Horstmann	Pearson / Oracle Press	2021 (12th ed.)
3	**Head First Java**	Kathy Sierra, Bert Bates & Trisha Gee	O'Reilly	2022 (3rd ed.)
4	**Modern Java in Action**	Raoul-Gabriel Urma, Mario Fusco & Alan Mycroft	Manning	2018 (2nd ed.)
5	**Introduction to Programming Using Java (open textbook)**	David J. Eck	Free / Hippocampus Press	2022 (v9, free online)
6	**Effective Java**	Joshua Bloch	Addison-Wesley	2018 (3rd ed.)

Problem Solving Using Programming (Java)

PSPP 26SC1101E

L-T-P-S	3-0-6-0	Credits	4	Contact Hrs/Wk	9
Trimester	T01	Mode	PBL (Project-Based)	Category	ESC
Pre-requisite	None				

Course Outcomes (COs)

	Statement	BTL
CO1	Translate a real-world problem into a precise algorithm and a working Java program using primitive types, expressions, and sequential console I/O.	BTL3 · Apply
CO2	Design control flow using selection and iteration to solve problems requiring decisions and repetition, choosing the appropriate construct and reasoning about termination.	BTL4 · Analyze
CO3	Modularise solutions using methods and recursion, and store/process homogeneous data with one- and two-dimensional arrays, applying standard processing patterns.	BTL4 · Analyze
CO4	Model a problem domain with classes and objects applying encapsulation and relationships, and make programs robust using Java's exception-handling mechanism.	BTL6 · Create
CO5	Process textual data with the String/StringBuilder APIs, apply standard library algorithms (searching, sorting, comparison), and persist data using file input/output.	BTL4 · Analyze
CO6	Manage collections of data and express data processing declaratively using the Collections framework, lambdas, functional interfaces, and Stream pipelines.	BTL6 · Create

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO5	PO8	PO9	PO11
CO1	M	**H**	M	L	**H**	–	–	M
CO2	M	**H**	M	L	**H**	–	–	M
CO3	M	**H**	**H**	M	**H**	–	–	M
CO4	M	**H**	**H**	M	**H**	–	–	M
CO5	M	**H**	M	M	**H**	–	–	M
CO6	M	**H**	**H**	M	**H**	M	M	**H**

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Module 1 - From Problem to Program: Computational Thinking & Java Foundations** What computation is: input → process → output; effective procedures and determinism.; Computational-thinking pillars: decomposition, pattern recognition, abstraction, algorithm design.; Problem specification: turning ambiguous statements into precise inputs, outputs, assumptions, constraints, and edge cases.; Algorithms vs programs: pseudocode, flowcharts, stepwise refinement; one problem, many correct algorithms.; Desk-checking and trace tables; informal pre-/post-conditions and the idea of an invariant.; The Java toolchain: source → compiler (javac) → bytecode → JVM; 'write once, run anywhere' (conceptual).; Structure of a minimal Java program: class, main, statements, blocks.; Primitive types (byte, short, int, long, float, double, char, boolean): sizes, ranges, selection.; Variables, identifiers, literals, constants (final); operators (arithmetic, relational, logical, assignment, ternary).; Expression evaluation: precedence, associativity, mixed-type arithmetic; type conversion and casting (widening/narrowing, integer division, overflow, floating-point imprecision).; Console I/O: formatted printing and reading typed input with Scanner; readability discipline (naming, comments).	CO1

