SCHOOL OF ENGINEERING AND TECHNOLOGY

Programme Handbook

(Programme Structure and Evaluation Scheme)

B.Tech (Computer Science & Engineering) with Specialization in AI & ML

Programme Code: 73

FOUR YEAR UNDERGRADUATE PROGRAMME

(with effect from 2024-25 session)

Approved in the 34th Meeting of Academic Council Held on 29 June 2024

Table of Contents

Contents

Preface

About University:

The K.R. Mangalam Group has made a name for itself in the field of education. Over a period of time, the various educational entities of the group have converged into a fully functional corporate academy. Resources at KRM have been continuously upgraded to optimize opportunities for the students. Our students are groomed in a truly inter-disciplinary environment wherein they develop integrative skills through interaction with students from engineering, management, journalism and media study streams.

The K.R. Mangalam story goes back to the chain of schools that offered an alternative option of world-class education, pitching itself against the established elite schools, which had enjoyed a position of monopoly till then. Having blazed a new trail in school education, the focus of the group was aimed at higher education. With the mushrooming of institutions of Higher Education in the National Capital Region, the university considered it very important that students take informed decisions and pursue career objectives in an institution, where the concept of education has evolved as a natural process.

K.R. Mangalam University was founded in the year 2013 by Mangalam Edu Gate, a company incorporated under Section 25 of the Companies Act, 1956.

Uniqueness of KRMU

i. Enduring legacy of providing education to high achievers who demonstrate leadership in diverse fields.

ii. Protective and nurturing environment for teaching, research, creativity, scholarship, social and economic justice.

Education Objectives

i. To impart undergraduate, post-graduate and Doctoral education in identified areas of higher education.

ii. To undertake research programs with industrial interface.

iii. To integrate its growth with the global needs and expectations of the major stake holders through teaching, research, exchange & collaborative programs with foreign, Indian Universities/Institutions and MNCs.

iv. To act as a nodal center for transfer of technology to the industry.

v. To provide job oriented professional education to the student community with particular focus on Haryana.

Categories of Courses

Major: The major would provide the opportunity for a student to pursue in-depth study of a particular subject or discipline.

Industry Driven Courses (IDC): The purpose of our industry-driven courses is to align academic learning with industry needs. Through engagement with industry experts, students receive hands-on training and real-world experience throughout the semester, ensuring they develop the practical skills needed to become industry-ready upon graduation.

Multidisciplinary (Open Elective): These courses are intended to broaden the intellectual experience and form part of liberal arts and science education. These introductory-level courses may be related to any of the broad disciplines given below:

- Natural and Physical Sciences
- Mathematics, Statistics, and Computer Applications
- Library, Information, and Media Sciences
- Commerce and Management
- Humanities and Social Sciences

A diverse array of Open Elective Courses, distributed across different semesters and aligned with the aforementioned categories, is offered to the students. These courses enable students to expand their perspectives and gain a holistic understanding of various disciplines. Students can choose courses based on their areas of interest.

Ability Enhancement Course (AEC): Students are required to achieve competency in a Modern Indian Language (MIL) and in the English language with special emphasis on language and communication skills. The courses aim at enabling the students to acquire and demonstrate the core linguistic skills, including critical reading and expository and academic writing skills, that help students articulate their arguments and present their thinking clearly and coherently and recognize the importance of language as a mediator of knowledge and identity.

Skills Enhancement Courses (SEC): These courses are aimed at imparting practical skills, hands-on training, soft skills, etc., to enhance the employability of students.

Value-Added Course (VAC): The Value-Added Courses (VAC) are aimed at inculcating Humanistic, Ethical, Constitutional and Universal human values of truth, righteous conduct, peace, love, non-violence, scientific and technological advancements, global citizenship values and life-skills falling under below given categories:

- Understanding India
- Environmental Science/Education
- Digital and Technological Solutions
- Health & Wellness, Yoga education, Sports, and Fitness

Discipline Specific Electives (DSE): The purpose of offering discipline-specific electives is to provide students with the flexibility to specialize in emerging and high-demand domains such as Full Stack Development, Cloud Computing, AI & ML, and Cyber Security. These electives are designed to equip students with advanced knowledge and skills in their chosen fields, ensuring they are well-prepared for specialized roles and industry demands in these cutting-edge areas.

Industry project/Research Project: Students choosing a 4-Year Bachelor's degree are required to take up Industry/research projects. The purpose of our full-time, 6-month industry project for final-year students is to provide them with practical exposure by working on real-world industry projects.

University Vision and Mission

3.1 Vision

K.R. Mangalam University aspires to become an internationally recognized institution of higher learning through excellence in inter-disciplinary education, research, and innovation, preparing socially responsible life-long learners contributing to nation building.

3.2 Mission

- Foster employability and entrepreneurship through futuristic curriculum and progressive pedagogy with cutting-edge technology
- Instill notion of lifelong learning through stimulating research, Outcomes-based education, and innovative thinking
- Integrate global needs and expectations through collaborative programs with premier universities, research centres, industries, and professional bodies.
- Enhance leadership qualities among the youth having understanding of ethical values and environmental realities

About the School

About School of Engineering and Technology:

Since its establishment in 2013, the School of Engineering and Technology at K.R. Mangalam University has rapidly developed into a hub of innovation, quality education, and skill development. Our focus is on delivering a transformative educational experience that equips students with advanced technical knowledge while fostering creativity and critical thinking. With state-of-the-art infrastructure, modern laboratories, and a distinguished faculty, we provide an environment that nurtures both academic and professional excellence.

Our school offers a comprehensive range of programs, including undergraduate (B.Tech, BCA, B.Sc), postgraduate (M.Tech, MCA), and doctoral studies across key engineering disciplines. We are proud to offer specialized B.Tech programs in high-demand fields such as Artificial Intelligence & Machine Learning, Data Science, Cyber Security, Full Stack Development, and UI/UX Development. These programs are designed to meet the evolving needs of the industry, ensuring that students are equipped with the skills and knowledge required to succeed in the modern workforce.

Our curriculum is grounded in best practices from leading global institutions and incorporates insights from the Open-Source Society University. It emphasizes interdisciplinary learning, problem-solving, and innovative teaching methodologies. This approach not only enhances students' technical competencies but also develops their ability to think critically and work collaboratively across diverse domains.

Industry integration is a key component of our educational model. We collaborate with renowned organizations such as IBM, Samatrix, Xebia, EC Council, and ImaginXP to provide students with practical, real-world experience through internships, projects, and workshops. These partnerships ensure that our students are well-prepared to meet industry demands. Additionally, we offer elective courses in areas such as AI, Cloud Computing, Cyber Security, and Full Stack Development, allowing students to tailor their learning experience to align with their career goals.

We are also committed to fostering innovation and entrepreneurship. Our **Entrepreneurship and Incubation Center** and initiatives like 'MindBenders,' 'Hack-KRMU,' and participation in the **Smart India Hackathon** inspire students to develop forward-thinking solutions and entrepreneurial ventures.

With cutting-edge computing facilities, advanced research opportunities, and a focus on practical application, the School of Engineering and Technology ensures that its graduates are well-prepared to excel in their careers. Our alumni have made significant contributions across various sectors, reflecting the high standards of education they receive.

School Vision and Mission

Vision

To excel in scientific and technical education through integrated teaching, research, and innovation.

Mission

- **Creating** a unique and innovative learning experience to enhance quality in the domain of Engineering & Technology.

- **Promoting** Curricular, co-curricular and extracurricular activities that support overall personality development and lifelong learning, emphasizing character building and ethical behavior.
- **Focusing** on employability through research, innovation and entrepreneurial mindset development.
- **Enhancing** collaborations with National and International organizations and institutions to develop cross-cultural understanding to adapt and thrive in the 21st century.

About the Programme

Definitions

- **Programme Outcomes (POs)**

  Programme Outcomes are statements that describe what the students are expected to know and would be able to do upon the graduation. These relate to the skills, knowledge, and behaviour that students acquire through the programme.

- **Programme Specific Outcomes (PSOs)**

  Programme Specific Outcomes define what the students should be able to do at the time of graduation and they are programme specific. There are two to four PSOs for a programme.

- **Programme Educational Objectives (PEOs)**

  Programme Educational Objectives of a degree programme are the statements that describe the expected achievements of graduates in their career, and what the graduates are expected to perform and achieve during the first few years after graduation.

- **Credit**

  Credit refers to a unit by which the course work is measured. It determines the number of hours of instructions required per week. One credit is equivalent to 14-15 periods for theory, or 28-30 periods for workshop/labs and tutorials

Programme Educational Objectives (PEO*)*

PEO1: Successful professionals in industry, government, academia, research, entrepreneurial pursuits and consulting firms.

PEO2: Able to apply their knowledge of computer science & engineering principles to solve societal problems by exhibiting a strong foundation in both theoretical and practical aspects of the field.

PEO3: Dedicated to upholding professional ethics and social responsibilities, with a strong commitment to advancing sustainability goals.

PEO4: Demonstrating strong leadership skills and a proven ability to collaborate effectively in diverse, multidisciplinary teams to successfully achieve project objectives.

Programme Outcomes (PO)

Engineering Graduates will be able to:

PO1. Core Competencies in Engineering: Graduates will possess a strong foundation in engineering knowledge, critical problem analysis, and solution design, equipped with skills for conducting thorough investigations to solve complex challenges.

PO2. Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO3. Societal and Environmental Responsibility

Apply contextual knowledge to evaluate societal, health, safety, legal, and cultural issues, while understanding the impact of engineering solutions on the environment and advocating for sustainable development.

PO4. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO5. Effective Communication and Team Collaboration

Excel in both individual and team roles within diverse and multidisciplinary settings, while communicating complex engineering concepts clearly through effective reports, presentations, and interactions.

PO6. Project management

Apply engineering and management principles to lead and manage projects effectively in computer science and engineering contexts.

PO7. Life-long learning: Embrace and actively pursue continuous learning to stay current with technological advancements and evolving practices in computer science and engineering.

Programme Specific Outcomes (PSO)

On completion of the program, students will be:

PSO1: Understanding the concepts, theories, tools, techniques, and methodologies of Computer Science & Engineering.

PSO2: Applying the concepts, theories, tools, techniques, and methodologies to solve real-world Computer Science & Engineering challenges.

PSO3: Analysing the methodologies context, problems, situations and issues related to Computer Science & Engineering.

PSO4: Evaluating the possible alternative solutions and making choices/decisions to solve problems in Computer Science & Engineering.

PSO5: Designing and **developing** innovative solutions to address complex problems in Computer Science & Engineering

Career Avenues

Graduates of the Bachelor of Technology (CSE) with specialization in AI & ML program have a wide array of career opportunities, including:

1. **Software Developer**: Design, code, and maintain software across various domains such as web, mobile, game, and enterprise applications.

2. **Systems Analyst**: Evaluate and enhance organizational computer systems, design new solutions, and optimize existing processes for improved efficiency.
3. **Data Scientist**: Analyze large datasets, apply statistical methods and machine learning, and develop predictive models to drive data-informed decisions.
4. **AI Engineer**: Develop and implement AI algorithms, including natural language processing, computer vision, and other intelligent systems.
5. **Cybersecurity Analyst**: Protect systems and data by identifying vulnerabilities, implementing security measures, and responding to security incidents.
6. **Network Engineer**: Design, deploy, and maintain reliable and secure network infrastructures, and troubleshoot network issues to ensure optimal performance.
7. **IT Project Manager**: Oversee technology projects from planning to execution, manage teams, coordinate resources, and ensure timely and budget-compliant delivery.
8. **Database Administrator**: Maintain database systems, ensure data integrity and security, and optimize performance for efficient data management.
9. **Quality Assurance Engineer**: Test and ensure the reliability and functionality of software applications by developing test plans, identifying issues, and collaborating with development teams.
10. **Research and Development**: Engage in cutting-edge research, explore emerging technologies, and contribute to innovation in academia, industry labs, or R&D departments.

Duration

4 Years (8 Semesters) - Full-Time Program

Eligibility Criteria

Passed 10+2 examination with Physics and Mathematics as mandatory course.

For remaining single course select any course from Chemistry/ Computer Science/ Electronics/ Information Technology/ Biology/ Informatics Practices/ Biotechnology/

Technical Vocational subject/ Agriculture/ Engineering Graphics/ Business Studies/ Entrepreneurship from any recognized Board/ University with minimum 50% aggregate marks.

Eligibility Criteria for Award of Degree

Students must successfully complete the minimum required credits of 166 to be eligible for award of degree without any backlog

Student's Structured Learning Experience from Entry to Exit in the Programme

a. **Education Philosophy and Purpose:**

   Learn to Earn a Living: At KRMU we believe in equipping students with the skills, knowledge, and qualifications necessary to succeed in the job market and achieve financial stability. All the programmes are tailored to meet industry demands, preparing students to enter specific careers and contributing to economic development.

   Learn to Live: The university believes in the holistic development of learners, fostering sensitivity towards society, and promoting a social and emotional understanding of the world. Our aim is to nurture well-rounded individuals who can contribute meaningfully to society, lead fulfilling lives, and engage with the complexities of the human experience.

b. **University Education Objective**

   Focus on Employability and Entrepreneurship through Holistic Education using Bloom’s Taxonomy. By targeting all levels of Bloom’s Taxonomy—remembering, understanding, applying, analysing, evaluating, and creating—students are equipped with the knowledge, skills, and attitudes necessary for the workforce and entrepreneurial success. At KRMU we emphasize on learners critical thinking, problem-solving, and innovation, ensuring application of theoretical knowledge in practical settings. This approach nurtures adaptability, creativity, and ethical decision-making, enabling graduates to excel in diverse professional environments and to innovate in entrepreneurial endeavours, contributing to economic growth and societal well-being.

c. **Importance of Structured Learning Experiences**:

A structured learning experience (SLE) is crucial for effective education as it provides a clear and organized framework for acquiring knowledge and skills. By following a well-defined curriculum, teaching-learning methods and assessment strategies, learners can build on prior knowledge systematically, ensuring that foundational concepts are understood before moving on to more complex topics. This approach not only enhances comprehension but also fosters critical thinking by allowing learners to connect ideas and apply them in various contexts. Moreover, a structured learning experience helps in setting clear goals and benchmarks, enabling both educators and students to track progress and make necessary adjustments. Ultimately, it creates a conducive environment for sustained intellectual growth, encouraging learners to achieve their full potential.

At K.R. Mangalam University SLE is designed as rigorous activities that are integrated into the curriculum and provide students with opportunities for learning in two parts:

Inside the Classroom:

Our educational approach within the classroom is designed to foster **cognitive development** and enhance **student-centric learning**. We prioritize active engagement and deep understanding by employing a variety of methods, tools, and techniques. These include **problem-based learning**, **case studies**, **interactive discussions**, and **technology-enhanced learning platforms**. Our faculty focuses on developing critical thinking, analytical reasoning, and problem-solving abilities, ensuring students achieve well-defined **cognitive outcomes**. Additionally, we integrate the use of **modern teaching tools**, such as Learning Management Systems (LMS), virtual labs, and multimedia resources, to enhance the learning experience and accommodate diverse learning styles. This comprehensive approach not only promotes academic excellence but also nurtures independent learning and lifelong intellectual curiosity.

Outside the Classroom:

Beyond the classroom, our focus shifts to developing students' **people skills** and **psychomotor skills** through hands-on experiences in **industry, community, and laboratory settings**. We encourage participation in internships, industrial visits, community engagement projects, and research opportunities, which allow students to apply theoretical knowledge to real-world challenges. These activities build essential interpersonal skills such as **teamwork, leadership, communication**, and **professional networking**. Simultaneously, students engage in **lab-based learning** and technical workshops that refine their psychomotor abilities, including precision, technical expertise, and problem-solving under practical conditions. Through these outside-the-classroom experiences, students gain a holistic skill set that prepares them to excel in both professional and societal contexts, aligning their education with real-world expectations and industry needs.

d. Educational Planning and Execution

The B.Tech in Computer Science & Engineering (CSE) at K.R. Mangalam University is designed to foster a holistic educational experience, integrating both theoretical knowledge and practical skills, aligned with the National Education Policy (NEP) 2020. The program offers students a structured path from entry to exit, ensuring they develop technical expertise, problem-solving skills, and professional competencies.

Entry Phase

Upon entering the B.Tech CSE program, students are introduced to the foundational concepts of engineering mathematics, physics/chemistry, and programming. This phase is designed to strengthen their understanding of core scientific and technical principles. Courses such as Engineering Calculus, Fundamentals of Computer Programming using Python, and Basics of Electrical & Electronics Engineering provide a strong foundation. Students also engage in hands-on laboratory sessions to complement theoretical learning, which helps them connect classroom knowledge with real-world applications.

Orientation Program: The university conducts a **one-day orientation program** for first-year students to familiarize them with the university's environment and key aspects. During the program, students are introduced to the university's highlights, important procedures, key functionaries, and the code of conduct. This orientation serves to ensure that students are well-informed and prepared for a smooth transition into university life.

In the first year, students are exposed to critical problem-solving approaches, basic programming, and ethics in engineering, laying the groundwork for their technical and professional growth.

Induction program: The School organizes a **5-day induction program** for first-year students, aimed at providing them with a comprehensive understanding of the school's various aspects. During the program, students are introduced to learning resources, facilities, and opportunities available to them, along with the rules and regulations governing academic and campus life. The induction also includes faculty introductions, guidelines on academic conduct, and detailed information about examination and evaluation methods, ensuring students are well-prepared for their academic journey.

Core Learning

As students advance through the program, they delve deeper into core computer science subjects such as Data Structures, Algorithms, Object-Oriented Programming (C++), Operating Systems, and Database Management Systems. This phase emphasizes both theoretical concepts and their practical application through lab work. The learning is enhanced through exposure to industry-standard tools and techniques, including programming languages like Java and Python, and systems for data management and networking.

The structured academic schedule, with a well-distributed credit system over eight semesters, ensures students acquire deep technical knowledge and skills in software development, systems design, and computing technologies. The Summer Internship Programs and Minor Projects in the curriculum allow students to apply their learning in real-life projects, facilitating experiential learning.

Summer Internships: School offers 2-credit summer internships spanning 6 weeks, where students are encouraged to pursue internships in startups, industries, or premier institutions such as IITs, NITs, and IIITs. In addition, students have the opportunity to earn global certifications during this period. The School also organizes in-house summer schools in collaboration with industry partners, providing further avenues for students to gain hands-on experience and enhance their professional skills. These initiatives are designed to offer students practical exposure, helping them develop industry-relevant expertise.

Value Added Courses: The School offers a range of 2-credit Value Added Courses (VACs) designed to equip students with industry-relevant skills. These courses aim to bridge the gap between academic knowledge and practical application by providing hands-on training that aligns with current industry demands, ensuring that students are well-prepared for professional challenges.

Skill Development

Throughout the program, there is a significant emphasis on developing practical skills and ensuring students are industry-ready. Courses on Artificial Intelligence, Machine Learning, Cloud Computing, and Cybersecurity provide students with cutting-edge knowledge in emerging fields. Value-Added Courses (VAC) like AWS Cloud Fundamentals, Software Testing, Cyber Security, and Design Thinking & Innovation help bridge the gap between academic learning and industry demands.

Collaborative projects, internships, and industry-based certification courses (offered through partnerships with organizations like IBM and Samatrix) further develop students' practical and professional skills, preparing them to thrive in a dynamic workplace.

Capstone and Exit Phase

In the final semesters, students undertake discipline-specific electives and capstone projects. These projects integrate the knowledge and skills they have acquired over the course of their studies. Electives such as Natural Language Processing, Generative AI, and Blockchain Technologies offer students the flexibility to specialize in areas of their interest.

The final Industrial Project or R&D Project in the eighth semester is a full-time engagement where students work on live industry problems, research projects, or start-up ideas. This project phase, combined with career readiness boot camps and placement preparation activities, ensures that students are equipped to enter the workforce with both technical competence and professional acumen.

Co-Curricular and Extra-Curricular Activities

Students are encouraged to participate in various clubs, societies, and extra-curricular activities. Engagement in activities such as hackathons, coding competitions, and leadership roles in clubs fosters teamwork, leadership, and creativity. These activities complement academic learning, contributing to the students' holistic development.

Community Connect

The B.Tech CSE program includes community engagement through activities like Extension Projects and social service initiatives. Students work on community projects and participate in programs aimed at addressing local and national challenges, promoting civic responsibility, and developing empathy towards society.

Ethics and Professional Values

The program places a strong emphasis on ethics and professionalism. Students are taught to incorporate ethical considerations in technological development and decision-making processes. This prepares them to not only be skilled engineers but also responsible professionals who contribute positively to society.

Career Counselling and Entrepreneurship

The university offers comprehensive **career counselling services**, providing students with expert guidance on **job placements, internships**, and **skill development** to help them effectively navigate their career paths. In addition, the university's **incubation center** plays a pivotal role in nurturing **entrepreneurial and leadership skills**, empowering students to explore innovative ideas and launch their own ventures. These initiatives are designed to equip students with the tools and resources necessary for professional success and entrepreneurial growth.

Course Registration

- Every student has to register at the beginning of each semester for the courses offered in the given semester. Major courses are registered centrally for the students. However, for other multidisciplinary courses (DSE, VAC, OE) the students have to register by themselves through ERP.

e. Student Support Services

Mentor-Mentee: At K.R. Mangalam University, the **Mentor-Mentee Program** plays a crucial role in fostering academic and personal growth. Each student is assigned a faculty mentor who serves as a guide throughout their academic journey. This program ensures continuous interaction, where mentors assist students with academic planning, help in resolving personal issues, and provide career guidance. The mentor-mentee relationship transcends the classroom and often involves personal development, professional growth, and overall well-being. The program aims to nurture a supportive environment that enhances the learning experience and helps students reach their full potential.

Counselling and Wellness Services: The university places a strong emphasis on the mental and emotional well-being of its students through its Counselling and Wellness Services. A dedicated team of trained counselors provides personalized sessions, workshops, and wellness programs to address the mental health needs of the student community. These services focus on holistic well-being, including stress management, emotional resilience, and coping strategies. Regular wellness programs, meditation sessions, and mental health awareness campaigns are conducted to promote a balanced lifestyle and ensure that students can focus on their studies while maintaining their emotional health.

Evaluation of Learning:

At K.R. Mangalam University, assessment and evaluation are integral components of the teaching-learning process, designed to ensure continuous academic progress and holistic development of students. The university follows a Learning Outcome-Based Framework (LOCF), where assessments are aligned with the specific learning outcomes of each program. A variety of assessment methods, including assignments, presentations, quizzes, practical examinations, and project work, are used to gauge students' understanding. The examination system is 100% automated, ensuring timely and transparent evaluation processes. Results are processed efficiently, typically within 13 days, and complaints related to evaluation are minimal, reflecting the university's commitment to maintaining a high standard of academic integrity. This robust system of continuous assessment and feedback fosters a culture of academic excellence and skill development among students.

I. Evaluation Scheme (Theory):

	Evaluation Components	**Weightage**
	Internal Marks (Theory)	
1.	**Continuous Assessment (30 Marks)** **(All the components to be evenly spaced)** Project/ Quizzes/ Assignments and Essays/ Presentations/ Participation/ Case Studies/ Reflective Journals (minimum of five components to be evaluated)	**30 Marks**
2.	**Internal Marks (Theory) – Mid Term Exam**	**20 Marks**
	External Marks (Theory): - End term Examination	**50 Marks**

	Total	**100 Marks**

***Note:** (It is compulsory for a student to secure 40% marks in Internal and End Term Examination separately to secure minimum passing grade).

II. Evaluation Scheme (Laboratory/Practical Courses):

Evaluation Components	Weightage
Internal Marks (Practical) –	
1. Conduct of Experiment	10 Marks
2. Lab Records	10 Marks
3. Lab Participation	10 Marks
4. Lab Project	20 Marks
External Marks (Practical): - End term Practical Exam and Viva Voce	50 Marks
Total	**100 Marks**

a. **Feedback and Continuous Improvement Mechanisms:**

K.R. Mangalam University is deeply committed to academic excellence through a robust **feedback and continuous improvement system**. This system is designed to gather comprehensive input from a diverse range of stakeholders, including **students, faculty, alumni, employers, and academic peers**. Feedback is systematically collected and thoroughly analyzed to identify areas for enhancement in **curricula, teaching methodologies, and academic processes**. Based on the insights gained, actionable measures are formulated and communicated to the appropriate bodies for timely implementation.

This structured feedback mechanism ensures that the university's programs remain aligned with **industry trends and societal needs**, providing students with a cutting-edge education that prepares them for real-world challenges. Moreover, the university demonstrates its commitment to continuous improvement through **regular curriculum updates** and the integration of **innovative teaching strategies**, fostering an environment where both faculty and students can grow and excel. By maintaining this cycle of feedback and improvement, K.R. Mangalam University ensures the continuous advancement of its academic offerings and the overall learning experience.

b. **Academic Integrity and Ethics:**

K.R. Mangalam University upholds the highest standards of academic integrity and ethics as a core value of its educational philosophy. The university implements a zero-tolerance policy towards academic misconduct, including plagiarism and other unethical practices. To ensure transparency and honesty in academic work, plagiarism detection software like Drillbit is used to maintain the originality of student submissions and research outputs. Students and faculty are regularly sensitized on the importance of ethical behavior through workshops, seminars, and classroom discussions. The university also integrates ethics and professional values into its curriculum across various disciplines, ensuring that graduates not only excel academically but also demonstrate integrity and responsibility in their professional and personal lives.

Scheme of Studies

Program Name	B.Tech (CSE) with specialization in AIML
Total Credits	168
Total Semesters	8

Credit Distribution Summary

Program Name	I	II	III	IV	V	VI	VII	VIII	Total Credits

B.Tech CSE (Cyber)	20	23	27	26	23	23	14	12	168

Semester I (Odd Semester)

SNo	Category	Course Code	Course Title	L	T	P	C	
1	Major-1	ENMA101	Engineering Calculus	3	1	-	4	
2	IDC-1	ENSP101	Clean Coding with Python	4	0	0	4	IBM
3	Major-2	ENPH101/ENCH101	Engineering Physics / Engineering Chemistry	3	1	-	4	
4	SEC-1	**SEC033**	Engineering Drawing & Workshop Lab	-	-	4	2	
5	IDC-2	ENSP151	Clean Coding with Python Lab	0	0	2	1	IBM
6	Major-3	ENPH151/ENCH151	Engineering Physics lab / Engineering Chemistry lab	-	-	2	1	
7	VAC-1		Environmental Studies & Disaster Management (Online Moodle)	2	-	-	2	
8	SEC-1	**SEC037**	Data Visualization using PowerBI	0	0	4	2	Samatrix
TOTAL				12	2	12	20	

Semester II (Even Semester)

SNo	Category	Course Code	Course Title	L	T	P	C	
1	Major-4	ENMA102	Linear Algebra and Ordinary Differential Equations	3	1	-	4	
2	Major-5	ENCH101/ENPH101	Engineering Chemistry / Engineering Physics	3	1	-	4	
3	Major-6	ENCS102	Object Oriented Programming using C++	3	1	-	4	
4	Major-7	ENCH151/ENPH151	Engineering Chemistry Lab/Engineering Physics lab	-	-	2	1	
5	Major-8	ENCS152	Object Oriented Programming using C++ Lab	-	-	2	1	
6	IDC-3	ENSP152	Overview of AI, Data Science, Ethics and Foundation of Data Analysis Lab	0	0	4	2	Samatrix
8	Open Elective-1		Students can choose one of the electives from the pool of open electives of University	3	-	-	3	
9	Proj-1	ENSI152	Minor Project-I	-	-	-	2	
10	SEC-2		Applied Generative AI: Practical Tools and Techniques	-	-	4	2	
TOTAL				12	3	12	23	

SEMESTER III (ODD SEMESTER)

S.N	Category	Course Code	Course Title	L	T	P	C	
1	Major-9	ENCS203	Discrete Mathematics	3	1	-	4	
2	Major-10	ENCS205	Data Structures	3	1	-	4	
3	Major-11	ENCS201	Java Programming	4	-	-	4	
4	Major-12	ENCS253	Data Structures Lab	-	-	2	1	
5	Open Elective-2		Students can choose one of the electives from the pool of open electives of University	3	-	-	3	Open Elective
6	SEC-3	SEC038	Probabilistic Modelling and Reasoning with Python Lab	-	-	4	2	Samatrix
7	VAC-2		VAC II	2	-	-	2	
8	AEC-1	AEC006	Verbal Ability	3	-	-	3	
9	INT-1	ENSI251	Summer Internship-I	-	-	-	2	

10	Major-13	ENCS251	Java Programming Lab	-	-	2	1	
11	**AUDIT-1**		Competitive Coding - I	**2**	**-**	**-**	**0**	
12	CS-1	CS001	Club/Society	1	-	-	1	
TOTAL				21	2	8	27	

*VAC-III					
CODE	COURSE TITLE	L	T	P	C
VAC170	Design thinking & Innovations for Engineers	-	-	-	2
VAC171	AWS Cloud Fundamentals	-	-	-	2
VAC172	Web Development with open-source Frameworks	-	-	-	2
VAC173	Google Data Analytics	-	-	-	2
VAC174	Software Testing using Open-Source Frameworks	-	-	-	2
VAC175	Database Management with Open-Source Frameworks	-	-	-	2
VAC176	Cyber Security with Open-source Frameworks	-	-	-	2
VAC185	Practical Robotics and UAV Applications	-	-	-	2
VAC186	Applied Automotive Engineering: Hands-On Practices and Innovations	-	-	-	2
VAC187	Practical Research Methodology for Engineers	-	-	-	2

Semester IV (Even Semester)

SN	Category	Course Code	Course Title	L	T	P	C	
1	Major-13	ENCS202	Analysis and Design of Algorithms	3	1	-	4	
2	Major-14	ENCS204	Database Management Systems	3	1	-	4	
3	IDC-4	ENSP202	Machine Learning and Pattern Recognition	4	-	-	4	Samatrix
4	Major-15	ENCS256	Analysis and Design of Algorithms Lab	-	-	2	1	
5	Major-16	ENCS254	Database Management Systems Lab	-	-	2	1	
6	AEC-2	AEC007	Communication & Personality Development	3	-	-	3	
7	Open Elective-3		Students can choose one of the electives from the pool of open electives of University	3	-	-	3	
9	IDC-5	ENSP252	Machine Learning Practical with Python, Scikit-learn, Matplotlib, TensorFlow	-	-	2	1	Samatrix
10	Proj-2	ENSI252	Minor project-II	-	-	-	2	
11	SEC-4	SEC039	R Programming for Data Science and Data	-	-	4	2	Samatrix

			Analytics Lab					
11	**AUDIT-2**		Competitive Coding- II	**2**	**-**	**-**	**0**	
12	CS-2	CS002	Community Service	**1**	**-**	**-**	**1**	
TOTAL				19	2	10	26	

Semester V (Odd Semester)

SNo	Category	Course Code	Course Title	L	T	P	C	
1	Major-17	ENCS301	Theory of Computation	3	1	-	4	
2	Major-18	ENCS303	Operating Systems	3	1	-	4	
3	IDC-6	ENSP302	Natural Language Processing	4	-	-	4	Samatrix
4	AEC-3	**AEC008**	**Arithmetic and Reasoning Skills**	3	-	-	3	
6	IDC-7	ENSP352	Natural Language Processing lab	-	-	2	1	Samatrix
7	SEC-5	SEC040	Data Science - Tools and Techniques Lab	-	-	4	2	Samatrix
8	Major-19	ENCS351	Operating System Lab	-	-	2	1	
9	IDC-8	ENSP359	Big Data Analysis with Scala and Spark Lab	-	-	4	2	IBM
10	INT-2	ENSI351	Summer Internship-II	-	-	-	2	

11	**AUDIT-3**		Competitive Coding - III	**2**	**-**	**-**	**0**	
TOTAL				15	2	12	23	

Students can choose one of the following Discipline Specific Electives:

	Discipline Specific Elective I (Artificial Intelligence)						
(i)	Minor	ENSP304	Image Processing & Computer Vision	4	-	-	4
	Minor	ENSP354	Image Processing & Computer Vision lab	-	-	2	1
(ii)	Minor	ENSP306	Introduction to Generative AI	4	-	-	4
	Minor	ENSP356	Generative AI lab	-	-	2	1
(iii)	Minor	ENSP308	Transfer Learning	4	-	-	4
	Minor	ENSP358	Transfer Learning lab	-	-	2	1

Semester VI (Even Semester)

SNo	Category	Course Code	Course Title	L	T	P	C	
1	Major-20	ENCS302	Computer Organization & Architecture	3	1	-	4	
2	DSE-1		Discipline Specific Elective - I	4	-	-	4	
3	Major-21	ENCS304	Computer Networks	4	-	-	4	
4	Major-22	ENSP310	Neural Networks and Deep Learning	4	-	-	4	Samatrix
5	Major-23	ENCS352	Computer Networks Lab	-	-	2	1	
6	Major-24	ENSP360	Deep Learning Practical with Python, TensorFlow and Keras	-	-	2	1	Samatrix
7	DSE-2		Discipline Specific Elective -I	-	-	2	1	
8	Proj-3	ENSI352	Minor Project-III				2	

9	**AUDIT-4**		Competitive Coding - IV	**2**	**-**	**-**	**0**	
10	MOOC-1		MOOC in the domain of AI & ML (Swayam/ NPTEL/AICTE's ELIS)	-	-	-	**2**	
TOTAL				17	1	6	23	

Discipline Specific Elective II(Cyber Security)							
(i)	DSE	ENSP301	Secure Coding and Vulnerabilities	4	-	-	4
	DSE	ENSP351	Secure Coding and Vulnerabilities lab	-	-	2	1
(ii)	DSE	ENSP303	Cyber Crime Investigation & Digital Forensics	4	-	-	4
	DSE	ENSP353	Cyber Crime Investigation & Digital Forensics lab	-	-	2	1
(iii)	DSE	ENSP305	AI in Cyber Security	4	-	-	4
	DSE	ENSP355	AI in Cyber Security Lab	-	-	2	1
(iv)	DSE	ENSP307	Social Media Security	4	-	-	4
	DSE	ENSP357	Social Media Security Lab	-	-	2	1

Semester VII (Odd Semester)

SNo	Category	Course Code	Course Title	L	T	P	C
1	DSE-3		Discipline Specific Elective -II	4	-	-	4
2	DSE-4		Discipline Specific Elective -III	4	-	-	4
3	DSE-5		Discipline Specific Elective -II Lab	-	-	2	1
4	DSE-6		Discipline Specific Elective III Lab	-	-	2	1
5	INT-3	ENSI451	Summer Internship-III	-	-	-	2
6	MOOC-2		Applied Programming and Problem-Solving Skills for Campus Interviews	-	-	-	2
	TOTAL			8	0	4	14

Note:

- Students can choose among the following Discipline Specific Electives:

Discipline Specific Elective - III (Full Stack Development)							
(i)	DSE	ENSP409	Mobile Application Development using iOS	4	-	-	4
	DSE	ENSP459	Mobile Application Development using iOS Lab	-	-	2	1
(ii)	DSE	ENSP411	DevOps & Automation	4	-	-	4
	DSE	ENSP461	DevOps & Automation Lab	-	-	2	1
(iii)	DSE	ENSP413	.Net FRAMEWORK	4	-	-	4
	DSE	ENSP463	.Net FRAMEWORK Lab	-	-	2	1
(iv)	DSE	ENSP415	New Age Programming languages	4	0	0	4
	DSE	ENSP465	New Age Programming languages Lab	0	0	2	1

Semester VIII (Even Semester)

SN	Category	Course Code	Course Title	L	T	P	C
1	PROJ-4	ENSI452	Industrial Project/R&D Project/Start-up Project	-	-	-	12
TOTAL							12

Note:

- For the "Summer Internship," students are required to complete a 6-week full-time industry internship during the summer and submit a completion certificate. The evaluation will occur in the 7th semester, with students graded on a scale of 100 marks.