	Title & Topics	CO
M2	**Module 2 - Control Flow: Selection & Iteration** Selection: if, if-else, nested if-else, the if-else-if ladder; choosing between them.; The switch statement (including switch expressions): structure, fall-through, when switch beats a ladder.; Boolean conditions and short-circuit evaluation of && and ||.; Iteration: while, do-while, the for loop, and the enhanced for-each; equivalences and idiomatic uses.; Loop anatomy: initialisation, condition, update, body; definite vs indefinite iteration.; Counter-, sentinel-, and flag-controlled loops; nested loops and two-dimensional reasoning.; Loop control: break and continue and their readability trade-offs.; Reusable iteration patterns: counting, summing, max/min, search-for-condition, accumulation.; Reasoning about termination and the loop-invariant idea (informally) to avoid infinite loops and off-by-one errors.; Decision tables for mapping complex multi-condition logic before coding.	CO2
M3	**Module 3 - Decomposition with Methods, Recursion & Arrays** Methods as named, reusable abstractions: declaration, signature, invocation; a method as a contract.; Parameters and arguments: formal vs actual; pass-by-value semantics for primitives and references.; Return values and void; variable scope and lifetime; the principle of least scope.; Method overloading and compiler resolution; single-responsibility decomposition into a call hierarchy.; Recursion: base case and recursive case; the call stack as a mental model; recursion vs iteration trade-offs (qualitative).; Classic recursive formulations (factorial, sum of digits, power, simple sequences) as control technique only.; One-dimensional arrays: declaration, instantiation, initialisation, indexing, length; bounds and defensive iteration.; Array-processing patterns: traversal, accumulation, linear search, finding extremes, counting, reversing (as applied patterns - formal complexity is DSA-1's).; Two-dimensional arrays: rows/columns, row-major intuition, nested-loop processing (tables, grids, matrices as data).; Arrays as method arguments/returns; arrays as reference types and the aliasing pitfall; the Math class and standard methods as black-box abstractions.	CO3
M4	**Module 4 - Object-Oriented Modelling, Design & Exception Handling** From procedures to objects: bundling state and behaviour to model the real world.; Classes as blueprints, objects as instances; fields and methods; object creation with new; the reference model and null.; Constructors: default vs parameterised; constructor overloading; the this reference.; Encapsulation: private fields, accessors/mutators, information hiding and its rationale.; Instance vs static members; static utility methods and constants; the toString idea for self-description.; Modelling relationships: association and composition ('has-a'); designing a class from a specification (nouns→classes, verbs→methods).; Introduction to inheritance and interfaces at a working level: 'is-a', method overriding, and programming to an interface (kept practical - deep polymorphism/generics design is revisited in later OOP-heavy courses).; Errors vs exceptions; the exception hierarchy (checked vs unchecked) conceptually.; try-catch-finally; multi-catch; throwing exceptions with throw and declaring with throws.; Designing for robustness: validating inputs, failing safely, and writing a simple custom exception; the role of exceptions in clean error reporting.	CO4
M5	**Module 5 - Strings, Library Algorithms & File I/O** The String abstraction: immutability, the character model, and core operations (length, indexing, substring, concatenation, comparison, case conversion, searching).; equals vs == for Strings (value vs reference) and why it matters; StringBuilder for efficient mutation.; Building and parsing text: split/join, trimming, validation; simple tokenisation of records.; Applying library algorithms (not analysing them): Arrays.sort / Collections.sort, Arrays.binarySearch, and ordering with Comparable/Comparator at a usage level.; Standard searching and sorting as ready-made tools - choosing and calling them correctly (internals and Big-O are deferred to DSA-1).; Files and persistence: why programs need to remember data across runs.; Reading and writing text files with modern APIs (java.nio.file Files/Path) and buffered readers/writers; reading a whole file, line-by-line processing, appending.; Parsing structured text files (delimited records) into program data and writing reports back out.; Exceptions in file work: handling missing files and malformed lines gracefully; try-with-resources for automatic closing.; Putting it together: a file-backed data workflow (load → process → persist) as the backbone of a small persistent application.	CO5
M6	**Module 6 - Functional Programming with Collections & Streams (as practical APIs)** Why collections beyond arrays: dynamic size and richer operations; the Collections framework map (List, Set, Map, Queue) as practical APIs.; Using ArrayList, HashMap, HashSet, and an ordered map/set at a usage level: adding, removing, looking up, iterating (when to pick which - internals/complexity belong to DSA-1).; Generics at a usage level: typed collections (List<Parcel>) and why type safety helps; autoboxing.; Iterating collections: for-each, iterators, and the move toward declarative processing.; Functional interfaces and lambda expressions: the idea of behaviour-as-data; method references.; Common functional interfaces in practice (Predicate, Function, Consumer, Supplier, Comparator) used to parameterise behaviour.; Stream pipelines: creating streams from collections/files; intermediate operations (filter, map, sorted, distinct, limit) and terminal operations (forEach, collect, count, reduce).; Collectors in practice: toList, groupingBy, counting, joining - for building reports and summaries.; Optional as a null-safe result and its basic use; reading file lines as a stream.; Imperative vs functional style: rewriting a loop-based aggregation as a stream pipeline and reasoning about readability and intent (no performance/complexity analysis - that is DSA-1's).	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Java: How to Program (Early Objects)**	Paul J. Deitel & Harvey Deitel	Pearson	2017 (11th ed.)
2	**Core Java, Volume I - Fundamentals**	Cay S. Horstmann	Pearson / Oracle Press	2021 (12th ed.)
3	**Head First Java**	Kathy Sierra, Bert Bates & Trisha Gee	O'Reilly	2022 (3rd ed.)
4	**Modern Java in Action**	Raoul-Gabriel Urma, Mario Fusco & Alan Mycroft	Manning	2018 (2nd ed.)
5	**Introduction to Programming Using Java (open textbook)**	David J. Eck	Free / Hippocampus Press	2022 (v9, free online)
6	**Effective Java**	Joshua Bloch	Addison-Wesley	2018 (3rd ed.)

Renewable Energy Systems & Grid Integration REGI REG · 26EE4111 PBL · 26EE4111E

L-T-P-S	2-0-4-0 / 3-0-6-0	Credits	4	Contact Hrs/Wk	6 / 9
Term (Reg · PBL)	S05 · T10	Mode	Regular & PBL	Category	PCC
Pre-requisite	EGTD				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze the renewable-energy landscape and resource assessment (solar irradiance, wind speed) to evaluate the potential of a site.	BTL4 · Analyze
CO2	Analyze solar photovoltaic systems — the cell/module/array, the I–V characteristic, MPPT and system design — for grid-connected and standalone applications.	BTL4 · Analyze
CO3	Analyze wind energy conversion systems — turbine aerodynamics, generators and control — and their power-curve behaviour.	BTL4 · Analyze
CO4	Apply power-electronic interfaces for renewables (PV inverters, MPPT converters, grid-tie inverters) using PEWC converter knowledge.	BTL3 · Apply
CO5	Analyze energy storage and emerging clean-energy carriers — batteries, pumped hydro, and green hydrogen — for balancing variable renewables.	BTL4 · Analyze
CO6	Analyze grid-integration of renewables — interconnection standards, grid codes, grid-forming inverters and the stability challenges of inverter-based resources.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO6	PO7	PSO1
CO1	H	H	–	–	H	H
CO2	H	H	–	–	–	H
CO3	H	H	–	–	–	H
CO4	H	–	H	–	–	H
CO5	H	H	–	–	H	H
CO6	H	–	–	H	H	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Renewable Landscape and Resource Assessment** The energy transition and India's 500-GW-by-2030 target.; Renewable sources overview: solar, wind, hydro, biomass, geothermal.; Solar resource: irradiance, sun path, the solar constant, measurement.; Wind resource: wind speed distribution (Weibull), wind atlas, measurement.; Capacity factor and the variability/intermittency challenge.; Levelised Cost of Energy (LCOE).; Site selection and feasibility.; Assessing a site for the project.	CO1
M2	**Solar Photovoltaic Systems** The PV cell: physics overview, the I–V and P–V characteristic.; Effect of irradiance and temperature.; Modules, strings and arrays; mismatch and shading.; Maximum Power Point Tracking (MPPT) algorithms.; PV system types: grid-tied, standalone, hybrid.; System sizing and component selection.; Solar-plant layout and balance-of-system.; Designing a grid-connected PV system for the site.	CO2
M3	**Wind Energy Conversion Systems** Wind turbine types: HAWT vs VAWT.; Aerodynamics: the power in the wind, the Betz limit, the power coefficient.; The turbine power curve: cut-in, rated, cut-out.; Generators for wind: DFIG and full-converter (PMSG).; Pitch and yaw control; MPPT for wind.; Onshore vs offshore.; Wind-farm layout and wake effects.; Analysing a wind system for the site.	CO3
M4	**Power-Electronic Interfaces for Renewables** Why renewables need power electronics (variable DC/AC → grid-quality AC).; PV inverters: string, central, microinverters.; DC-DC converters and MPPT implementation (topologies from PEWC).; Grid-tie inverters: synchronisation, current control.; Power quality: harmonics, anti-islanding.; Inverter standards and certification.; Reactive-power support from inverters.; Specifying the inverter/converter chain for the project.	CO4

	Title & Topics	CO
M5	**Energy Storage and Green Hydrogen** Why storage: balancing variable renewables, the duck curve.; Battery storage for the grid (chemistry depth shared with EVBT).; Pumped-hydro and other mechanical storage.; Green hydrogen: electrolysis, India's Green Hydrogen Mission.; Fuel cells (concept).; Sizing storage for a renewable plant.; Hybrid renewable + storage systems.; Adding storage to the project plant.	CO5
M6	**Grid Integration of Renewables** Interconnection standards (IEEE 1547) and Indian grid codes.; Impact of high renewable penetration on the grid (links to MPSG).; Grid-following vs grid-forming inverters.; Stability of inverter-based resources (low inertia).; Forecasting renewable generation.; Curtailment and the need for flexibility.; Microgrids and rural electrification.; A grid-integration study for the project plant for the demo.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Renewable Energy: Power for a Sustainable Future**	Godfrey Boyle (ed.)	Oxford University Press	2012 (3rd Ed.)
2	**Solar Photovoltaics: Fundamentals, Technologies and Applications**	Chetan Singh Solanki	PHI	2015 (3rd Ed.)
3	**Wind Energy Explained: Theory, Design and Application**	J. F. Manwell, J. G. McGowan & A. L. Rogers	Wiley	2009 (2nd Ed.)
4	**Renewable and Efficient Electric Power Systems**	Gilbert M. Masters	Wiley	2013 (2nd Ed.)
5	**Grid Integration of Solar Photovoltaic Systems**	Mahesh Kumar & Anurag K. Srivastava	CRC Press	2019

Robotic Motion Planning, Navigation & Control Systems

RBMT 26EC3316E

L-T-P-S	3-0-6-0	Credits	4	Contact Hrs/Wk	9
Trimester	T09	Mode	PBL (Project-Based)	Category	PCC
Pre-requisite	RBFN				

Course Outcomes (COs)

	Statement	BTL
CO1	Evaluate motion-planning and navigation architectures — purely reactive, deliberative-then-execute, hybrid hierarchical, anytime planners — against environment characteristics (static vs dynamic, structured vs unstructured) and select an appropriate stack.	BTL5 · Evaluate
CO2	Apply sampling-based motion planning — Probabilistic Roadmaps (PRM), Rapidly-exploring Random Trees (RRT, RRT*), the asymptotic-optimality argument — to plan in high-dimensional configuration spaces.	BTL3 · Apply
CO3	Apply search-based motion planning — A*, weighted A*, hybrid A* for non-holonomic vehicles, D* Lite for replanning under map updates — to compute paths with optimality guarantees.	BTL3 · Apply
CO4	Apply local planning and trajectory generation — Dynamic Window Approach (DWA), Timed Elastic Band (TEB), polynomial-and-spline trajectories, the planner-controller boundary — to follow global plans while reacting to local conditions.	BTL3 · Apply
CO5	Analyze optimal-control techniques — Linear Quadratic Regulator (LQR), Model Predictive Control (MPC) at depth, constraints handling, the differentiable-control connection to learning — to design controllers that minimise a cost while respecting constraints.	BTL4 · Analyze
CO6	Apply navigation in dynamic and uncertain environments — social-navigation patterns, behaviour-tree integration, recovery behaviours and fallback hierarchies, safety supervisors, the freezing-robot problem — to operate robots in real workplaces.	BTL3 · Apply