- Students are required to undertake a full-time industry internship for the entire semester. They are not permitted to enroll in any courses as an alternative to the internship. Students can choose from the following internship options:
  - Industry project
  - Research & Development project
  - Start-up project

  Evaluation will be based on internal assessments, with no end-term exams applicable.

Syllabus

Semester: 1

ENGINEERING CALCULUS

Program Name	**B. Tech (Computer Science and Engineering)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Engineering Calculus	**ENMA101**	3-1-0	4
Type of Course:	Major-1		
Version			
Total Contact Hours			

Pre-requisite(s): Calculus knowledge at higher secondary level

Course Perspective. This course is to familiarize students with techniques in calculus, multivariate calculus, vector calculus, and their applications. It aims to equip students with standard concepts and tools from intermediate to advanced levels that will enable them to tackle more advanced mathematical and engineering problems relevant to their disciplines. The course is divided into 4 modules:

a) Differential Calculus- I
b) Multivariable Calculus (Partial Differentiation and applications)
c) Multivariable Calculus-II (Integration)
d) Vector Calculus

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Recalling** fundamental concepts such as limits, derivatives, integrals, and convergence tests for series and sequences
CO 2	**Understanding** and interpret the geometric interpretations of calculus theorems like Mean Value Theorem and Taylor's Theorem
CO 3	**Applying** differential calculus techniques to optimize engineering solutions, including finding maxima and minima of functions. Use multivariable calculus methods to calculate volumes, surface areas, and centers of mass in practical engineering scenarios
CO 4	**Analyzing** complex functions using Taylor and Maclaurin series for approximation and representation

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Differential Calculus- I**	**No. of hours: 10**

Content:

- Introduction to limits, continuity, and differentiability.
- Rolle’s Theorem, Lagrange’s Mean value theorem with geometrical interpretation and applications.
- Cauchy’s Mean value Theorem.
- Taylor’s Series.
- Applications of definite integrals to evaluate surface areas and volumes of revolutions of curves (Cartesian coordinates).

- Successive Differentiation, Leibnitz theorem and its application.
- Curve tracing in Cartesian and Polar coordinates.
- Infinite series: Tests for convergence of series (Comparison, Ratio, Root test)
- Alternating series
- Absolute convergence
- Conditional convergence.

Unit Number: 2	Title: Multivariable Calculus (Partial Differentiation and applications)	No. of hours: 10

Content:

- Partial derivatives.
- Total derivative.
- Euler's Theorem for homogeneous functions.
- Taylor and Maclaurin theorems for functions of one and two variables.
- Maxima and Minima of functions of several variables.
- Lagrange Method of Multipliers.

Unit Number: 3	Title: Multivariable Calculus-II (Integration)	No. of hours: 10

Content:

- Area between two curves; Polar Coordinates.
- Volumes by slicing, Washer and Shell Methods.
- Length of a plane curve.
- Areas of Surfaces of Revolution.
- Evaluation of Double Integrals (Cartesian and polar coordinates).
- Change of order of integration (Cartesian form).
- Evaluation of Triple Integrals: Change of variables (Cartesian to polar for double, Cartesian to Spherical and Cylindrical polar for triple integrals).
- Applications: Areas (by double integrals) and volumes (by double and triple integrals).
- Centre of mass and centre of gravity (Constant and variable densities).

Unit Number: 4	Title: Vector Calculus	No. of hours: 10

Content:

- Vector differentiation: Gradient, Curl, and Divergence with physical interpretation.
- Directional derivatives, Tangent and Normal planes.
- Vector Integration: Line integral, Surface integral, Volume integral.
- Applications to work done by the force
- Gauss's Divergence theorem, Green's theorem, Stoke's theorem (without proof) and applications.

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Use PPTs and MATLAB to explain key calculus concepts.
2. **Conceptual Understanding**: Cover theorems (Rolle's, Taylor's, etc.) and solve problems.
3. **Problem-Solving Sessions**: In-class exercises on differential and multivariable calculus.
4. **Theory Assignments**: Solve theoretical problems, reviewed in class.
5. **Group Work**: Collaborative problem-solving for real-world engineering tasks.
6. **Case Studies**: Discuss real-world applications of calculus concepts.
7. **Continuous Feedback**: In-class quizzes and feedback sessions.

Outside Classroom Learning Experience

1. **Theory Assignments**: Apply calculus techniques in take-home assignments.
2. **Lab Projects**: Work on hands-on calculus applications using software.
3. **Question Bank**: Practice with model papers and self-assessment.
4. **Online Forums**: Discuss and collaborate on calculus problems online.
5. **Self-Study for Case Studies**: Research and apply calculus to real-world scenarios.
6. **Collaborative Projects**: Group work on applying multivariable and vector calculus.

Textbooks:

- G.B. Thomas and R.L. Finney, Calculus and Analytic Geometry, 9th Edition, Pearson, Reprint, 2002.

Reference Books:

- B. V. Ramana, Higher Engineering Mathematics, Tata Mc Graw-Hill, 2008.
- B. S. Grewal, Higher Engineering Mathematics, Khanna Publisher, 2005.
- R K. Jain & S R K. Iyenger, Advance Engineering Mathematics, Narosa Publishing House, 2002.
- E. Kreyszig, Advanced Engineering Mathematics, John Wiley & Sons, 2005.
- Ray Wylie C and Louis C Barret, Advanced Engineering Mathematics, Tata Mc-Graw-Hill, Sixth Edition.

Additional Readings:

1. Link to NPTEL course contents: https://onlinecourses.nptel.ac.in/noc18_ma05/preview
2. Link to topics related to course: https://www.whitman.edu/mathematics/calculus_online/chapter14.html

CLEAN CODING WITH PYTHON

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: CLEAN CODING WITH PYTHON	**Course Code**	**L-T-P**	**Credits**
	ENSP101	4-0-0	4
Type of Course:	IDC-1		
Pre-requisite(s), if any:			

Course Perspective. "Clean Coding with Python" is designed to teach students the principles and practices of writing clean, maintainable, and efficient Python code. This course covers essential coding standards, best practices, and design patterns that promote readability, simplicity, and scalability in software development.

The course is structured into four comprehensive modules:

- Introduction to Python
- Python Data Structure
- Python Decorators and generators
- Python advanced modules

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Working** with user input to create fun and interactive programs.
CO 2	**Developing**, run and manipulate Python programs using Core data structures like Lists, Dictionaries, and use of Strings Handling methods.

CO 3	**Developing**, run and manipulate Python programs using File Operations and searching pattern using regular expressions.
CO 4	**Determining** the need for scraping websites and working with CSV, JSON and other file formats.
CO 5	**Creating** simple games with images, animations, and audio using our custom beginner-friendly programming library.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1 **Title: Introduction to Python** **No. of hours: 8**

Content:
Python Introduction and Setup: Command Line Basics, Installation of Python. Text Editor (VS Code, PyCharm, Anaconda)
Python basics and control structures: Python data types,Numbers, Variables, Getting input from the user, Operators,Statements (If, else, elif), Nested statements, Loops and loop control statements (Break, continue and pass), Strings (Indexing, slicing and formatting).

Unit Number: 2 **Title: Python Data Structure** **No. of hours: 10**

Content:
Python Data Structures: Lists, Tuples, Sets, Dictionaries. Methods and Functions: Introduction to functions, def keyword, *args and **kwargs in python, exercise on functions, Lambda expressions, Map and Filter functions.

Unit Number: 3 **Title: Python Decorators and generators** **No. of hours: 10**

Content:
Modules and Package : Installation using pip
Errors and Exception Handling: Errors, Exceptions, Try and Except Statement, Catching Specific Exception, Try with else, Finally, Keyword, Raising an exception.
File Handling using Python.

Unit Number: 4 **Title: Python advanced modules** **No. of hours: 10**

Content:
Python advanced modules: Datetime module, Math and Random module, OS module
Regular Expressions: re module, Web Scraping using Python: Web Scraping libraries and practical implementation, Working with images using

python

Unit Number: 5	**Title: Working with Excel sheets and CSV files**	**No. of hours: 8**

Content :
Python GUI programming: Tkinter, Adding Widgets, Buttons etc.SQL queries (DDL, DML, DCL, TCL) – Joins, Sub-Queries, Constraints and Inbuilt functions (Date, String, Math)Database handling in python using MySQL db, Fetching and Inserting data using MySQL db

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Use PPTs and coding demos to explain clean coding principles.
2. **Conceptual Understanding**: Cover key topics like readability, modularity, and documentation.
3. **Problem-Solving Sessions**: Conduct in-class coding exercises focusing on refactoring and best practices.
4. **Theory Assignments**: Assign projects requiring clean code principles, reviewed in class.
5. **Group Work**: Collaborate on coding tasks to promote peer review and collective learning.
6. **Case Studies**: Analyze examples of well-written Python code and common pitfalls.
7. **Continuous Feedback**: Implement in-class code reviews and quizzes for ongoing assessment.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home coding projects emphasizing clean coding techniques.
2. **Lab Projects**: Facilitate hands-on programming tasks using real-world scenarios.
3. **Question Bank**: Provide practice problems and resources for self-assessment.
4. **Online Forums**: Create platforms for students to discuss coding challenges and solutions.
5. **Self-Study for Case Studies**: Encourage independent research on best practices in Python programming.
6. **Collaborative Projects**: Organize group projects focusing on developing clean, efficient code.

Text and Reference Book

1. J. Peterson, A. Silberschatz, and P. Galvin, “Operating System Concepts”, Addison Wesley. 2012
2. V. Aho, R. Sethi, and J. D. Ullman, “Compilers: Principles, Techniques and Tools”, Addison-Wesley. 2013
3. R. El. Masri and S. B. Navathe, “Fundamentals of Data Base Systems”,Benjamin Cummings. 2013

Additional Readings:

R 1. https://www.tutorjoes.in/python_programming_tutorial/
R 2. https://www.udemy.com/course/100-days-of-code/
R 3. https://favtutor.com/blog-details/7-Python-Projects-For-Beginners
R 4. https://github.com/NaviRocker/100-days-of-python
R 5. https://hackr.io/blog/python-projects

Online Learning Resources

1. **Codecademy**
   - Offers interactive Python courses that cover basic to advanced programming concepts, including data types, control structures, functions, and object-oriented programming.
   - Link: Codecademy Python
2. **Python.org**
   - The official Python website provides a comprehensive beginner's guide, documentation, and tutorials to get started with Python programming.
   - Link: Python Beginner's Guide

CLEAN CODING WITH PYTHON LAB

Department:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Clean coding with Python Lab	**ENSP151**	0-0-2	1
Type of Course:	IDC-2		
Pre-requisite(s), if any: Integration/Differentiation			

Defined Course Outcomes

COs	Statement
CO 1	**Developing** solutions to simple computational problems using Python programs.
CO 2	**Solving** problems using conditionals and loops in Python. DevelopPython programs by defining functions and calling them.
CO 3	**Implementing** Python lists, tuples and dictionaries for representing compound data.
CO 4	**Understanding** the Machine Learning Algorithms.

Proposed Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	Develop programs to understand the control structures of python	CO1
2	Develop programs to implement list	CO2
3	Develop programs to implement Dictionary	CO2

4	Develop programs to implement tuples	CO1
5	Develop programs to implement function with stress on scoping	CO1
6	Develop programs to implement classes and objects	CO2
7	Develop programs to implement exception handling.	CO2
8	Develop programs to implement linear search and binary search.	CO3
9	A) Develop programs to implement insertion sort	CO3
10	Develop programs to implement bubble sort.	CO3
11	Develop programs to implement quick sort.	CO3
12	Develop programs to implement heap sort.	CO3

Projects to be covered: (at least 4-5 projects).

Projects Title:

- Weather Forecasting App
- Web scraping Facebook bot
- Tic tac toe game
- Snake and ladder game
- Multiplayer Game - Connect4

Online learning resources

- **Codecademy**
  - Interactive coding platform that offers hands-on Python courses, teaching both the basics and more advanced topics in Python. Ideal for practicing specific programming tasks.
  - Link: Codecademy Python Course
- **HackerRank**
  - Provides a vast range of programming problems across various domains of computer science, along with a dedicated Python domain. Great for practicing coding skills and understanding algorithms.

- Link: HackerRank Python

- **LeetCode**
  - Known for its extensive array of programming challenges that can help improve your understanding of data structures and algorithms. It's particularly good for preparing for technical job interviews.
  - Link: LeetCode

ENGINEERING PHYSICS

<table>
<tr><td>Program Name</td><td colspan="3">Bachelor of Technology (CSE) with specialization in AI & ML</td></tr>
<tr><td rowspan="2">Course Name:
Engineering Physics</td><td>Course Code</td><td>L-T-P</td><td>Credits</td></tr>
<tr><td>ENPH101</td><td>4-0-0</td><td>4</td></tr>
<tr><td>Type of Course:</td><td colspan="3">Major-2</td></tr>
<tr><td>Version</td><td colspan="3"></td></tr>
<tr><td>Total Contact Hours</td><td colspan="3"></td></tr>
<tr><td colspan="4">Pre-requisite(s), if any: Integration/Differentiation</td></tr>
</table>

Course Perspective. This course introduces students to the fundamental concepts of Engineering Physics, bridging the gap between theoretical physics principles and practical engineering applications. Engineering Physics is crucial for understanding and designing new technologies and systems in various engineering fields such as electronics, materials science, and mechanical engineering. Students will explore core topics including mechanics, Optics, Polarization, and modern physics, with a special emphasis on their relevance to real-world engineering problems. The course is divided into 4 modules:

a) Mechanics
b) Optics
c) Polarization
d) New Engineering Materials

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the principles and applications of lasers, fiber optics, and electromagnetic waves.
CO 2	**Applying** the concepts of polarization to analyze and manipulate light in various optical systems.
CO 3	**Evaluating** the properties and applications of dielectric materials, superconducting materials, and nano-materials in engineering contexts.
CO 4	**Designing** and propose innovative applications of lasers, fiber optics, and smart materials for specific engineering challenges.
CO 5	**Analyzing** problems related to the behavior of electromagnetic waves, polarization, and optical communication systems**.**

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1 Title: Mechanics No. of hours: 10

Content:

Centre of mass, centre of mass of two particle system and a rigid body, Rotational motion, Moment of Inertia and its physical significance, Radius of gyration, Acceleration due to gravity, simple harmonic motion, differential equation of S.H.M., Examples of S.H.M. (simple and compound pendulum)

Unit Number: 2	Title: Optics	No. of hours: 10

Content:

Light: Introduction of light, properties of light, Dual Nature of light, refraction, Refraction by prism, Interference of light, interference by divison of wavefront (Young's double slit experiment), Interference by division of wave amplitude (Newton's ring), difference between diffraction and interference, types of diffraction, Fraunhoffer diffraction (single and double slit), theory of plane diffraction grating, determination of wavelength of a spectral line using transmission grating

Laser: Introduction, principle of Laser, stimulated and spontaneous emission, Ruby laser, He-Ne Laser, Application of Lasers.

Unit Number: 3	Title: Polarization	No. of hours: 10

Content:

Polarization: Polarization by reflection and refraction, Brewster's law, double refraction, nicol prism, quarter and half-wave plates, Production and analysis of circularly and elliptically polarized light

Unit Number: 4	Title: New Engineering Materials	No. of hours: 10

Content:

Dielectric materials: Definition – Dielectric Breakdown – Dielectric loss – Internal field – Claussius Mossotti relation.

Superconducting materials: Introduction – Properties- Meissner effect – Type I & Type II superconductors – BCS theory-Applications.

Nanomaterials: Introduction – Synthesis of nano materials – Top down and Bottom-up approach- Ball milling- PVD method- Applications. Smart materials: Shape memory alloys-Biomaterials (properties and applications)

Learning Experience

Classroom Learning Experience

1. **Interactive Lectures**: Use PPTs and simulations to explain key physics concepts.
2. **Conceptual Understanding**: Cover fundamental principles and solve related problems.
3. **Problem-Solving Sessions**: Conduct in-class exercises on mechanics, optics, and modern physics.
4. **Theory Assignments**: Assign theoretical problems with solutions discussed in class.
5. **Group Work**: Engage in collaborative problem-solving for engineering applications.
6. **Case Studies**: Analyze real-world applications of physics concepts.
7. **Continuous Feedback**: Implement in-class quizzes and feedback sessions.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects applying physics concepts to real-world situations.
2. **Lab Projects**: Facilitate hands-on experiments to explore physics principles.
3. **Question Bank**: Provide practice problems and model papers for self-assessment.
4. **Online Forums**: Create platforms for students to discuss and collaborate on problems.
5. **Self-Study for Case Studies**: Encourage independent research on engineering applications.
6. **Collaborative Projects**: Organize group projects on practical applications of physics concepts.