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO4	PO9	PSO1
CO1	H	H	H	H	–	H
CO2	H	H	H	–	–	H
CO3	H	H	H	–	–	H
CO4	H	–	H	–	–	H
CO5	H	H	–	–	–	H
CO6	H	–	H	H	M	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**Autonomous Navigation as a System** A working NavBot on day one — students place KartBot in a multi-room arena, give it a goal pose in another room, and watch it navigate autonomously through doorways and around obstacles before any internals.; The canonical autonomous-navigation stack: high-level goal → behaviour tree → global planner → local planner → controller → actuator (closing the loop with sensors via RPCS).; The Nav2 framework: the modern ROS 2 navigation stack; the planner-server, controller-server, behaviour-tree-navigator architecture.; The reactive-vs-deliberative spectrum: pure reactive (Braitenberg vehicles), pure deliberative (plan-then-execute), modern hybrid; what each is good for.; Anytime planners: produce a plan quickly, refine with more time; RRT* and ARA* as canonical examples.; Choosing a planning architecture by environment: static + structured (warehouse) → grid-based A*; static + unstructured (outdoor) → sampling-based RRT*; dynamic indoor → hybrid with DWA local planner.; A reading of NavBot's codebase: the Nav2 configuration, the custom RRT* planner plugin, the custom DWA-equivalent controller, the behaviour tree.; What this course owns vs upstream/downstream: planning + control here; perception in RPCS (already done); social autonomy in ARHR.	CO1

	Title & Topics	CO
M2	**Configuration Space and Sampling-Based Planning** Configuration space (C-space): the abstract space of all robot configurations; the obstacle inflation that maps physical obstacles to C-space.; Why C-space matters: what looks like a 2D robot in a 2D world is really a 3D problem (x, y, theta) for a non-circular robot; manipulators have 6+ dimensions.; Probabilistic Roadmap (PRM): build a graph of valid configurations and connect them; query by connecting start/goal to the graph; multi-query advantage.; Rapidly-exploring Random Tree (RRT): grow a tree from start toward goal by sampling and extending; the single-query workhorse.; RRT*: the asymptotically-optimal variant; how rewiring works; the convergence to optimal as samples → infinity.; Bidirectional RRT and RRT-Connect: growing two trees toward each other; the practical speedup.; Sampling strategies: uniform sampling, goal biasing, narrow-passage sampling.; Kinodynamic planning at intuition level: planning under dynamics constraints (e.g. car can't teleport); kinodynamic RRT.	CO2
M3	**Search-Based Planning** Grid-based representation: the discrete C-space; the resolution-vs-memory trade-off.; A* revisited from DSA-2 at the robot scale: state space, heuristic design (Euclidean, Manhattan, octile for 8-connected grids).; Weighted A* and ARA*: trading optimality for speed; the anytime nature.; Hybrid A*: search in continuous-state grids for non-holonomic vehicles (cars, differential drive at speed); the trick that makes A* work for vehicles that can't turn in place.; D* Lite: incremental replanning when the map updates; reusing previous computation.; Lifelong Planning A* (LPA*): the algorithmic foundation under D* Lite.; Theta* and any-angle path planning: paths that aren't restricted to grid-aligned moves; smoother trajectories.; When search-based beats sampling-based: low-dimensional, structured environments where optimality matters.	CO3
M4	**Local Planning and Trajectory Generation** The local-vs-global planner split: global plans the path; local turns the path into commands while reacting to immediate obstacles.; Dynamic Window Approach (DWA): sample feasible velocity commands within the dynamic window; score by goal-progress + obstacle-clearance + speed; pick the best.; Timed Elastic Band (TEB): the optimisation-based local planner; treats the trajectory as an elastic band that deforms around obstacles.; Polynomial trajectories: quintic and septic polynomials for smooth start-to-goal motion; jerk-bounded variants for comfort.; Spline trajectories: cubic splines, B-splines; the C2-continuity property.; Differential flatness at intuition level: when a system is differentially flat, trajectory generation in a flat output space is much simpler.; Trajectory smoothing: post-processing a piecewise-linear path into a feasible smooth trajectory.; The planner-controller boundary: what the planner outputs (sequence of poses, velocity profile) and what the controller consumes.	CO4
M5	**Optimal Control** Optimal control as the unifying lens: minimise a cost over a horizon, subject to dynamics and constraints.; Linear Quadratic Regulator (LQR): the canonical analytical solution for linear systems with quadratic cost; the Riccati equation.; LQR for trajectory tracking: linearise the dynamics around the reference, use LQR locally; the gain-scheduling pattern.; Model Predictive Control (MPC) at depth: receding horizon, the optimisation problem solved at every step, the warm-start trick.; Constraints in MPC: input limits (max steering angle), state limits (stay in lane), obstacle constraints (stay away from points); how each is encoded.; Linear MPC vs nonlinear MPC (NMPC): when to linearise, when to bite the nonlinear bullet.; MPC solver landscape: OSQP, acados, CasADi; the open-source MPC toolchain.; Differentiable control: the gradient-friendly variant; the bridge to learning controllers from data; the connection to reinforcement learning at intuition level.	CO5
M6	**Navigation in Dynamic and Uncertain Environments** The freezing-robot problem: when conservative planning prevents the robot from moving in a crowd; the engineering of "graceful navigation around people".; Social navigation patterns: the social-force model, the human-aware-cost-map approach, the proxemics-aware planning.; Behaviour trees revisited at depth: composite nodes (sequence, fallback, parallel), decorators, the BehaviorTree.CPP library used in Nav2.; Recovery behaviours: clear-cost-map, spin, back-up; the layered fallback hierarchy that prevents the robot from getting stuck.; Safety supervisors: independent watchdog logic that overrides the navigator if a critical condition is met (low battery, collision detected, geo-fence violation).; Localisation failure recovery: detecting that AMCL has gotten lost; the global-relocalisation behaviour.; Multi-goal missions: queueing goals, mission cancellation, the API the operator uses.; A glance at policy learning for navigation: end-to-end learned policies, hybrid approaches (learned local planner with classical global planner); the deeper coverage in ARHR.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Planning Algorithms**	Steven M. LaValle	Cambridge University Press / Free PDF	2006

#	Title	Authors	Publisher	Year
2	**Principles of Robot Motion — Theory, Algorithms, and Implementations**	Howie Choset, Kevin M. Lynch, Seth Hutchinson, George A. Kantor, Wolfram Burgard, Lydia E. Kavraki & Sebastian Thrun	MIT Press	2005
3	**Modern Robotics — Mechanics, Planning, and Control**	Kevin M. Lynch & Frank C. Park	Cambridge University Press	2017
4	**Predictive Control for Linear and Hybrid Systems**	Francesco Borrelli, Alberto Bemporad & Manfred Morari	Cambridge University Press	2017
5	**Optimization-Based Control**	Richard M. Murray (lecture notes)	Caltech / Free PDF	2023
6	**Probabilistic Robotics**	Sebastian Thrun, Wolfram Burgard & Dieter Fox	MIT Press	2005
7	**Robotics, Vision and Control — Fundamental Algorithms in Python**	Peter Corke	Springer	2023 (3rd Ed.)

Robotic Perception, Computer Vision & Sensor Fusion

RBPR 26EC3214E

L-T-P-S	3-0-6-0	Credits	4	Contact Hrs/Wk	9
Trimester	T08	Mode	PBL (Project-Based)	Category	PCC
Pre-requisite	RBFN				

Course Outcomes (COs)

	Statement	BTL
CO1	Analyze a robotic perception stack as a layered system — scene understanding, segmentation, detection-and-tracking, sensor fusion, calibration, raw sensor data — and trace one perception event from a sensor frame to a planner-consumable scene description.	BTL4 · Analyze
CO2	Apply camera models — pinhole, distortion correction, intrinsic and extrinsic calibration, stereo matching, epipolar geometry, depth from disparity — to extract 3D structure from images.	BTL3 · Apply
CO3	Apply visual object detection and tracking on a robot — adapting deep detectors from DLAC for on-robot inference, multi-object tracking (Hungarian assignment, ByteTrack), tracking-by-detection vs tracking-by-regression — to maintain identities across frames.	BTL3 · Apply
CO4	Analyze 3D perception with lidar — point-cloud representations (raw, voxelised), point-cloud filtering and segmentation, 3D detection at intuition level, ground-plane extraction, the lidar / camera / depth-from-stereo trade-off — to reason about the 3D scene structure.	BTL4 · Analyze
CO5	Apply multi-sensor fusion — the Kalman filter at depth, extended (EKF) and unscented (UKF) variants, factor graphs at intuition level, the multi-sensor calibration discipline, time synchronisation — to produce a single coherent state estimate.	BTL3 · Apply
CO6	Analyze SLAM (Simultaneous Localisation and Mapping) — feature-based visual SLAM (ORB-SLAM3), lidar-inertial SLAM (LIO-SAM), loop closure, the front-end / back-end architecture — to enable a robot to build a map of an unknown environment while localising in it.	BTL4 · Analyze

CO–PO / PSO Articulation Matrix

	PO1	PO2	PO3	PO5	PSO1
CO1	H	H	H	–	H
CO2	H	H	–	–	H
CO3	H	–	H	–	H
CO4	H	H	–	–	H
CO5	H	–	H	M	H
CO6	H	H	H	–	H

H = High (3) · M = Moderate (2) · L = Low (1) · blank = no significant correlation