Text and Reference Book

1. Fundamentals of Physics" by David Halliday, Robert Resnick, and Jearl Walker
2. University Physics with Modern Physics" by Hugh D. Young and Roger A. Freedman
3. Engineering Physics" by Gaur and Gupta
4. Concepts of Modern Physics" by Arthur Beiser
5. Physics for Scientists and Engineers" by Raymond A. Serway and John W. Jewett

Additional Readings:

Online Learning Resources for Engineering Physics

R 1. MIT OpenCourseWare - Physics

- MIT offers a variety of free courses in physics that cover topics from classical mechanics to quantum physics.
- Link: MIT OpenCourseWare Physics

R 2. Physics LibreTexts

- A comprehensive online library covering numerous topics in physics at various levels of complexity. It's part of the LibreTexts project, which aims to develop freely accessible textbooks.
- Link: Physics LibreTexts

ENGINEERING PHYSICS LAB

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Engineering Physics Lab	**ENPH151**	0-0-2	1
Type of Course:	Major-3		
Contact Hours			
Version			
Pre-requisite(s), if any:	**Integration/Differentiation**		

Defined Course Outcomes

COs	
CO 1	**Understanding** the principles and concepts related to the experiments involving bar pendulum, flywheel, Kater's pendulum, Newton's ring apparatus, plane diffraction grating, spectrometer, and half shade polarimeter.
CO 2	**Applying** the principles and concepts learned to conduct experiments and analyze experimental data, plot graphs, and interpret the results to determine various physical quantities.
CO	**Evaluating** the accuracy and reliability of experimental measurements and

3	results obtained from the conducted experiments.
CO4	**Applying** critical thinking and problem-solving skills to troubleshoot experimental setups, identify sources of errors, and propose solutions to improve the accuracy and precision of measurements

Lab Experiments

Experiment Title	Mapped CO/COs
To plot a graph between the distance of the knife edge from the center of gravity and the time period of the bar pendulum. From the graph, find the acceleration due to gravity, the radius of gyration and the moment of inertia of the bar about an axis.	CO2, CO3
To determine the moment of inertia of a flywheel about its own axis of motion.	CO1, CO2, CO3, CO4
To determine the value of acceleration due to gravity using Kater`s pendulum.	CO1, CO2, CO3, CO4

To determine the wavelength of sodium light using Newton`s ring apparatus.	CO1, CO2, CO3
To determine the wavelength of prominent lines of mercury by plane diffraction grating.	CO1, CO2, CO3
To determine the refractive index of the material of the prism for the given colours (wavelengths) of mercury light with the help of spectrometer.	CO1, CO2, CO3
To determine the specific rotation of cane sugar solution with the help of half shade polarimeter.	CO1, CO2, CO3, CO4
To determine the wavelength of He-Ne LASER using transmission diffraction grating.	CO1, CO2, CO3

ENGINEERING DRAWING & WORKSHOP LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Engineering Drawing and Workshop Lab	**Course Code**	**L-T-P**	**Credits**
	SEC033	0-	2

0-
4

Type of Course: SEC-1

Pre-requisite(s), if any:

Proposed Lab Experiments

Defined Course Outcomes

COs	Statements
CO1	**Understanding** the polygons, circles and lines with different geometric conditions
CO2	**Drawing** the projection of points, lines and planes under different conditions and orthographic views from isometric views of simple objects
CO3	**Determining** manufacturing methods in different fields of engineering and Practical exposure to different fabrication techniques
CO4	**Creating** of simple components using different materials
CO5	**Exposing** to some of the advanced and latest manufacturing techniques being employed in the industry.

EXPERIMENT NO.	EXPERIMENT TITLE
Engineering Graphics	

1	Manual drafting of basic geometric constructions and shapes using set squares, and compass.
2	Understand and draw different projections, apply to create points and lines, in all quadrants.
3	Draw orthographic projections of simple objects like cubes, cylinders, and prisms.
4	Create isometric drawings of simple assemblies.
5	Introduction to CAD System & AutoCAD and understand basic commands.
6	Use AutoCAD to recreate the manually drawn orthographic projections.
7	Create similar drawings using an open-source tool like LibreCAD.
8	Model simple objects (like a nut and bolt) in 3D using AutoCAD or similar software.
9	Draw and assemble a small mechanical device (like a piston or gear assembly) using CAD software
10	Design and draw an entire machine or a significant part of it, incorporating all the skills learned (Mini Project)
Workshop Technology	
1	Demonstrate safe handling and use of various hand tools and power tools.
2	File, saw, and drill a metal piece to create a simple object such as a fitting job.
3	Create a joint or assemble parts using hand tools, ensuring tight fit and proper alignment.
4	Perform simple welding and brazing tasks to join metal pieces.
5	Design and create a shaft on lathe machine using MS Rod.
6	Design and create a flat V Job on Shaper machine using MS Block.
7	Design and manufacture a sheet metal tray, In sheet Metal Shop.
8	Measure, cut, and assemble wooden parts to create a simple structure, such as frame or T-Joint.
9	Ability to program basic CNC machine operations and understand CNC machining processes.
10	Design and 3D print a small part or model using CAD software.

DATA VISUALIZATION USING POWERBI

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: DATA VISUALIZATION USING POWERBI	**Course Code**	**L-T-P**	**Credits**
	SEC037	0-0-4	2

Type of Course:	SEC-1
Pre-requisite(s), if any: Basic knowledge of Excel & data numbers	

Course Perspective. The course on Data Visualization using Power BI provides a comprehensive understanding of how to transform raw data into meaningful insights through visual representation. It begins with the fundamentals of data visualization, emphasizing its importance in decision-making and comparing various tools, with a special focus on Power BI. Students learn to navigate the Power BI interface, connect to diverse data sources, and utilize Power Query for data transformation. The course covers creating and customizing a wide range of visualizations, including charts, maps, and tables, to effectively communicate data insights. The course is divided into 4 modules:

a) Foundation to Data Analytics

b) Data Science Processes

c) Power BI Analytics

d) Introduction to Data Manipulation Using Function

e) Advance Function

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Building** data design using power BI and manage and manipulate data to extract useful information and insights.
CO 2	**Applying** functions to manipulate and analyze data.
CO 3	To **understanding** different data science processes, tools and techniques
CO 4	**Outlining** the key concepts of data handling and how data has evolved

CO 5	**Identifying** the key concepts of data visualization using power BI and will be able to understand the dash board.
CO 6	**Distinguishing** key Data Science concepts such as structured and unstructured data

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Foundation to Data Analytics**	**No. of hours: 8**
Content Summary: Introduction to Data Analytics: Working with Formula and Functions, Introduction to Power BI & Charts, Logical functions using Excel, Analysing Datawith Excel.		
Unit Number: 2	**Title: Data Science Processes**	**No. of hours: 8**
Content Summary: Six steps of data science processes, define research goals, data retrieval,cleansing data, and correct errors as early as possible, integrating – combine data from different sources, transforming data, exploratory data analysis, Data modelling, model and variable selection, model execution, model diagnostic and model comparison, presentation and automation.		
Unit Number: 3	**Title: Power BI Analytics**	**No. of hours: 8**
Content Summary: Power BI Analytics, Data Validation & data models, Power Map for visualize data, Power BI-Business Intelligence , Data Analysis using statistical methods, Dashboard designing.		
Unit Number: 4	**Title: INTRODUCTION TO DATA MANIPULATION USING FUNCTION**	**No. of hours: 8**
Content Summary: Heat Map, Tree Map, Smart Chart,Azure Machine learning , Column Chart, Line Chart , Pie,Bar, Area, Scatter Chart, Data Series, Axes ,Chart Sheet , Trendline , Error Bars, Sparklines, Combination Chart, Gauge, Thermometer Chart.		

Unit Number: 5	Title: Advan Function	No. of hours: 8
Content Summary: Gantt Chart , Pareto Chart etc , Frequency Distribution, Pivot Chart, Slicers ,Tables: Structured, References, Table Styles , What-If Analysis: Data Tables| Correlation model |Regression model.		

Learning Experiences

Classroom Learning Experience

1. Interactive Lectures:
   - Use PowerPoint presentations and live demonstrations to explain key concepts of data visualization and Power BI functionalities.
2. Hands-on Workshops:
   - Conduct practical sessions where students create visualizations, dashboards, and reports using Power BI with provided datasets.
3. Group Discussions:
   - Facilitate discussions on data analytics and visualization techniques, encouraging students to share insights and approaches.
4. Case Study Analysis:
   - Analyze real-world case studies of successful data visualization projects to understand best practices and common pitfalls.
5. Problem-Solving Sessions:
   - Engage students in identifying issues in data analysis and visualization, encouraging them to develop and present solutions.
6. Collaborative Projects:
   - Assign group projects where students work together to analyze a dataset and create a comprehensive Power BI report.
7. In-Class Quizzes:
   - Use short quizzes to reinforce understanding of key concepts and provide immediate feedback on student comprehension.

Outside Classroom Learning

1. Assignments:

- Provide take-home assignments that require students to analyze datasets and create visualizations using Power BI.

2. Online Learning Resources:
   - Offer access to online tutorials, video lectures, and articles that students can explore at their own pace.
3. Self-Directed Projects:
   - Encourage students to choose a dataset of their interest to analyze and visualize, promoting independent learning and creativity.
4. Discussion Forums:
   - Create online platforms (e.g., discussion boards or chat groups) where students can collaborate, ask questions, and share resources.
5. Peer Review Sessions:
   - Set up sessions where students present their work and provide constructive feedback to each other on their visualizations.

Text and Reference Book

1. Microsoft Power BI Complete Reference: Bring Your Data to Life with the Powerful Features of Microsoft Power BI Book by Brian Knight, Devin Knight, and Mitchell Pearson.

SEMESTER: 2

LINEAR ALGEBRA AND ORDINARY

DIFFERENTIAL EQUATIONS

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **Linear Algebra and Ordinary Differential Equations**	**Course Code**	**L-T-P**	**Credits**
	ENMA102	3-1-0	4
Type of Course:	Major-4		

Pre-requisite(s): Single variable calculus, Matrices, Differentiation and Integration

Course Perspective. This course aims to equip engineering students with the fundamental mathematical tools of linear algebra and ordinary differential equations (ODEs) for solving various engineering problems. By the end of the course, students will be able to: Analyze and solve systems of linear equations using matrix operations and numerical techniques, understand eigenvalues, eigenvectors, and their applications in engineering problems, work with vector spaces, linear transformations, and inner product spaces, and solve first-order and second-order ODEs using various analytical and numerical methods. The course is divided into 4 modules:

a) Matrices and Systems of Linear Equations
b) Eigenvalues and Eigenvectors
c) Vector Spaces and Numerical Linear Algebra
d) First-Order and Second-Order Ordinary Differential Equations

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Identifying** the properties of various types of matrices, such as symmetric, skew-symmetric, Hermitian, skew Hermitian, unitary, and orthogonal matrices.
CO 2	**Analyzing** quadratic forms and apply eigenvalues and eigenvectors in practical situations.
CO 3	**Defining** vector spaces, subspaces, linear independence, and basis.
CO 4	**Determining** the dimension of vector spaces and compute row space, column space, and null space of matrices.
CO 5	**Solving** first-order linear, separable, exact, and homogeneous differential equations.

Course Outline:

Unit Number: 1	Title: Matrices and Systems of Linear Equations	No. of hours: 10

Content:

- Introduction to matrices and their operations (addition, subtraction, scalar multiplication, multiplication).
- Types of Matrices (Symmetric, Skew-Symmetric, Hermitian, Skew-Hermitian, Unitary, Orthogonal).
- Introduction to Determinants and their properties.
- Systems of Linear Equations: Homogeneous and non-homogeneous systems.
- Gaussian Elimination and Row Echelon Form for solving systems of linear equations.
- Rank of a matrix and its connection to solvability of systems.

Unit Number: 2	Title: Eigenvalues and Eigenvectors	No. of hours: 10

Content:

- Definition and properties of eigenvalues and eigenvectors.
- Importance of eigenvalues and eigenvectors in engineering problems (e.g., stability

analysis, vibration modes).

- Diagonalization of matrices (when possible).
- Properties of eigenvalues and eigenvectors of special matrices (Symmetric, Skew-Symmetric, Hermitian, Skew-Hermitian, Unitary, Orthogonal).
- Introduction to minimal polynomial and characteristic polynomial.
- Cayley Hamilton theorem

Unit Number: 3	**Title: Vector Spaces**	**No. of hours: 10**

Content:

- Introduction to vector spaces: Definition, axioms, subspaces, spanning sets, linear independence, basis, and dimension.
- Row space, column space, and null space of a matrix.
- Introduction to linear transformations and their representation using matrices.
- Numerical Methods for Linear Algebra:
  - Gaussian elimination with LU decomposition for efficient solution of linear systems.
  - Iterative methods (Jacobi, Gauss-Seidel) for solving large systems.
  - Introduction to condition number and its implications for numerical stability.

Unit Number: 4	**Title: Ordinary Differential Equations**	**No. of hours: 10**

Content:

Introduction to ordinary differential equations (ODEs) and their classification.

First-Order Differential Equations:

- Separable differential equations and methods for solving them.
- Exact differential equations and integrating factors.
- Applications of first-order ODEs in engineering (e.g., population growth, decay models).

Second-Order Linear Differential Equations:

- Homogeneous and non-homogeneous equations.
- Method of undetermined coefficients for solving non-homogeneous equations.
- Variation of parameters for solving non-homogeneous equations.
- Applications of second-order ODEs in engineering (e.g., spring-mass systems,

electrical circuits).

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Use PPTs and visual aids to explain key concepts in linear algebra and differential equations.
2. **Conceptual Understanding**: Cover fundamental topics like matrix operations, eigenvalues, and solutions of ODEs.
3. **Problem-Solving Sessions**: Conduct in-class exercises on systems of equations and differential equations.
4. **Theory Assignments**: Assign theoretical problems with solutions discussed in class.
5. **Group Work**: Collaborate on problem-solving for real-world applications in engineering and science.
6. **Case Studies**: Analyze applications of linear algebra and differential equations in various fields.
7. **Continuous Feedback**: Implement in-class quizzes and feedback sessions to assess understanding.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects applying concepts to practical problems.
2. **Lab Projects**: Facilitate hands-on projects involving software tools for linear algebra and ODEs.
3. **Question Bank**: Provide practice problems and model papers for self-assessment.
4. **Online Forums**: Create platforms for students to discuss and collaborate on problems.
5. **Self-Study for Case Studies**: Encourage independent research on applications of linear algebra and ODEs.
6. **Collaborative Projects**: Organize group projects focused on modelling real-world phenomena using linear algebra and differential equations.

Textbooks

1. Linear Algebra and Its Applications" by David C. Lay, Steven R. Lay, and Judi J. McDonald.
2. "Linear Algebra" by Gilbert Strang.
3. Advanced engineering mathematics: Kreyszig; Wiley. ISBN : 978-81-265-3135-6
4. Advanced engineering mathematics: Peter V. O’Neil Cengage Learning. ISBN : 978-81-315-0310-2
5. "Differential Equations with Boundary-Value Problems" by Dennis G. Zill and Michael R. Cullen.
6. "Ordinary Differential Equations" by Morris Tenenbaum and Harry Pollard.

Reference Books

1. "Matrix Analysis" by Roger A. Horn and Charles R. Johnson.
2. "Numerical Linear Algebra" by Lloyd N. Trefethen and David Bau III.
3. "Theory and Problems of Linear Algebra" (Schaum's Outline) by Seymour Lipschutz and Marc Lipson.
4. "Ordinary Differential Equations and Stability Theory" by David A. Sanchez.

ENGINEERING CHEMISTRY

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: ENGINEERING CHEMISTRY	**Course Code**	**L-T-P**	**Credits**
	ENCH101	4-0-0	4
Type of Course:	Major-5		
Pre-requisite(s), if any: Nil			

Course Perspective. This course introduces students to the fundamental concepts and applications of chemistry in engineering. It is tailored specifically for engineering students to understand the chemical principles underlying various technological processes and materials essential in modern engineering. By exploring topics like water technology, chemical fuels, battery technology, and polymers, the course aims to provide students with a robust foundation in the chemical sciences that directly relates to their future fields of work. The course is divided into 4 modules:

a) Water technology

b) Chemical Fuels

c) Battery Technology

d) Polymer

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the methods for water hardness and alkalinity testing, and the basics of boiler water treatment.
CO 2	**Explaining** the process of dissolved oxygen determination and chemical oxygen demand analysis.
CO 3	**Determining** various methods to enhance the quantity & quality of Fuel.
CO 4	**Identifying** between hard and soft water, solve the related numerical problems on water purification and its significance in industry and daily life.

CO 5	**Articulating** basic concepts of chemistry in daily life.
CO 6	**Designing** efficient process for water analysis and purification

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Water technology**	**No. of hours: 10**

Content Summary:
Introduction to Water Technology: Importance and applications of water in various industries.
Water Analysis: Hardness: Determination by EDTA method, Alkalinity: Determination by double indicator method.
Treatment of Boiler Feed Water
Internal Treatment: Phosphate conditioning, Colloidal conditioning, Calgon conditioning
External Treatment: on exchange process, Lime-soda process, Zeolite process
Determination of Dissolved Gases: Dissolved oxygen: Determination by Winkler's method, Chemical oxygen demand: Determination.
Boiler Scales Formation and Prevention: Formation and ill effects of boiler scales., Methods of prevention of scales.
Numerical Problems: Calculations related to water analysis and treatments.

Unit Number: 2	**Title: Chemical Fuels**	**No. of hours: 10**

Content Summary:
Fuels: Introduction, classification, calorific value (HCV & LCV), Determination of calorific value of fuel using Bomb calorimeter.
Solid fuel: Coal- its analysis by proximate and ultimate analysis, Numerical problems.
Liquid fuels: Refining of petroleum, Petroleum cracking, Reformation of petrol-explanation with reactions, Knocking in IC engine, its ill effects and prevention of knocking. Anti-knocking agent: Leaded and unleaded petrol. Power alcohol and its advantages. Synthetic petrol - Bergius process.
Gaseous fuels: LPG, CNG and their applications.

Unit Number: 3	**Title: Battery Technology**	**No. of hours: 10**

Content Summary:
Introduction to Battery Technology: Galvanic cell, Electrode potential, EMF of the cell, Cell representation.
Batteries and Their Importance: Classification of batteries: Primary, Secondary, and Reserve batteries. Examples of each type.
Battery Characteristics: Voltage, Capacity, Energy density, Power density, Energy efficiency, Cycle life, Shelf life.
Commercial Batteries: Basic requirements for commercial batteries.
Construction, Working, and Applications: Ni-Cd battery, Lithium-ion battery.
Fuel Cells: Differences between batteries and fuel cells. Classification of fuel cells based on: Type of fuel, Electrolyte, Temperature.