Module-wise Syllabus

	Title & Topics	CO
M1	**The Perception System as a Whole** A working SeeBot on day one — students teleoperate KartBot through a dynamic environment and watch the perception stack identify, track, and map.; The canonical robotic-perception stack: scene description (semantic + geometric) ← segmentation ← detection-and-tracking ← sensor fusion ← raw sensor frames.; The contrast between FRMS's perception (a-priori map + AMCL) and RPCS's perception (online mapping + understanding); what becomes possible.; The autonomous-vehicle perception stack as the canonical reference: AV-grade is over-engineered, but understanding it gives the right vocabulary.; Sensor selection: camera (rich, computationally expensive, lighting-dependent), lidar (precise depth, expensive hardware, sparse), radar (robust to weather, low resolution), depth camera (close-range RGBD); the multi-sensor norm.; Latency budget for a robot: 100ms is fast, 1000ms is too slow; how the perception stack must be carved up.; A reading of SeeBot's codebase: detector node, tracker node, lidar processor, fusion node, SLAM node; how data flows across the ROS graph.; What this course owns vs upstream/downstream: perception here; vision DL in DLAC; motion planning in RMNC.	CO1

	Title & Topics	CO
M2	**Camera Models and Multi-View Geometry** The pinhole camera model: focal length, principal point, intrinsic matrix K; the projection from world to image.; Lens distortion: radial and tangential coefficients; the OpenCV calibration pipeline.; Camera calibration in practice: checkerboard, ChArUco, the calibration_camera ROS package; what good calibration looks like.; Extrinsic calibration: locating one camera relative to another, or relative to a lidar; the hand-eye calibration problem.; Stereo geometry: rectification, epipolar lines, the baseline-disparity-depth relationship.; Stereo matching: block matching, semi-global matching (SGBM), the modern deep-learning approaches at intuition level.; Depth-from-stereo in practice: the depth-camera (Intel RealSense, ZED) as the integrated commercial answer.; Triangulation from multiple views: the bedrock technique for visual SLAM.	CO2
M3	**Visual Detection and Tracking** Adapting DLAC's detector to the robot: model selection (YOLOv8-n, MobileNet-SSD), hardware target (Pi 4 CPU, Jetson Orin Nano GPU), latency budget.; Detection on streaming video: temporal coherence, double-detection avoidance, NMS across frames.; Multi-object tracking by detection: the assignment problem, Hungarian algorithm, the cost matrix design (IoU + appearance).; Modern trackers: SORT, DeepSORT, ByteTrack; the motion-model-plus-appearance pattern.; Tracking by regression: Siamese trackers, the single-target case; when to use which paradigm.; Re-identification: maintaining identity through occlusion; appearance-feature-extractor backbones.; 3D bounding boxes from 2D detections + depth: the practical recipe; the limits when depth is wrong.; Action recognition at intuition level: temporal aggregation; the bridge to behaviour prediction (which lives more fully in ARHR).	CO3
M4	**Lidar Processing and 3D Perception** Point clouds: the data structure, the standard ROS PointCloud2 message, intensity vs RGB.; Point Cloud Library (PCL): filtering (voxel grid downsample, statistical outlier removal), segmentation (RANSAC plane fitting), clustering (Euclidean cluster extraction).; Ground-plane extraction: the canonical first-step for autonomous driving and warehouse robots.; Voxelisation and occupancy grids in 3D: octrees (OctoMap), the trade-off between resolution and memory.; 3D object detection at intuition level: PointPillars, CenterPoint; the challenge of working in continuous 3D space.; Lidar odometry at intuition level: ICP (Iterative Closest Point), generalised ICP, NDT; the algorithm that runs every 100ms in autonomous vehicles.; Camera-lidar fusion at point-cloud level: projection of lidar into image, fusion of detections.; When lidar wins, when camera wins, when both are needed: the practical fusion-strategy menu.	CO4
M5	**Sensor Fusion** The Kalman filter at depth: state-space model, predict step, update step, the optimal-linear-estimator derivation.; The Extended Kalman Filter (EKF): linearising around the current estimate; the workhorse of practical robotics.; The Unscented Kalman Filter (UKF): sigma-point sampling; better accuracy on non-linearities at moderate cost.; Particle filter at intuition level (already seen in FRMS for AMCL): when non-Gaussian beats Gaussian.; Factor graphs and pose-graph optimisation at intuition level: the foundation of modern SLAM back-ends; GTSAM as the standard library.; Multi-sensor calibration: time synchronisation (PTP, ROS's message_filters approximate sync), spatial calibration (the extrinsics from M2 revisited).; Sensor failure handling: detecting that a sensor is misbehaving (covariance explosion, innovation gating) and downgrading gracefully.; A walk-through of SeeBot's fusion node: combining wheel odometry, IMU, and visual odometry into a single state estimate.	CO5
M6	**SLAM** The SLAM problem: simultaneously map an unknown environment AND localise within the partial map you're building.; The front-end / back-end architecture: front-end finds correspondences (features, scan-matching); back-end optimises the global pose graph.; Visual SLAM: ORB-SLAM3 as the modern reference; feature detection (ORB), descriptor matching, tracking, mapping, loop closure.; Lidar SLAM: LOAM lineage (LIO-SAM, FAST-LIO); the lidar-IMU coupling that makes it robust.; Loop closure: detecting when the robot returns to a previously-mapped place; the dramatic accuracy improvement; bag-of-visual-words and modern learned alternatives.; The kidnapped-robot problem and global relocalisation: when a robot is teleported and must re-localise from scratch.; SLAM in dynamic environments at intuition level: the moving-objects problem; the masking-of-dynamic-pixels mitigation.; A practical SLAM walkthrough: running ORB-SLAM3 on SeeBot's recorded bag and comparing the produced map against the ground-truth FRMS map.	CO6

Reference Books

#	Title	Authors	Publisher	Year
1	**Computer Vision — Algorithms and Applications**	Richard Szeliski	Springer	2022 (2nd Ed.)
2	**Multiple View Geometry in Computer Vision**	Richard Hartley & Andrew Zisserman	Cambridge University Press	2004 (2nd Ed.)
3	**Probabilistic Robotics**	Sebastian Thrun, Wolfram Burgard & Dieter Fox	MIT Press	2005

#	Title	Authors	Publisher	Year
4	**State Estimation for Robotics**	Timothy D. Barfoot	Cambridge University Press	2024 (2nd Ed.)
5	**Robotics, Vision and Control — Fundamental Algorithms in Python**	Peter Corke	Springer	2023 (3rd Ed.)
6	**An Invitation to 3-D Vision — From Images to Geometric Models**	Yi Ma, Stefano Soatto, Jana Košecká & S. Shankar Sastry	Springer	2003
7	**Introduction to Visual SLAM — From Theory to Practice**	Xiang Gao & Tao Zhang	Springer	2021

5 · Program Structure (Trimester-wise)

Year 1 · Trimesters 1–3

Trimester 1

Code	Acronym	Course Title	Category	L-T-P-S	CH	Cr	Pre-req
26PE5101E	PECM	Power Electronic Converters and Modulation	Core	3-1-3-0	7	3.5	—
26PE5102E	MPSA	Modern Power System Analysis and State Estimation	Core	3-1-3-0	7	3.5	—
26PE5103E	MAEM	Modelling and Analysis of Electrical Machines	Core	3-1-3-0	7	3.5	—
26PE5104E	DSPP	Digital Signal Processing for Power Engineers	Core	3-1-3-0	7	3.5	—
26UC5101E	CSTW	Communication Skills and Technical Writing	Audit	0-0-0-3	3	0	—
Trimester total				**31 CH**		**14**	

Trimester 2

Code	Acronym	Course Title	Category	L-T-P-S	CH	Cr	Pre-req
26PE5205E	RESI	Renewable Energy Systems and Integration	Core	3-1-3-0	7	3.5	MPSA
26PE5206E	ASGT	AI-Enabled Smart Grid Technologies	Core	3-1-3-0	7	3.5	MPSA
26PE5207E	EDRV	Electric Drives	Core	3-1-3-0	7	3.5	MAEM
PEC-1	PEC1	Professional Elective – I	Elective	3-1-3-0	7	3.5	—
26IE5201E	RMIP	Research Methodology and Intellectual Property Rights	Project	3-0-0-0	3	2	—
Trimester total				**31 CH**		**16**	

Trimester 3

Code	Acronym	Course Title	Category	L-T-P-S	CH	Cr	Pre-req
26PE5308E	PQHA	Power Quality and Harmonics	Core	3-1-3-0	7	3.5	PECM
26PE5309E	PSPA	Power System Protection and Substation Automation	Core	3-1-3-0	7	3.5	MPSA
26PE5310E	PSDY	Power System Dynamics and Stability	Core	3-1-3-0	7	3.5	MPSA
PEC-2	PEC2	Professional Elective – II	Elective	3-1-3-0	7	3.5	PEC1
26UC5302E	PTPR	Placement Training and Professional Readiness	Audit	0-0-0-3	3	0	—
Trimester total				**31 CH**		**14**	

Year 2 · Trimesters 4–6

Trimester 4

Code	Acronym	Course Title	Category	L-T-P-S	CH	Cr	Pre-req
26PE6112E	PSOF	Power System Operation and AI Forecasting	Core	3-1-3-0	7	3.5	MPSA
26PE6111E	HVDC	HVDC and FACTS	Core	3-1-3-0	7	3.5	MPSA
PEC-3	PEC3	Professional Elective – III	Elective	3-1-3-0	7	3.5	PEC1
PEC-4	PEC4	Professional Elective – IV	Elective	3-1-3-0	7	3.5	PEC1
Trimester total				**28 CH**		**14**	

Trimester 5

Code	Acronym	Course Title	Category	L-T-P-S	CH	Cr	Pre-req
26IE6202E	CAP1	Capstone Research / Industry Project – Phase I	Project	0-0-36-0	36	12	RMIP
Trimester total				**36 CH**		**12**	

Trimester 6

Code	Acronym	Course Title	Category	L-T-P-S	CH	Cr	Pre-req
26IE6303E	CAP2	Capstone Research / Industry Project – Phase II	Project	0-0-36-0	36	12	CAP1
Trimester total				**36 CH**		**12**	