Unit Number: 4	Title: Polymer	No. of hours: 10

Content Summary:
Basic Concepts of Polymers: Definition and types of polymers.
Types of Polymers: Thermoplastic polymers, Thermosetting plastics.
Preparation and Applications of Industrially Important Polymers: Natural rubber, Buna S, Buna-N, Neoprene, Isoprene, Nylon-6, Nylon-6,6, Dacron, Terylene.
Advanced Polymers: Conducting polymers, Biodegradable polymers.

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures**: Use PPTs and demonstrations to explain key chemistry concepts relevant to engineering.
2. **Conceptual Understanding**: Cover fundamental topics like thermodynamics, kinetics, and material science.
3. **Problem-Solving Sessions**: Conduct in-class exercises on chemical calculations and reactions.
4. **Theory Assignments**: Assign theoretical problems, with solutions discussed in class.
5. **Group Work**: Collaborate on projects involving chemical processes and materials.
6. **Case Studies**: Analyze real-world applications of chemistry in engineering fields.
7. **Continuous Feedback**: Implement in-class quizzes and feedback sessions to assess understanding.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects applying chemistry concepts to engineering challenges.
2. **Lab Projects**: Facilitate hands-on experiments that explore chemical principles in practical applications.

3. **Question Bank**: Provide practice problems and model papers for self-assessment.
4. **Self-Study for Case Studies**: Encourage independent research on recent advancements in engineering chemistry.
5. **Collaborative Projects**: Organize group projects focused on developing sustainable chemical processes or materials.

Text Books

T1: Principles of Physical Chemistry by B. R. Puri, L. R. Sharma and M. S. Pathania, S. Nagin Chand and Co.

T2: Text book of Physical Chemistry by Soni and Dharmatha, S. Chand & Sons.

T3: Text book of Polymers science by Gowarikar and Vishwanathan.

Reference Books:

R 1. Corrosion Engineering by M. G. Fontana, Mc Graw Hill Publications.

R 2. Engineering Chemistry by Jain and Jain.

Additional Readings:

Basics of electrochemistry:

https://mrcet.com/downloads/digital_notes/HS/4%20ENGINEERING%20CHEMISTRY.pdf

Basics of polymer:

https://gnindia.dronacharya.info/APS/Downloads/SubjectInformation/Chemistry/Unit2/Lecture_1_13022019.pdf

ENGINEERING CHEMISTRY LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: ENGINEERING CHEMISTRY LAB	**Course Code**	**L-T-P**	**Credits**
	ENCH151	0-0-2	1
Type of Course:	Major-7		

Proposed Lab Experiments

Defined Course Outcomes

CO 1	**Applying** various experimental techniques commonly used in chemistry labs, such as titrations, distillations, extractions, chromatography, spectroscopy, and electrochemical methods.
CO 2	**Acquiring** proficiency in handling and operating laboratory equipment, including but not limited to balances, pipettes, burettes, spectrophotometers, pH meters, and other analytical instruments.
CO 3	**Developing** skills in recording and analysing experimental data, including data interpretation of results.
CO 4	**Understanding** hands-on experience in synthesizing various chemical compounds and organic polymers
CO 5	**Illustrating** to write **concise** and accurate laboratory reports, including experimental procedures, observations, results, and conclusions.
CO 6	**Understanding** the ethical responsibilities and laboratory safety protocols associated with conducting experiments.

Ex. No	**Experiment Title**	**Mapped CO/COs**
1	Determination of temporary and permanent hardness in water sample using EDTA.	CO1, CO3, CO5
2	Determination of alkalinity in the given water sample.	CO1, CO3, CO5
3	Determination of viscosity of given liquid.	CO2, CO3, CO5
4	Determination of surface tension of given liquid.	CO2, CO3, CO5
5	Determination of pH by pH-metric titration.	CO1, CO3, CO5
6	Preparation of Phenol-formaldehyde and Urea-	CO4,

	formaldehyde resin	CO5, CO6
7	To determine the iron concentration in the given water sample by Spectrophotometer using potassium thiocyanate as colour developing agent.	CO1, CO3, CO5
8	Determination of chloride content in water sample.	CO1, CO3 CO5, CO6
9	Estimation dissolved oxygen (DO) content in the given water sample by Winkler's method.	CO1, CO3, CO5
10	Determination of iron content in the given solution by Mohr's method.	CO1, CO3, CO5
11	Determination of rate constant of hydrolysis of esters.	CO3, CO5
12	To determine the Iron content in the given salt by using external indicator	CO1, CO3, CO5
13	Determination of wavelength of absorption maximum and colorimetric estimation of Fe^{3+} in solution	CO2, CO3, CO5
14	Determination of molar absorptivity of a compound ($KMnO_4$ or any water-soluble food colorant).	CO2, CO3, CO5
15	Preparation of a nickel complex $[Ni(NH_3)_6]Cl_2$ and estimation of nickel by complexometric titration.	CO4, CO5, CO6
16	Synthesis of drug like Aspirin, /Paracetamol etc.	CO4, CO5, CO6

OBJECT ORIENTED PROGRAMMING USING C++

Program Name	**B. Tech (Computer Science and Engineering)**		
Course Name: Object Oriented Programming using C++	**Course Code**	**L-T-P**	**Credits**

	ENCS102	4-0-0	4
Type of Course:	Major-6		
Contact Hours	40		
Version			
Pre-requisite(s), if any: Fundamental knowledge of computers will be added advantage			

Course Perspective. This course introduces students to the advanced principles and techniques of object-oriented programming (OOP) using C++. It is designed to build upon foundational programming knowledge, particularly for those who have a basic understanding of C programming. The course focuses on teaching students how to think about software development in an object-oriented way, enabling them to design and implement software solutions that are modular, extensible, and maintainable. The course is divided into 4 modules:

a) Foundations of Object-Oriented Programming
b) Classes, Objects, and Advanced Features
c) Inheritance, Polymorphism, and Software Engineering Principles
d) File Handling, Exception Management, and Unit Testing

The Course Outcomes (COs). On completion of the course the participants will be able to:

COs	Statements
CO 1	**Understanding** the procedural and object-oriented paradigm with concepts of streams, classes, functions, data and objects.

CO 2	**Analyzing** dynamic memory management techniques using pointers, constructors, destructors, etc
CO 3	**Applying** the concept of function overloading, operator overloading, virtual functions and polymorphism
CO 4	**Classifying** inheritance with the understanding of early and late binding, usage of exception handling, file handling and generic programming.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Foundations of Object-Oriented Programming**	**No. of hours: 10**
Content Summary: ▪ **Programming Approaches:** Procedure-Oriented Approach vs. Object-Oriented Approach ▪ **Introduction to C++:** Basic syntax and structure of a C++ program, Data Types and Variables, Operators and Expressions, Control Structures, Functions, Arrays and Strings, Pointers ▪ **Basic Concepts of Object-Oriented Programming:** Objects and Classes, Principles of OOP: Abstraction, Encapsulation, Inheritance, Polymorphism, Dynamic Binding and Message Passing ▪ **Characteristics of Object-Oriented Languages:** Benefits and features of OOP languages		
Unit Number: 2	**Title: Classes and Objects**	**No. of hours: 10**

Content Summary: ▪ **Abstract Data Types and Classes:** Concept of abstract data types, Objects and classes, attributes, and methods ▪ **C++ Class Declaration:** Declaring classes in C++, State, identity, and behaviour of objects ▪ **Objects:** Local Objects and Global Objects, Scope resolution operator ▪ **Functions in C++:** Friend Functions, Inline Functions ▪ **Constructors and Destructors:** Instantiation of objects, Types of constructors (default, parameterized, copy), Static Class Data, Array of Objects, Constant member functions and objects ▪ **Memory Management Operators:** New and delete operators for dynamic memory allocation		
Unit Number: 3	**Title: Inheritance and Polymorphism**	**No. of hours: 10**
Content Summary: ▪ **Inheritance:** Types of inheritance (single, multiple, hierarchical, multilevel, hybrid), Access specifiers: public, private, and protected, Abstract Classes, Ambiguity resolution using scope resolution operator and virtual base class ▪ **Advanced Inheritance Concepts:** Aggregation and composition vs. classification hierarchy, Overriding inheritance methods ▪ **Polymorphism:** Types of Polymorphism (compile-time and run-time), Function Overloading, Operator Overloading ▪ **Pointers and Virtual Functions:** Pointer to objects, this pointer, Virtual Functions and pure virtual functions		
Unit Number: 4	**Title: Advanced C++ Features**	**No. of hours: 10**

Content Summary:

- **Strings and Streams:** Manipulating strings, Streams and file handling, File streams and string streams
- **Operators and Error Handling:** Overloading operators, Error handling during file operations, Formatted I/O
- **Generic Programming:** Function templates, Class templates
- **Exception Handling:** Throwing an exception, The try block, Catching an exception, Exception objects, Exception specifications, Rethrowing an exception, Catching all exceptions

Learning Experience

Classroom Learning Experience

1. **Interactive Lectures**: Use PPTs and live coding demonstrations to explain key OOP concepts.
2. **Conceptual Understanding**: Cover fundamental topics like classes, objects, inheritance, and polymorphism.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on implementing OOP principles in C++.
4. **Theory Assignments**: Assign programming problems that reinforce OOP concepts, discussed in class.
5. **Group Work**: Collaborate on projects that require designing and implementing class structures.
6. **Case Studies**: Analyze real-world applications of OOP in software development.
7. **Continuous Feedback**: Implement in-class quizzes and code reviews for ongoing assessment.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects emphasizing OOP design principles in C++.
2. **Lab Projects**: Facilitate hands-on programming tasks that apply OOP concepts to real-world scenarios.
3. **Question Bank**: Provide practice problems and resources for self-assessment.

4. **Online Forums**: Create platforms for students to discuss coding challenges and share solutions.
5. **Self-Study for Case Studies**: Encourage independent research on OOP best practices and design patterns.
6. **Collaborative Projects**: Organize group projects focused on developing software applications using OOP in C++.

Text books:

T1: Robert Lafore, "Object-Oriented Programming in C++", Sams Publishing, 4th Edition, 2004.

T2: E. Balagurusamy, "Object-Oriented Programming with C++", McGraw Hill Education, 6th Edition, 2017**.**

Reference Book

1. Schildt Herbert, "C++: The Complete Reference", Wiley DreamTech, 2005.Parasons, "Object Oriented Programming with C++", BPB Publication, 1999.
2. Steven C. Lawlor, "The Art of Programming Computer Science with C++", Vikas Publication, 2002.
3. Yashwant Kanethkar, "Object Oriented Programming using C++", BPB, 2004

Additional Readings:

Online Learning

R 1. C++ Documentation on cppreference.com

- A comprehensive reference that includes detailed documentation of C++ syntax, library functions, and features organized by version.
- **Link: cppreference.com**

R 2. LearnCpp.com

- A free website that teaches the basics and subtleties of C++ programming. It covers everything from basic syntax to advanced features.
- **Link: LearnCpp.com**
- **Link: Pluralsight C++ Path**

R 3. GitHub

- **Explore and contribute to open-source C++ projects, which can provide practical experience and exposure to real-world coding practices and collaboration.**
- **Link: [GitHub]**

OBJECT ORIENTED PROGRAMMING USING C++ LAB

Program Name	**B. Tech (Computer Science and Engineering)**		
Course Name: Object Oriented Programming using C++ Lab	**Course Code**	**L-T-P**	**Credits**
	ENCS152	0-0-2	1
Type of Course:	Major-8		

Pre-requisite(s), if any: Basics of C programming

Defined Course Outcomes

COs	Statements
CO 1	**Understanding** class object concepts by using C++.
CO 2	**Developing** programs using inheritance and polymorphism.
CO 3	**Demonstrating** the significance of constructors and destructor.
CO 4	**Illustrating** generic classes using template concepts.
CO 5	**Implementing** the concept of file handling.

Lab Experiments

Defined Course Outcomes

((	**Lab tasks**

Implement a simple calculator in C++ that can perform basic arithmetic operations such as addition, subtraction, multiplication, and division. The program should prompt the user to enter two numbers and an operator, then display the result. Use appropriate data types and control structures to handle the calculations and validate user inputs.

Create a C++ program that checks if a given number is a prime number. The program should prompt the user to enter a number, then use control structures and functions to determine if the number is prime. Display an appropriate message indicating the result.

Implement a C++ program that sorts an array of integers using the bubble sort algorithm. The program should allow the user to input the array elements, then use a function to sort the array in ascending order. Display the sorted array as the output.

Write a C++ program to demonstrate pointer arithmetic by creating an array of integers and using pointers to traverse and manipulate the array elements. Implement functions to calculate the sum, average, and maximum value of the array using pointer arithmetic.

Write a C++ program to perform basic matrix operations such as addition, subtraction, and multiplication. Use two-dimensional arrays to represent the matrices and implement functions for each operation. Ensure the program handles matrices of appropriate sizes and displays the results accurately.

Create a C++ program that generates the Fibonacci sequence up to a specified number of terms. Use a loop and control structures to generate the sequence and store the terms in an array. Display the generated sequence as the output.

Write a C++ program to evaluate a string expression containing numbers, arithmetic operators (+, -, *, /), and parentheses. Implement a function that parses the expression and computes the result, considering operator precedence and parentheses.

Write a C++ program to find all unique palindromic substrings in a given string. The function should take a string as input and return a set of strings containing all unique palindromic substrings.

Create a class Rational to represent rational numbers with attributes numerator and denominator, implementing default, parameterized, and copy constructors, methods to add, subtract, multiply, and divide rational numbers, overloading the << and >> operators for input and output, and a friend function to compare two rational numbers.

Create a class Matrix that represents a 2D matrix with dynamic memory allocation, implementing default, parameterized constructors, and a destructor, methods to add, subtract, and multiply matrices, overloading the [] operator to access matrix elements, and inline functions for basic matrix operations.

Create a class Student with attributes studentID, name, and grades (an array of integers), implementing default, parameterized constructors, and a destructor, methods to calculate the average grade and display student details, using constant member functions to display details, and implementing dynamic memory allocation for the grades array.

Create an abstract class Shape with a pure virtual function calculateArea(), deriving classes Circle, Rectangle, and Triangle each with attributes relevant to their shapes, implementing default and parameterized constructors, methods to calculate and display the area of each shape, and an array of Shape pointers to store different shapes and calculate their areas.

Create a class InventoryItem with attributes itemID, itemName, and quantity, implementing default, parameterized, and copy constructors, methods to add, remove, and display inventory items, overloading the ++ and -- operators to increase and decrease item quantity, and implementing dynamic memory allocation for inventory items.

Create a class Polynomial to represent a polynomial with dynamic memory allocation for coefficients, implementing default, parameterized constructors, and a destructor, methods to add, subtract, and multiply polynomials, overloading the +, -, and * operators for polynomial operations, and friend functions to input and output polynomials.

Develop a Vehicle Management System that demonstrates different types of

C 3	inheritance and polymorphism in C++. The system should manage various types of vehicles, including cars, trucks, and motorcycles, and should be able to perform operations such as adding new vehicles, displaying vehicle details, and comparing vehicles.
C C 3	Create a base class Account with methods deposit() and withdraw(). Derive classes SavingsAccount and CurrentAccount from Account. Overload the deposit() and withdraw() methods in the derived classes to include additional parameters like interest rate for SavingsAccount and overdraft limit for CurrentAccount.
C C 3	Create a class ComplexNumber to represent complex numbers. Implement operator overloading for +, -, *, and / operators to perform arithmetic operations on complex numbers. Use inheritance to extend the class with additional functionality for polar representation.
C C 3	Create a base class Animal with a virtual function makeSound(). Derive classes Dog and Cat from Animal, each implementing makeSound(). Write a function playWithAnimal() that takes a pointer to Animal and calls makeSound(). Demonstrate polymorphism by calling playWithAnimal() with pointers to Dog and Cat.
C C 3	Create a base class Person with attributes name and age. Derive classes Student and Teacher from Person. Further derive a class TeachingAssistant from both Student and Teacher. Use a virtual base class to avoid ambiguity in accessing attributes of Person.
C C 3	Create a base class Vehicle with attributes make and model, and methods start() and stop(). Derive classes Car, Truck, and Motorcycle from Vehicle. Use dynamic memory allocation (new and delete operators) to create and manage objects of these classes. Implement a function to display details of all vehicles.
C C 4	Write a C++ program that compresses a string using the counts of repeated characters. For example, the string "aabcccccaaa" would become "a2b1c5a3". If the "compressed" string would not become smaller than the original string, the function should return the original string. Use streams for efficient string manipulation.

Write a template-based function in C++ to sort an array of any data type using the quicksort algorithm. Ensure the function works with different data types such as integers, floating-point numbers, and strings.

Create a custom exception class InvalidInputException in C++ to handle invalid inputs. Implement a function that takes user input and throws an InvalidInputException if the input is not valid. Use try, catch, and throw blocks to handle the exception and display an appropriate error message.

Write a C++ program that reads a text file, processes the text to remove punctuation, convert to lowercase, and count the frequency of each word. Use string streams for text manipulation and file streams for reading and writing files.

Implement a template-based stack class in C++ that supports basic stack operations such as push, pop, top, and isEmpty. Ensure the class works with different data types and includes appropriate exception handling for stack underflow and overflow.

Write a C++ program that reads data from a file and processes it. Implement error handling to catch exceptions if the file does not exist, is empty, or cannot be read. Use exception specifications to define the exceptions that the functions might throw.