Total programme credits: 82

Professional Electives — Course Basket

Code	Acronym	Elective Title	Stream	L-T-P-S	CH	Cr	Pre-req
26PE52A1E	EPBM	EV Powertrain and Battery Management Systems	Stream A : E-Mobility and Power Electronics	3-1-3-0	7	3.5	—
26PE53A2E	SECI	Smart EV Charging and IoT Energy Management	Stream A : E-Mobility and Power Electronics	3-1-3-0	7	3.5	EPBM
26PE61A3E	MDEV	Motor Drives for Electric Vehicles	Stream A : E-Mobility and Power Electronics	3-1-3-0	7	3.5	EPBM
26PE61A4E	PERE	Power Electronics for Renewable Energy	Stream A : E-Mobility and Power Electronics	3-1-3-0	7	3.5	EPBM
26PE52B1E	MDAM	Microgrid Design and AI Energy Management	Stream B : Smart Grid and Energy Systems	3-1-3-0	7	3.5	—
26PE53B2E	BESS	Battery Energy Storage Systems	Stream B : Smart Grid and Energy Systems	3-1-3-0	7	3.5	MDAM
26PE61B3E	PMET	Power Markets and Energy Trading	Stream B : Smart Grid and Energy Systems	3-1-3-0	7	3.5	MDAM
26PE61B4E	GIRE	Grid Integration of Renewables	Stream B : Smart Grid and Energy Systems	3-1-3-0	7	3.5	MDAM

Syllabus : **AI DRIVEN ECU DESIGN FOR AUTONOMOUS VEHICLES**

Old syllabus-Y22	Modified syllabus- Y23 L-T-P-S : 3-0-0-0 3 Cr
CO1 Apply embedded system functions for autonomous vehicle ECU functioning PO1,PO5,PSO1 3 CO2 Analyze machine learning functions for autonomous vehicle ECU network PO1,PO5,PSO1 4 CO3 Analyze deep learning functions for autonomous vehicle ECU network PO1,PO5,PSO1 4 CO4 Analyze ECU performance with integrated artificial intelligent control PO1,PO5,PSO1 4	CO1 Model complex control structures for Autonomous driving PO1,PSO1,PO5 3 CO2 Analyze advanced sensor integration for AI driven ECU PO2,PO5,PSO1 4 CO3 Analyze advanced communication architectures for AI driven ECU PO2,PO5,PSO1 4 CO4 Analyze ECU integration and validation for real-time complex scenarios of autonomous driving PO2,PO5,PSO1 4
AEV ECU Product features overview: Overview of Autosar Architecture, Sensor Data Acquisition using Autosar MCAL, Microcontroller, Memory, IO, Communication, CAN Communication Establishing using Autosar COM-CAN, CAN If, PDUR, COM Stack, Vehicle Diagnostics using Autosar Diagnostics, DCM, DEM, Memory Handling through Autosar MEM-MEMIF, NVM, FEE, FLS, EA, Derive the Output stage driver using Autosar Complex Device Driver AI/ML algorithm to identify and track vehicles, pedestrian, and traffic signs, Classify and categorize the images using DL algorithms, Generate best paths to avoid obstacle and reach destination, Support Complex maneuvers for Lane detection and parking assist, Predictive control and optimize control using AI/ML, Natural Language processing for HMI Interface Battery Degradation and Life cycle prediction using AI/ML, Estimation of SOC using ML, Predicting Energy Consumption using Vehicle parameters, Optimize charging strategies, smart Charging, Efficient management of V2G interactions during peak demand, Design consideration Design Guide, Defining Vehicle purpose, Select Components, Develop Planning and Control system, Integrate AI&ML, Develop HMI, Ensure Safety and Compliance, Design Consideration, Passenger Capacity, Range, Charging, Driving Conditions, safety features, cost, Autonomous	Complex Control Structures for autonomous driving stack: Introduction to constrained control using simple model predictive control formulation for position and velocity tracking, MPC using discrete prediction horizon and cost function minimization, comparison of proportional control and MPC linear regression and lookup-based model for control tuning, handling actuator limits and delay in control signals, integration of control output with sensor input in a perception to control pipeline using ROS2 nodes. Perception stack design: IMU-based orientation estimation using gyroscope integration, implementation of encoder and IMU fusion using complementary filter or basic Kalman filter, development of ROS2 sensor nodes for publishing encoder and IMU data, real-time visualization of raw and fused data using ROS2 tools, implementation of simple camera-based lane detection and publishing results as ROS2 topics, integration of perception output with control input in ROS2 pipeline. Advanced Communication Architectures: Introduction to high-speed automotive communication using Automotive Ethernet and need for high

Vehicle Design, Embedded Design - Data Acquisition through various sensing elements, Ultrasonic, LIDAR, Camera, Deploy AI&ML algorithm to perceive decision making, Control various actuator, Motor control, By Wire technology, Design Steer by Wire, Introduction to Flexray Protocol, Introduction to Ethernet Controller	bandwidth in AI-based ECUs, basic concepts of TCP/IP and UDP communication in embedded systems, understanding of deterministic communication and introduction to time-sensitive networking for real-time data transfer, implementation of ROS2 publish-subscribe communication, structuring of distributed ECU system with separate sensor, processing and control nodes, introduction to service-based communication and modular ECU software design. Autonomous ECU Integration and validation: Integration of sensor nodes, state estimation, control and communication modules into a complete ROS2-based ECU pipeline, configuration of multi-node ROS2 system with defined topics and data flow between perception, estimation and control nodes, implementation of end-to-end pipeline from sensor input to actuator command synchronization of sensor data and control loops with appropriate execution rates, testing of complete system for trajectory tracking using encoder and IMU feedback, implementation of scenario-based testing.