OVERVIEW OF AI, DATA SCIENCE, ETHICS AND FOUNDATION OF DATA ANALYSIS LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **OVERVIEW OF AI, DATA SCIENCE, ETHICS AND FOUNDATION OF DATA ANALYSIS LAB**	**Course Code**	**L-T-P**	**Credits**
	ENSP152	0-0-4	2
Type of Course:	IDC-3		

Pre-requisite(s): Basic knowledge of Excel

Course Perspective. This course aims to equip engineering students with the Introduction to Data Science, Natural Language, Machine generated Data, Graph based or Network Data, Audio, Image, Video, Streaming data. Also, Six steps of data science processes define research goals, data retrieval, cleansing data, and correct errors as early as possible, integrating – combine data from different sources, transforming data, exploratory data analysis, Data modelling, model and variable selection, presentation and automation would be taught to the students. Introduction to machine Learning and Introduction to Data Analytics are also included in the syllabus. The course is divided into 5 modules:

a) Introduction to Data Science

b) Introduction to Data Science Processes

c) Introduction to Machine Learning

d) Introduction to AI

e) Introduction to Data Analytics

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Outlining** the key concepts of AI and how AI has evolved
CO 2	**Identifying** the key concepts of Machine Learning and will be able to differentiate between key algorithms such as supervised learning and unsupervised learning
CO 3	**Distinguishing** key Data Science concepts such as structured and unstructured data, SQL and NoSQL Database
CO 4	**Examining** the process required the successfully execute a Machine Learning or Data Science project
CO 5	**Infering** the large scale data using Excel

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number : 1	Title: Introduction to Data Science	No. of hours: 10

Content:
Defining Data Science and Big Data, Benefits and Uses of Data Science and Big Data, Facets of Data, Structured Data, Unstructured Data, Natural Language, Machine generated Data, Graph based or Network Data, Audio, Image, Video, Streaming data, Data Science. Process, Big data ecosystem and data science, distributed file systems, Distributed programming framework, data integration framework, machine learning framework, No SQL Databases, scheduling tools, benchmarking tools, system deployments

Unit Number : 2	Title: Data Science Processes	No. of hours: 10

Content:
Six steps of data science processes define research goals, data retrieval, cleansing data, and correct errors as early as possible, integrating – combine data from different sources, transforming data, exploratory data analysis, Data modelling, model and variable selection, model execution, model diagnostic and model comparison, presentation and automation

Unit Number : 3	Title: Introduction to Machine Learning	No. of hours: 10

Content:
Data for Machine Learning, Leveraging Machine Learning, Descriptive vs Predictive Analytics, Machine Learning and Statistics, Artificial Intelligence and Machine Learning, Types of Machine Learning – Supervised, Unsupervised, Semi-supervised, Reinforcement Learning, Types of Machine Learning Algorithms, Classification vs Regression Problem, Bayesian, Clustering, Decision Tree, Dimensionality Reduction, Neural Network and Deep Learning, Training machine learning systems.

Unit Number : 4	Title: Introduction to AI	No. of hours: 10

Content:
What is AI, Turing test, cognitive modelling approach, law of thoughts, the relational agent approach, the underlying assumptions about intelligence, techniques required to solve AI problems, level of details required to model human intelligence, successfully building an intelligent problem, history of AI.

Unit Number: 5	Title: Introduction to Data Analytics	No. of hours: 4

Content Summary:
Working with Formula and Functions, Introduction to Power BI & Charts, Logical functions using Excel, Analysing Data with Excel.

Learning Experience

Classroom Learning Experience

1. Interactive Lectures:
   - Utilize presentations and demonstrations to explain foundational concepts of AI, data science, machine learning, and data analytics.
2. Hands-on Workshops:
   - Conduct practical sessions where students use tools like Excel, Power BI, and programming languages (e.g., Python) to analyze data and create visualizations.
3. Group Discussions:
   - Facilitate discussions on ethical implications of AI and data science, encouraging students to consider real-world scenarios.
4. Case Study Analysis:
   - Analyze case studies of successful AI and data science projects, focusing on the methodologies used and lessons learned.
5. Collaborative Projects:
   - Assign group projects where students work together to define a data science problem, gather data, analyze it, and present their findings.
6. Quizzes and Knowledge Checks:
   - Use short quizzes to assess understanding of key concepts, providing immediate feedback to students.
7. Simulation Exercises:
   - Engage students in simulations of real-world data analysis scenarios, where they must apply the steps of the data science process.

Outside Classroom Learning

1. Research Assignments:
   - Assign research projects that require students to explore current advancements and ethical issues in AI and data science.
2. Online Learning Modules:

- Provide access to MOOCs or online resources for students to deepen their understanding of machine learning algorithms and data analytics techniques.

3. Self-Directed Projects:
   - Encourage students to select their datasets to analyze independently, applying the concepts learned in class.
4. Discussion Forums:
   - Create online forums for students to discuss topics related to AI and data science, share resources, and collaborate on ideas.

Textbooks

1. Artificial Intelligence 3e: A Modern Approach Paperback – By Stuart JRussell & Peter Norvig; Publisher – Pearson
2. Artificial Intelligence Third Edition By Kevin Knight, Elaine Rich, B. Nair –McGrawHill

Reference Books

1. Artificial Intelligence Third Edition By Patrick Henry Winston – Addison-Wesley Publishing Company

Proposed Lab Experiments

Experiment Title	Mapped CO/ COs
Write a program that uses functions to perform the following operations on singly linked list: i.Creation ii.Insertion iii.Deletion iv.Traversal	CO1, CO2
Write a program that uses functions to perform the following operations on doubly linked list: i.Creation ii.Insertion iii.Deletion iv.Traversal	CO1, CO2
Write a program that uses functions to performthe following operations on circular linked list:	CO1,

i.Creation ii.Insertion iii.Deletion iv.Traversal	CO2
Write a program that implement stack and itsoperations using: i.Arrays ii.Pointers	CO1, CO2
Write a program that implement queue and itsoperations using: i.Arrays ii.Pointers	CO1, CO2
Write a program that implements the following sorting methods to sort a given list of integers inascending order: i. Bubble sort Selection sortiii.Insertion sort	CO1, CO2
Write a program that use both recursive and non-recursive functions to perform the followingsearching operations for a Key value in a given list of integers: i.Linear searchii.Binary search	CO1, CO2
Write a program to implement the tree traversalmethods.	CO1, CO2
Write a program to implement the graphtraversal methods.	CO1, CO2
Program on Comparative Analysis of MatchingAlgorithms	CO3, CO4
Analyzing the Impact of COVID-19 using Data Science: A Comprehensive Case Study	CO3, CO4
Program for Enhancing Data Visualization with Conditional Formatting	CO3, CO4
Exploring Pivot Tables in Data Science	CO3, CO4
Data Visualization with Power Map	CO3, CO4
Write a program for Data Science with Power BI	CO3, CO4
Write a program for Building Predictive Models indata science	CO3

	, CO4
Analyzing Sales Wallet Transactions using Data Science: Extracting Insights and Driving BusinessGrowth	CO3 , CO4
Harnessing the Power of Power Query in Data Science: Extract, Transform, and Analyze Data with Efficiency and Precision	CO3 , CO4
"Exploring Correlation Methods in Data Science:Unveiling Relationships and Patterns in Complex Datasets	CO3 , CO4

Applied Generative AI: Practical Tools and Techniques

Program Name	**B. Tech (Computer Science and Engineering)**		
Course Name: Applied Generative AI: Practical Tools and Techniques for the Modern Professional	**Course Code**	**L-T-P**	**Credits**
		0-0-	2

4

Type of Course: SEC-2

Defined Course Outcomes

CO	Statements
1	Applying basic functionalities of Hugging Face and LangChain to generate text-based applications and automate simple tasks
2	Analyzing ethical dilemmas and create basic data visualizations using GenAI tools, automating standard business communications and financial predictions.
3	Creating roleplaying chatbots, automate market insights extraction, and generate social media content using GenAI.
4	Evaluating advanced GenAI models for automating complex tasks, generating interactive visualizations, and conducting ethical analyses.

Proposed Lab Experiments

Experiment Title	Mapped CO/COs
Experiment 1: Introduction to Generative AI: Overview of Generative AI, Hugging Face, and LangChain. a. Explore basic functionalities of Hugging Face and LangChain by creating a simple text generation application. b. Learn to create simple prompts for GenAI models to generate various types of text outputs. c. Automate tasks such as scheduling and data entry using GenAI. d. Perform basic data analysis and generate summary reports with GenAI. e. Generate automated content such as emails and reports using GenAI.	
Experiment 2: Ethical Considerations and Data Visualization	**CO2**

a. Create a presentation or report outlining ethical issues and potential solutions.
b. Develop scripts for generating data visualizations like bar charts and pie charts using GenAI.
c. Automate business communications such as appointment reminders
d. Use GenAI for financial predictions based on historical data
e. Plan and manage tasks using GenAI for project scheduling

Experiment 3: Advanced GenAI Applications and Customer Interaction **CO3**

a. Create a chatbot to handle basic customer queries.
b. Automate market insights extraction with GenAI.
c. Generate presentations and reports using GenAI.
d. Develop roleplaying chatbots for customer service training.
e. Generate social media posts and content using GenAI.

Complex Applications and Ethical Frameworks **CO4**

Experiment 4: Explore Advanced GenAI Models and Their Applications in Various Industries

Explore Advanced GenAI Models and Their Applications in Various Industries. Explore advanced Generative AI models such as GPT-4, DALL-E, and BERT. Develop a comprehensive report or presentation detailing these models and their potential uses in various industries, including healthcare, finance, marketing, and customer service. Example models like GPT-4 (OpenAI), DALL-E (OpenAI), BERT (Google), T5 (Google), and CLIP (OpenAI) will be covered. The outcome will be a thorough understanding of how these models can be applied to natural language processing, image generation, conversational agents, and automated content creation.

Project 1: Intelligent Email Assistant CO1,CO2

Problem Statement: Develop an intelligent email assistant that uses Hugging Face and LangChain to draft, respond to, and organize emails. This project aims to streamline email management for professionals by leveraging generative AI tools.	, CO3,CO4
Project 2: Social Media Content Generator	
Problem Statement: Design a social media content generator that uses generative AI models to create posts, captions, and hashtags for different platforms. This project will help social media managers generate engaging content efficiently.	CO1,CO2 , CO3,CO4
Project 3: Ethical AI Implementation Framework for Healthcare	
Problem Statement: Develop a comprehensive ethical AI implementation framework for healthcare organizations to ensure the responsible use of generative AI in medical applications. This project addresses the ethical challenges and ensures that AI is used in a fair, transparent, and accountable manner.	CO1,CO2 , CO3,CO4
Project 4: Financial Report Generation System	
Problem Statement: Create a financial report generation system that uses generative AI models to analyze financial data and generate comprehensive reports. This project will assist financial analysts in making informed decisions based on accurate and data-driven insights.	CO1,CO2 , CO3,CO4

Learning Experiences

- Interactive Learning: Utilize lecture PPTs, video lectures, and interactive teaching boards to engage with fundamental concepts and practical applications in real-time.
- Hands-On Practice: Participate in project-based lab assignments and problem-based theory assignments to apply theoretical knowledge to practical scenarios, enhancing understanding through hands-on experience.

- Continuous Assessment: Engage in continuous assessment through quizzes, assignments, and projects to track progress and receive timely feedback, allowing for iterative improvement.
- Collaborative Projects: Work on collaborative group projects and case studies, promoting teamwork and peer-to-peer learning while tackling complex problems.
- ICT Integration: Leverage Moodle LMS for accessing course materials, submitting assignments, and receiving feedback, enhancing the learning experience through technology integration.
- Support & Feedback: Benefit from regular support and feedback from the course instructor, available for additional help and clarification, ensuring personalized learning support.
- Practical Applications: Develop skills through real-world projects and applications, such as creating text-based applications and ethical AI frameworks, preparing students for practical and industry-related challenges.

Text Books:

- "Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play" by David Foster
- "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Aurélien Géron

Online References

General Introduction to Generative AI

1. **OpenAI Blog**: Articles and research updates on advancements in AI - OpenAI Blog
2. **DeepMind Publications**: Research and analysis on the latest AI technologies - DeepMind Research

Tools and Libraries

1. **Hugging Face Documentation**: Comprehensive guide and API references for using transformer models - Hugging Face Docs
2. **LangChain Documentation**: Tools and libraries for building language applications - LangChain GitHub

Ethical Frameworks for AI

1. **AI Ethics Guidelines by the European Commission**: Framework for trustworthy AI - Ethics Guidelines for Trustworthy AI
2. **Partnership on AI**: Research and partnership initiatives on AI ethics - Partnership on AI

Prompt Engineering and Usage

3. **Practical Prompt Engineering Guide by OpenAI**: Guidelines on effective prompt engineering - Prompt Engineering with OpenAI
4. **Prompt Engineering Workshop**: Online courses and tutorials on prompt engineering - Prompt Engineering Course

Minor Project-I

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
COURSE NAME: Minor Project-I	**COURSE CODE**	**L-T-P**	**CREDITS**
	ENSI152	0-0-0	2
TYPE OF COURSE:	Proj-1		
PRE-REQUISITE(S), IF ANY: NA			

Course Perspective:

The objective of Minor Project-I for the B. Tech (Computer Science and Engineering) program is to provide students with the opportunity to apply theoretical knowledge to real-world societal problems. This course aims to develop students' ability to identify and understand complex societal issues relevant to computer science, engage in critical thinking to formulate and analyze problems, and conduct comprehensive literature reviews to evaluate existing solutions. Through this project, students will enhance their research skills, document their findings in a well-structured manner, and effectively present their analysis and conclusions. The course fosters professional development by encouraging students to approach problems from multiple perspectives, develop innovative solutions, and improve their communication and documentation skills. Ultimately, the Minor Project-I course seeks to prepare students for future professional challenges by integrating academic knowledge with practical problem-solving experiences.

Duration: 6 weeks.

Project must focus on following aspects:

Project Requirements:

1. **Understanding of Societal Problems:**

- Students must have a basic understanding of societal problems, the concerned domain, and relevant issues.

2. **Critical Thinking and Problem Formulation:**
   - Students are expected to think critically about formulated problems and review existing solutions.
3. **Presentation of Findings:**
   - Students must be able to present findings from existing solutions in an appropriate format.
4. **Implementation:**
   - Students are not strictly expected to provide or implement these existing solutions.

Guidelines:

1. **Project Selection:**
   - Choose a societal problem relevant to the field of computer science and engineering.
   - Ensure the problem is specific and well-defined.
2. **Literature Review:**
   - Conduct a thorough review of existing literature and solutions related to the problem.
   - Identify gaps in existing solutions and potential areas for further investigation.
3. **Analysis and Critical Thinking:**
   - Analyze the problem critically, considering various perspectives and implications.
   - Evaluate the effectiveness and limitations of current solutions.
4. **Documentation:**
   - Document the entire process, including problem identification, literature review, analysis, and findings.
   - Use appropriate formats and standards for documentation.
5. **Presentation:**
   - Prepare a presentation summarizing the problem, existing solutions, analysis, and findings.
   - Ensure the presentation is clear, concise, and well-structured.

Evaluation Criteria for Minor Project (Out of 100 Marks):

1. **Understanding of Societal Problems (20 Marks):**
   - Comprehensive understanding of the problem: 20 marks
   - Good understanding of the problem: 15 marks
   - Basic understanding of the problem: 10 marks
   - Poor understanding of the problem: 5 marks
   - No understanding of the problem: 0 marks
2. **Critical Thinking and Analysis (30 Marks):**
   - Exceptional critical thinking and analysis: 30 marks
   - Good critical thinking and analysis: 25 marks
   - Moderate critical thinking and analysis: 20 marks
   - Basic critical thinking and analysis: 10 marks
   - Poor critical thinking and analysis: 5 marks
   - No critical thinking and analysis: 0 marks
3. **Literature Review (20 Marks):**
   - Comprehensive and detailed literature review: 20 marks
   - Good literature review: 15 marks
   - Moderate literature review: 10 marks
   - Basic literature review: 5 marks
   - Poor literature review: 0 marks
4. **Documentation Quality (15 Marks):**
   - Well-structured and detailed documentation: 15 marks
   - Moderately structured documentation: 10 marks
   - Poorly structured documentation: 5 marks
   - No documentation: 0 marks
5. **Presentation (15 Marks):**
   - Clear, concise, and engaging presentation: 15 marks
   - Clear but less engaging presentation: 10 marks
   - Somewhat clear and engaging presentation: 5 marks
   - Unclear and disengaging presentation: 0 marks

Total: 100 Marks

Course Outcomes:

By the end of this course, students will be able to:

1. **Understand Societal Issues:**
   - Demonstrate a basic understanding of societal problems and relevant issues within the concerned domain.
2. **Critical Thinking:**
   - Think critically about formulated problems and existing solutions.
3. **Literature Review:**
   - Conduct comprehensive literature reviews and identify gaps in existing solutions.
4. **Documentation:**
   - Document findings and analysis in a well-structured and appropriate format.
5. **Presentation Skills:**
   - Present findings and analysis effectively, using clear and concise communication skills.
6. **Problem Analysis:**
   - Analyze problems from various perspectives and evaluate the effectiveness of existing solutions.
7. **Professional Development:**
   - Develop skills in research, analysis, documentation, and presentation, contributing to overall professional growth.

Learning Experiences

- Real-World Application: Students will apply theoretical knowledge to analyze societal problems, gaining hands-on experience in tackling real-world issues related to computer science.
- Critical Thinking and Problem Solving: By identifying, formulating, and evaluating complex problems, students will enhance their critical thinking and analytical skills.
- Research Skills: Students will conduct comprehensive literature reviews, learning to assess existing solutions and identify research gaps for future exploration.

- Effective Communication: Through structured documentation and presentations, students will develop clear and concise communication skills essential for professional settings.
- Multi-Perspective Analysis: Students will learn to evaluate problems from diverse perspectives, fostering innovative thinking and problem-solving abilities.
- Professional Development: The project encourages research, analysis, and presentation skills, preparing students for future professional challenges in the tech industry.

SEMESTER: III

JAVA PROGRAMMING

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
COURSE NAME: JAVA PROGRAMMING	**COURSE CODE**	**L-T-P**	**CREDITS**
	ENCS201	4-0-0	4
TYPE OF COURSE:	Major-9		

PRE-REQUISITE(S), IF ANY: C PROGRAMMING

Course Perspective. This course provides a comprehensive introduction to Java, one of the most popular and widely used programming languages in the world, particularly known for its portability across platforms from mainframe data centers to smartphones. The "Java Programming" course is meticulously designed to introduce students to the core concepts of object-oriented programming using Java, covering everything from basic constructs to advanced programming features. The curriculum is structured to not only impart theoretical knowledge but also to enhance practical skills through extensive lab sessions, thereby preparing students for real-world software development. The course is divided into 4 modules:

a) Introduction to Java and OOP
b) Inheritance and Polymorphism (Abstract Class, Packages, and Interfaces)
c) Exception Handling, Multithreading and Wrapper Class
d) I/O Stream, File Handling, and Collections

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Applying** Java fundamentals and basic constructs to write Java programs.
CO 2	**Designing** object-oriented solutions using classes, objects, inheritance, and polymorphism.
CO 3	**Utilizing** interfaces and packages for code structure and reusability.
CO 4	**Implementing** error handling with try-catch-finally and custom exceptions.
CO 5	**Designing** multithreaded applications using synchronization.
CO 6	**Performing** file I/O, work with Java Collections Framework, and manipulate data using collections

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Introduction to Java and OOP**	**No. of hours: 10**

Content:

Introduction to Java –

Features, and Importance, Java Virtual Machine, Byte Code; Keywords, constants, variables and Data Types, Operators and Expressions, Type casting and conversion;

Java Control Structure - Decision making – if, if-else, if-else-if ladder, nested if, switch-case, Loop – do, while, for, jump statements – break and continue;

Simple Input and Output - Scanner Class; Arrays Handling - Single and Multi-dimensional, Referencing Arrays Dynamically;

Java Strings: String class, Creating & Using String Objects, Manipulating Strings, String Immutability & Equality, Passing Strings To & From Methods.

OOP Paradigm: Features of OOP, Class and Object in Java: Creating Classes and Objects. Defining Data Members and Member Methods, Overloading Member Methods, Static Members, this Keyword. Constructors: default, parameterized and copy constructors.

Unit Number: 2	**Title: Inheritance and Polymorphism (Abstract Class, Packages, and Interfaces)**	**No. of hours: 10**

Content:

Access Specifies, Introduction to Inheritance – Derived Class and Super class, super Keyword;

Types of inheritance – simple, multilevel, multilevel, hierarchical, and hybrid;

Polymorphism – Static (Method overloading), Dynamic (Method Overriding);

Final Class and Method, finalize keyword, Garbage Collection;

Abstract Method and Abstract Class. Interfaces - Defining an Interface, Implementing an Interface;

Packages - Creating Package, Naming a Package, Using Package Members, Extending Interfaces and Packages, Package and Class Visibility.

Unit	**Title: Exception Handling,**	**No. of hours:**

Number: 3	**Multithreading and Wrapper Class**	**10**

Content:

Exception Handling - Definition, Dealing with Errors, The Classification of Exceptions, Declaring Checked Exceptions, Throw an Exception, Creating Exception Classes, Catching Exceptions, finally clause;

Multithreaded Programming - Fundamentals, Java thread model: priorities, synchronization, messaging, thread classes, Runnable interface, inter thread Communication, suspending, resuming, and stopping threads.

Wrapper Classes - Autoboxing/Unboxing, Enumerations.

Unit Number: 4	**Title: I/O Stream, File Handling, and Collections**	**No. of hours: 10**

Content:

File Handling: File Class Methods, Reading from a File, Writing to a File, Buffered I/O, Character Streams, Byte Streams, File Input/Output Stream, FileReader, FileWriter, BufferedWriter, BufferedReader, FileInputStream, FileOutputStream, File Navigation, File Permissions, Directory Operations, File and Directory Attributes

Java Collections Framework: Introduction to Java Collections Framework

Collection Interfaces: List (ArrayList, LinkedList, Vector), Set (HashSet, LinkedHashSet, TreeSet), Queue (PriorityQueue), Map (HashMap, LinkedHashMap, TreeMap), Iterators, Comparable and Comparator Interfaces, Sorting Collections, Generics in Collections

Working with Collections: Adding, Removing, Searching Elements, Iterating Elements

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key Java concepts using PPTs and live coding demonstrations.
2. **Conceptual Understanding**: Cover topics like object-oriented programming, exception handling, and Java APIs.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on coding challenges and algorithm implementation.
4. **Theory Assignments**: Assign programming problems that reinforce Java concepts, discussed in class.

5. **Group Work**: Collaborate on projects requiring teamwork in Java application development.
6. **Case Studies**: Analyze real-world applications of Java in software development and enterprise solutions.
7. **Continuous Feedback**: Implement quizzes and code reviews to provide ongoing assessment and improvement.
8.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects applying Java programming concepts to practical problems.
2. **Lab Projects**: Facilitate hands-on programming tasks that apply Java in real-world scenarios.
3. **Question Bank**: Provide practice problems and resources for self-assessment in Java.
4. **Online Forums**: Create platforms for discussing Java programming challenges and solutions.
5. **Self-Study for Case Studies**: Encourage independent research on Java best practices and frameworks.
6. **Collaborative Projects**: Organize group projects focused on developing Java applications and systems.

Text Book

1. Herbert Schildt, ―java – the complete reference‖, oracle press.
2. Cay s. Horstmann, ―core java volume – i fundamentals‖, pearson.

Additional Readings:

Online Learning Resources

1. **Oracle Java Tutorials**
   - The official tutorials from Oracle, which owns Java, are a great starting point. These cover the basics and advanced features of Java.
   - Link: Oracle Java Tutorials
2. **Java Code Geeks**
   - A community-driven site that offers free Java tutorials, articles, and examples. It's a valuable resource for practical tips and best practices.
   - Link: Java Code Geeks
3. **LeetCode**
   - Excellent for practicing Java coding problems, LeetCode helps in enhancing problem-solving skills in Java, which is crucial for technical interviews.

- Link: LeetCode

JAVA PROGRAMMING LAB

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
COURSE NAME:	**COURSE CODE**	**L-T-P**	**CREDITS**
JAVA PROGRAMMING LAB	**ENCS251**	0-0-2	1
TYPE OF COURSE:	Major-13		
Contact Hours			
Version			

PRE-REQUISITE(S), IF ANY: basic working knowledge of C++ programming will be an added advantage

PROPOSED LAB EXPERIMENTS

DEFINED COURSE OUTCOMES

COS	
CO1	**Demonstrating** the use of primitive data types, type casting, and basic input/output operations in Java.
CO2	**Implementing** control structures such as conditional statements and loops to perform arithmetic operations and generate sequences.
CO3	**Creating** and manipulating arrays and demonstrate basic inheritance, polymorphism, and class hierarchies in Java applications.
CO4	**Developing** and test advanced Java applications using multithreading, file handling, collections, and exception handling to solve real-world problems.

Lab Task	MAPPED CO/COS
A Java Application that manages a small library system	CO1

A Java Application to manage a simple bank account system	**CO1**
A Java class hierarchy for a simple educational institution system	**CO2**
A system for managing different types of vehicles in a rental service	**CO2**
A Java Application for a basic shape drawing application	**CO3**
A Java multithreaded application that simulates a banking system	**CO3**
A file management system that supports operations such as reading,writing, copying, and navigating files and directories	**CO4**
A contact management system that utilizes different data structures like Lists, Sets, Queues, and Maps	**CO4**

DISCRETE MATHEMATICS

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Discrete Mathematics	**Course Code**	**L-T-P**	**Credits**
	ENCS203	3-1-0	4
Type of Course:	Major -9		

Pre-requisite(s), if any: Basic of Mathematics

Course Perspective. The Discrete Mathematics course offers a foundational exploration into key mathematical concepts essential for computer science and engineering. Covering topics such as set theory, logic, relations, functions, combinatorics, recurrence relations, and graph theory, the course equips students with critical problem-solving skills. By emphasizing both theoretical understanding and practical application, students learn to tackle complex problems in algorithm design, data analysis, and system modeling. Through this comprehensive curriculum, students gain a robust grasp of discrete structures, which are pivotal in computational fields, enhancing their analytical capabilities and preparing them for advanced studies and professional challenges.

The Course Outcomes (COs).

COs	Statements
CO 1	**Applying** set theory concepts, analyze logical expressions, and use mathematical induction in proofs**.**
CO 2	**Understanding** Model relations, understand graph theory basics, and solve problems related to graphs**.**
CO 3	**Analyzing** Count combinatorial objects, explore discrete structures, and apply optimization techniques
CO 4	**Utilizing** number theory concepts and understand cryptographic algorithms.

Course Outline:

Unit Number: 1	**Title: Set Theory, Logic, and Proofs**	**No. of hours: 10**

Set Theory: Definitions, Venn Diagrams, Operations on Sets, Cartesian Products, Power Sets.

Logic: Propositions, Logical Connectives, Truth Tables, Tautologies, Contradictions, Logical Equivalence, Conditional and Biconditional Statements.

Predicate Logic: Quantifiers, Proof Methods (Direct, Indirect, Contradiction, and Induction).

Boolean Algebra: Basic Operations, Boolean Expressions, Simplification, Logic Gates, Applications to Digital Circuits. Proofs using mathematical induction

Unit Number: 2	**Title: Relations and Functions**	**No. of hours:**

10

Relations: Definition, Properties (Reflexive, Symmetric, Transitive), Equivalence Relations, Partial Orderings.

Functions: Types of Functions (Injective, Surjective, Bijective), Composition of Functions, Inverses, Pigeonhole Principle.

Matrices and Relations: Matrix Representation, Warshall's Algorithm for Transitive Closure.

Counting: Permutations and Combinations, Pigeonhole Principle, Principle of Inclusion and Exclusion.

Unit Number: 3	**Title: Combinatorics and Recurrence Relations**	**No. of hours: 10**

Basic Counting Principles: Multiplication and Addition Principles, Permutations, Combinations.

Recurrence Relations: Solving Linear Recurrence Relations, Homogeneous and Non-Homogeneous Recurrences.

Generating Functions: Basics, Solving Recurrences using Generating Functions.

Binomial Theorem: Applications, Generalizations, Multinomial Theorem**.**

Unit Number: 4	**Title: Graph Theory and Trees**	**No. of hours: 10**

Graph Theory: Definitions, Types of Graphs (Undirected, Directed, Weighted), Graph Terminology (Degree, Path, Cycle, Connectedness), Adjacency Matrix, Incidence Matrix.

Graph Algorithms: Shortest Path (Dijkstra's Algorithm), Minimum Spanning Tree (Kruskal’s and Prim’s Algorithms).

Trees: Properties, Tree Traversals (Preorder, Inorder, Postorder), Binary Trees, Spanning Trees.

Planar Graphs: Euler's Formula, Graph Coloring, Applications in Networks.

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key concepts in discrete mathematics using PPTs and examples.
2. **Conceptual Understanding**: Cover topics like logic, set theory, combinatorics, and graph theory.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on solving discrete math problems.
4. **Theory Assignments**: Assign theoretical problems, with solutions discussed in class.
5. **Group Work**: Collaborate on projects involving applications of discrete mathematics.
6. **Case Studies**: Analyze real-world applications of discrete math in computer science and cryptography.
7. **Continuous Feedback**: Implement quizzes and feedback sessions to assess understanding.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home problems emphasizing discrete math concepts.
2. **Lab Projects**: Facilitate hands-on activities applying discrete math to programming or algorithms.
3. **Question Bank**: Provide practice problems and resources for self-assessment.
4. **Online Forums**: Create platforms for discussing discrete math problems and solutions.
5. **Self-Study for Case Studies**: Encourage independent research on applications of discrete mathematics.
6. **Collaborative Projects**: Organize group projects focused on real-world problems using discrete math.

Text Books:

1. Lipschutz, S., & Lipson, M. (2007). Schaum's Outline of Discrete Mathematics. McGraw-Hill Education.
2. Rosen, K. H. (2011). Discrete Mathematics and Its Applications. McGraw-Hill Education.
3. Graham, R. L., Knuth, D. E., & Patashnik, O. (1994). Concrete Mathematics: A Foundation for Computer Science. Addison-Wesley.
4. Johnsonbaugh, R. (2010). Discrete Mathematics. Pearson.
5. Grimaldi, R. P., & Ramana, B. V. (2014). Discrete and Combinatorial Mathematics: An Applied Introduction. Pearson.

References

R 1. Elements of Discrete Mathematics, C. L Liu, McGraw-Hill Inc, 1985. Applied Combinatorics, Alan Tucker.

R 2. Concrete Mathematics, Ronald Graham, Donald Knuth, and Oren Patashnik, 2nd Edition - Pearson Education Publishers.

DATA STRUCTURES

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **Data Structures**	**Course Code**	**L** **-** **T** **-**	**Credits**

	P	
ENCS205	4-0-0	4

Type of Course: Major-10

Pre-requisite(s), if any: Basics of Computer Programming

Course Perspective. This course provides a comprehensive introduction to data structures and algorithms, essential components in the field of computer science that are critical for designing efficient software systems. Data structures serve as the building blocks for data management and organization, crucial for implementing effective algorithms that solve real-world computational problems. The course is structured to not only impart theoretical knowledge but also practical skills through hands-on implementation and problem-solving.
The curriculum is meticulously designed to cover a range of topics from basic to advanced data structures, enabling students to understand and apply various data management techniques effectively.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** and apply basic and advanced data structures.
CO 2	**Analyzing** and compare various sorting and searching algorithms.
CO 3	**Designing** and utilize algorithms for advanced data manipulation.
CO 4	**Implementing** and evaluate algorithms using hashing and advanced algorithmic techniques.

Course Outline:

Unit Number : 1	Title: Foundations of Data Structures	No. of hours: 9

Introduction: Abstract Data Type, Elementary Data Organization.

Measuring efficiency of an Algorithm: Time and Space Complexity Analysis, Asymptotic notations.

Arrays: Single and Multidimensional Arrays, Representation of Arrays: Row Major Order, and Column Major Order, Application of arrays, Sparse Matrices.

Unit Number : 2	Title: Linear Data Structures	No. of hours: 11

Linked lists: Array and Dynamic Implementation of Single Linked Lists, Doubly Linked List, Circularly Linked List, Operations on a Linked List. Insertion, Deletion, Traversal, Polynomial Representation, Addition and Multiplication.

Stacks: Stack operations: Push & Pop, Array and Linked list implementation of Stack, Applications: Prefix and Postfix Expressions, Evaluation of postfix expression, Recursion.

Queues: Queue operations: Create, Add, Delete, full and empty queues, Array and linked implementation of queues, Dequeue, Circular queues and Priority Queue.

Unit Number : 3	Title: Searching and Sorting	No. of hours: 10

Searching: Sequential search, Binary Search.

Sorting: Insertion Sort, Selection, Bubble Sort, Quick Sort, Merge Sort, Heap Sort, Radix Sort, Bucket Sort, Shell Sort.

Hashing: Hash Function, Hash Table, Collision Resolution Strategies.

Unit Number : 4	Title: Trees & Graph Algorithms	No. of hours: 10

Trees: Basic terminology, Binary Trees, Array and linked list implementation, Types of Binary Tree, Extended Binary Trees, Algebraic Expressions, Tree Traversal algorithms: Inorder, Preorder and Postorder, Threaded Binary trees, Search, Addition and deletion of an element in a binary tree, AVL Trees, Heaps, B Trees, B+ Trees and their applications, Evaluating an expression tree

Graphs: Representation (Matrix and Linked), Traversals, Shortest path, Topological sort. Dijkstra's Algorithm, Floyd Warshall's Algorithm, Minimum Spanning Tree Algorithms (Kruskal's Algorithm, Prim's Algorithm).

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures:** Introduce key concepts in data structures using PPTs and coding demonstrations.
2. **Conceptual Understanding**: Cover topics like arrays, linked lists, stacks, queues, trees, and graphs.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on implementing and using various data structures.
4. **Theory Assignments**: Assign theoretical problems that reinforce data structure concepts, discussed in class.
5. **Group Work**: Collaborate on projects that require designing and optimizing data structures.
6. **Case Studies**: Analyze real-world applications of data structures in software development.
7. **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding and coding practices.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects that apply data structure concepts to practical problems.
2. **Lab Projects**: Facilitate hands-on programming tasks using data structures in real-world scenarios.
3. **Question Bank**: Provide practice problems and resources for self-assessment on data structures.

4. **Online Forums**: Create platforms for discussing data structure challenges and solutions.
5. **Self-Study for Case Studies**: Encourage independent research on efficient data structure implementations.
6. **Collaborative Projects**: Organize group projects focused on developing applications using various data structures.

Textbooks

1. Seymour Lipschutz, "Data Structures", 2nd Edition, 2015
2. Aaron Tanenbaum, "Data Structures Using C", 2nd edition, 2016
3. Ellis Horowitz and Sartaj Sahni, "Fundamentals of data structures" 2nd edition, 2017
4. Data Structures Using C (2nd. ed.). Reema Thareja. Oxford University Press, Inc., USA. 2018.

References

1. E. Horowitz and S. Sahani, "Fundamentals of Data Structures", Galgotia Book source Pvt. Ltd.
2. Data Structures & Algorithms in Python by John Canning, Alan Broder, Robert Lafore Addison-Wesley Professional ISBN: 9780134855912.
3. "Introduction to Algorithms" by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein.
4. Problem Solving with Algorithms and Data Structures Using Python" by Brad Miller and David Ranum.

Additional Readings:

Online References for Learning Data Structures

I) **MIT OpenCourseWare - Introduction to Algorithms (6.006)**
   a. Free course materials from MIT's undergraduate course on algorithms, which includes data structures. Lectures, assignments, and exams are available online.
   b. Link: MIT OpenCourseWare - Introduction to Algorithms

II) **LeetCode - Data Structures**
   a. A platform for practicing coding problems. It provides numerous problems related to data structures, complete with solutions and discussions.

b. Link: LeetCode - Data Structures

DATA STRUCTURES LAB

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **Data Structures lab**	**Course Code**	**L-T-P**	**Credits**
	ENCS253	0-0-2	1
Type of Course:	Major-12		
Version			
Contact Hours	40		

Pre-requisite(s), if any: Basics of Computer Programming with C++

Lab Experiments

Defined Course Outcomes

COs	
CO 1	**Analyzing** and evaluate the time and space complexity of algorithms for various scenarios, demonstrating an understanding of asymptotic notations.
CO 2	**Implementing** and manipulate single-dimensional and multi-dimensional arrays, including operations like insertion, deletion, and traversal.
CO 3	**Developing** and perform operations on linked lists (single, doubly, and circularly linked), stacks, and queues using both array and linked list representations.
CO 4	**Designing** and analyze the efficiency of different sorting and searching algorithms, as well as implement and compare advanced data structures like

binary search trees, AVL trees, and graph algorithms.

Experiment Title	**Mapped CO/COs**
Given an array of integers, perform the following operations: reverse the array, find the maximum and minimum elements, and calculate the sum and average of the elements. Implement functions to perform each operation and ensure the time complexity is optimal.	CO1
Given an array, rotate the array to the right by k steps, where k is non-negative. Implement the rotation in-place with O(1) extra space.	CO1
Write a function to merge two sorted arrays into a single sorted array. The function should handle arrays of different lengths and ensure the final array is sorted.	CO1

Given an array containing n distinct numbers taken from 0, 1, 2, ..., n, find the one that is missing from the array. Implement an efficient algorithm with O(n) time complexity.	CO1
Find the kth largest element in an unsorted array. Note that it is the kth largest element in sorted order, not the kth distinct element. Implement an efficient algorithm with O(n log n) time complexity.	CO1
Given an unsorted array of integers, find the length of the longest consecutive elements sequence. Your algorithm should run in O(n) time complexity.	CO1
Suppose an array sorted in ascending order is rotated at some pivot unknown to you beforehand. Write a function to search for a target value in the array. If found, return its index; otherwise, return -1. Your algorithm should run in O(log n) time complexity.	CO1
Given an integer array nums, find the contiguous subarray (containing at least one number) which has the largest sum and return its sum. Implement an efficient algorithm with O(n) time complexity using Kadane's Algorithm.	CO1
Write a function to move all zeros to the end of an array while maintaining the relative order of the non-zero elements. Implement the function with O(n) time complexity and O(1) extra space.	CO2
Write a class to implement a singly linked list with methods to insert an element at the head, insert an element at the tail, delete an element by value, and traverse the list to print all elements.	CO2
Using linked lists, write a function to add two polynomials. Each node in the linked list represents a term in the polynomial with its coefficient and exponent. Implement	CO2

the function to handle polynomials of different degrees.	
Implement a doubly linked list with methods to insert an element at the head, insert an element at the tail, delete an element by value, and reverse the list. Ensure that all operations handle edge cases appropriately.	CO2
Create a circular linked list with methods to insert an element, delete an element by value, and traverse the list. Ensure that the list maintains its circular nature after each operation.	CO2
Write a function to evaluate a given postfix expression using a stack. The function should support basic arithmetic operations (+, -, *, /) and handle invalid expressions gracefully.	CO2
Implement a stack using a singly linked list with methods for push, pop, and peek operations. Ensure that the stack handles edge cases, such as popping from an empty stack, appropriately.	CO2
Write a function to convert an infix expression to a postfix expression using a stack. The function should handle parentheses and operator precedence correctly.	CO2
Create a circular queue using an array with methods for enqueue, dequeue, and checking if the queue is empty or full. Ensure that the circular nature of the queue is maintained after each operation.	CO2
Given a sorted array that has been rotated at an unknown pivot, write a function to search for a target value in the array. If the target exists, return its index; otherwise, return -1. Implement an efficient algorithm with O(log n) time complexity using binary search.	CO2
Find the kth largest element in an unsorted array. Note that it is the kth largest element in sorted order, not the	CO

kth distinct element. Implement an efficient algorithm with O(n log n) time complexity.	2
Given a collection of intervals, merge all overlapping intervals and return an array of the non-overlapping intervals that cover all the intervals in the input. Implement an efficient algorithm with O(n log n) time complexity.	CO2
Given a non-empty array of integers, return the k most frequent elements. Implement an efficient algorithm with O(n log k) time complexity.	CO2
Given an array of integers that is already sorted in ascending order, find two numbers such that they add up to a specific target number. Return the indices of the two numbers (1-indexed) as an integer array. Implement an algorithm with O(n) time complexity.	CO2
Given an array that has been rotated at an unknown pivot, write a function to search for a target value in the array. Implement the solution using binary search with O(log n) time complexity.	CO3
Write a function to merge k sorted linked lists and return it as one sorted list. Implement an efficient solution using a min-heap with O(N log k) time complexity, where N is the total number of nodes.	CO3
Given a non-empty array of integers, return the k most frequent elements. Implement the solution with O(n log k) time complexity using a min-heap and a hash map.	CO3
Implement various sorting algorithms including Quick Sort, Merge Sort, Heap Sort, and analyze their performance on different input sizes. Ensure the implementation handles edge cases such as duplicate values and nearly sorted arrays.	CO3

Given preorder and inorder traversal of a tree, construct the binary tree. Implement an efficient algorithm with O(n) time complexity using a hash map to store the index of elements in the inorder traversal.	CO4
Implement Dijkstra's algorithm to find the shortest path from a source vertex to all other vertices in a weighted graph. Use both adjacency matrix and adjacency list representations for the graph. Ensure the algorithm handles negative weights appropriately.	CO4
Implement Kruskal's algorithm to find the minimum spanning tree of a graph. Use a union-find data structure to detect cycles and ensure the algorithm runs in O(E log E) time complexity.	CO4
Given a binary tree representing an arithmetic expression, write a function to evaluate the expression and return the result. Each leaf node is an operand, and each internal node is an operator. Implement an efficient recursive algorithm.	CO4

PROBABILISTIC MODELLING AND REASONING WITH PYTHON LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Probabilistic Modelling and Reasoning with Python Lab	**Course Code**	**L-T-P**	**Credits**
	SEC038	0-0-4	2
Type of Course:	SEC-3		
Pre-requisite(s), if any: Basic knowledge of the Statistics and Python			

Course Perspective The course on Probabilistic Modelling and Reasoning with Python Lab is designed to equip students with the skills to develop and apply probabilistic models for reasoning and decision-making under uncertainty. The course integrates theoretical concepts with practical Python programming, providing a hands-on approach to learning. Students begin by exploring the fundamentals of probability theory and its applications in modelling real-world phenomena. They learn to implement probabilistic models using Python libraries such as NumPy, SciPy, and PyMC3. The course is divided into 4 modules:

a) Introduction to Statistics

b) Probability Theory

c) Point Estimation

d) Test of Statistical Hypothesis and p-values

The Course Outcomes (COs).

COs	Statements
CO 1	**Explaining** the data gathering techniques
CO 2	**Inspecting** the data using descriptive statistics
CO 3	**Illustrating** the probability and conditional probability concepts

CO 4	**Distinguishing** between various probability distributions and analyze the data following different probability distributions
CO 5	**Solve** the inferential statistics problems using point and intervalestimation techniques. Infer the statistical problems using hypothesis testing and p value

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number:1	**Title: Introduction to Statistics**	**No. of hours: 8**
Content Summary: **Introduction to Statistics**: Role of statistics in scientific methods, current applications of statistics **Scientific data gathering:** Sampling techniques, scientific studies, observationalstudies, data management. Displaying data on a single variable (graphical methods, measure of central tendency, measure of spread), displaying relationship between two or more variables, measure of association between two or more variables.		
Unit Number:2	**Title: Probability Theory**	**No. of hours: 8**
Content Summary: Sample space and events, probability, axioms of probability, independent events, conditional probability, Bayes' theorem. **Random Variables:** Discrete and continuous random variables. Probability distribution of discrete random variables, binomial distribution, poisson distribution. Probability distribution of continuous random variables, The uniform distribution, normal (gaussian) distribution, exponential distribution, gamma distribution, beta distribution, t-distribution, χ^2 distribution. Expectations, variance andcovariance. Probability Inequalities. Bivariate distributions.		
Unit Number:3	**Title: Point Estimations**	**No. of hours: 8**

Content Summary: Methods of finding estimators, method of moments, maximum likelihood estimators, bayes estimators. Methods of evaluating estimators, mean squared error, best unbiased estimator, sufficiency and unbiasedness **Interval Estimations:** Confidence interval of means and proportions, Distribution free confidence interval of percentiles		
Unit Number:4	**Title: Test of Statistical Hypothesis and p-values**	**No. of hours: 8**
Content Summary: Tests about one mean, tests of equality of two means, test about proportions, p- values, likelihood ratio test, Bayesian tests **Bayesian Statistics:** Bayesian inference of discrete random variable, Bayesian inference of binomial proportion, comparing Bayesian and frequentist inferences of proportion, comparing Bayesian and frequentist inferences of mean **Univariate Statistics using Python:** Mean, Mode. Median, Variance, Standard Deviation, Normal Distribution, t-distribution, interval estimation, Hypothesis Testing, Pearson correlation test, ANOVA F-test about intelligence, techniques required to solve AI problems, level of details required to model human intelligence, successfully building an intelligent problem, history ofAI		

Text Books

1. Achim Klenke, (2014), Probability Theory A Comprehensive Course Second Edition, Springer, ISBN 978-1-4471-5360-3
2. Christian Heumann, Michael Schomaker Shalabh (2016), Introduction to Statistics and Data Analysis With Exercises, Solutions and Applications in R, Springer International Publishing, ISBN 978-3-319-46160-1
3. Douglas C. Montgomery, (2012), Applied Statistics and Probability for Engineers, 5th Edition, , Wiley India, ISBN: 978-8-126-53719-8

Verbal Ability

Program Name	**B. Tech CSE with specialization in AI &ML**		
Course Name: Life Skills for Professionals - I	**Course Code**	**L-T-P**	**Credits**
	AEC006	3-0-0	3

Type of Course: AEC-1

Duration: 36 hours

Pre-requisite(s), if any: Nil

Course Perspective: The course aims to improve language proficiency in three key areas: grammar, vocabulary and identification of grammatical errors in writing. Language proficiency enables students to comprehend lectures, understand course materials and enhances students' ability to express themselves clearly and effectively. In many professions, strong language skills are a prerequisite. Whether in business, medicine, law, or science, being able to communicate fluently and accurately is essential for collaboration, negotiation, and advancement. A strong command of verbal abilities can significantly impact job interviews. It allows candidates to answer questions confidently, demonstrate their qualifications effectively and leave a positive impression on potential employers.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the grammar rules and word meaning (Vocabulary).
CO 2	**Applying** grammar rules and vocabulary in different context & purpose
CO 3	**Analyzing** situations/ context of communication and selecting appropriate grammar and words.
CO 4	**Developing** sentences and paragraphs to describe and narrate a situation.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Vocabulary Development and Application**	**No. of hours: 10**

Content:

Understanding the concept of root words, Prefix and suffix, Ways to enhance Vocabulary, Crosswords and word quizzes, Confusing words, One word substitution, Odd one out, Synonyms and Antonyms, Commonly misspelt words, Idioms and Phrases.

Unit Number: 2	**Title: Fundamentals of Grammar and Sentence Structure**	**No. of hours: 8**

Content: Introduction to Parts of Speech, Tenses and its ‘rules, Sentences (Simple, Compound and Complex), Subject Verb Agreement, Pronoun Antecedent agreement, Phrases and Clauses.

Unit Number: 3	**Title: Mastering Sentence Accuracy and Completion Skills**	**No. of hours: 12**

Content:

Spot the error (grammatical errors in a sentence), Sentence Correction (Improvement of sentences based on Grammar rules), Sentence Completion, Cloze Tests

Unit Number: 4	**Title: Enhancing Sentence Structure and Reading Comprehension**	**No. of hours: 6**

Content:

Logical Arrangement of Sentences, Comprehending passages, Contextual questions, Anagrams, Analogies

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key life skills concepts using PPTs and real-life examples.
2. **Conceptual Understanding**: Cover topics like communication, teamwork, and problem-solving strategies.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on practical scenarios and decision-making.
4. **Theory Assignments**: Assign reflective essays on personal development and professional growth.
5. **Group Work**: Collaborate on projects that enhance interpersonal skills and teamwork.
6. **Case Studies**: Analyze successful professionals and their life skills in various industries.
7. **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding and application of skills.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects focused on applying life skills in real-world contexts.
2. **Workshops**: Facilitate hands-on sessions for practicing communication and leadership skills.
3. **Question Bank**: Provide resources for self-assessment on life skills development.
4. **Online Forums**: Create platforms for discussing life skills challenges and sharing experiences.
5. **Self-Study for Case Studies**: Encourage independent research on effective life skills practices.
6. **Collaborative Projects**: Organize group projects aimed at community engagement and skill application.

Mapping /Alignment of COs with POs & PSOs

References

1. **R1.** Norman Lewis – Word Power Made Easy
2. **R2.** Wren & Martin – High School English Grammar & Composition
3. **R3.** R.S. Agarwal & Vikas Agarwal – Quick Learning Objective General English
4. **R4.** S.P. Bakshi - Objective General English

5. **R 5.** Praxis Groups -Campus Recruitment Complete Reference

Additional Readings:

Communication Resources

I) **LinkedIn Learning - Communication Foundations**

   a. This course offers foundational knowledge on effective communication strategies, including verbal and non-verbal communication.

   b. Link: LinkedIn Learning - Communication Foundations

II) **Khan Academy - Grammar**

   a. Khan Academy provides a detailed course on grammar which is fundamental to clear and effective communication.

   b. Link: Khan Academy - Grammar

Non-Verbal Communication Resources

R 1. LinkedIn Learning - Developing Your Emotional Intelligence

- Enhancing emotional intelligence is key to improving non-verbal communication skills. This course covers practical strategies.
- Link: LinkedIn Learning - Developing Your Emotional Intelligence

R 2. YouTube - TED Talks on Body Language

- TED Talks provide insightful videos on the importance of body language and how to master it.
- Link: YouTube - TED Talks on Body Language

Number Systems and Basic Mathematics Resources

1. **Khan Academy - Arithmetic and Pre-Algebra**
   - Comprehensive lessons on basic arithmetic and pre-algebra, including number systems, divisibility, and more.
   - Link: Khan Academy - Arithmetic and Pre-Algebra
2. **Coursera - Introduction to Mathematical Thinking**
   - This course introduces mathematical thinking, including logic, which is fundamental to understanding number systems.
   - Link: Coursera - Introduction to Mathematical Thinking
3. **edX - Introduction to Algebra**

- Offers a strong foundation in algebra, covering topics such as factors, LCM, HCF, and simplification.
- Link: edX - Introduction to Algebra

Time Management Resources

1. **Coursera - Work Smarter, Not Harder: Time Management for Personal & Professional Productivity**
    - This course provides practical strategies for effective time management and productivity.
    - Link: Coursera - Time Management
2. **edX - Time Management Strategies for Project Management**
    - Focuses on time management within the context of project management, offering valuable insights and techniques.
    - Link: edX - Time Management Strategies

VAC III

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**			
Course Name: Community Engagement Service	**Course Code:**	**L-T-P**	**Credits**	**Semester**
	VAC III	2-0-0	2	II
Type of Course:	Value Added Course			
Duration	30 Hrs			

Course Objectives:

- To engage students in meaningful social service activities.
- To develop socially responsible engineers.
- To apply technical and non-technical skills for the benefit of society.
- To foster community engagement and support.

Course Outline:

1. Introduction

Overview of the Course: The **Community Engagement Service (VAC II)** course at K.R. Mangalam University is designed to integrate social responsibility with technical education.

This 30-hour value-added course encourages students to engage in meaningful social service activities, applying their technical and non-technical skills to benefit various sections of society. Through hands-on involvement, students will develop a deeper understanding of community needs and contribute positively to societal development.

Importance of Social Service in Engineering Education: Incorporating social service into technical education is crucial for nurturing well-rounded professionals who are not only technically proficient but also socially conscious. By participating in community-oriented projects, students can bridge the gap between theory and practice, gaining real-world experience that enhances their problem-solving skills. Engaging in social service fosters empathy, teamwork, and leadership qualities, which are essential attributes for successful engineers dedicated to making a positive impact on society.

Expectations and Requirements: Students enrolled in this course are expected to actively participate in chosen social service activities, dedicating at least 30 hours over weekends. They must document their engagement through video clips and photographs, maintaining a detailed logbook of their activities. Additionally, students are required to prepare a comprehensive report and a 10-minute video presentation demonstrating their engagement, learning experiences, and the impact of their initiatives. Evaluation will be based on the quality and relevance of documentation, the depth of the report, and the effectiveness of the video presentation in showcasing their contributions and outcomes.

2. Possible Engagement Activities

Students can choose from a variety of activities, including but not limited to:

Development and Innovation

Develop Innovative Tools: Create solutions such as mobile apps and web-based platforms to address societal needs.

1. **Lever-Powered Wheelchairs**: Develop control applications to enhance mobility for differently-abled individuals.
2. **Assistive Devices**: Design simple devices using basic sensors to improve daily living for people with disabilities.
3. **Environmental Monitoring**: Build introductory systems using Arduino and web dashboards to raise community awareness about air and water quality.
4. **Eco-Friendly Practices**: Create web applications that promote sustainable living and track user participation.
5. **Waste Management**: Implement basic data management systems for efficient waste management in local communities.
6. **Energy Optimization**: Develop algorithms to optimize energy consumption in households and public buildings.

7. **Water Quality Monitoring**: Design systems with sensors and mobile apps to ensure safe drinking water in rural areas.
8. **Smart Agriculture**: Create tools using microcontrollers to support farmers with automated irrigation and soil condition monitoring.
9. **Cybersecurity**: Implement basic practices to protect sensitive data in sustainable technology applications.
10. **Health Tracking**: Develop simple mobile applications to monitor fitness and wellness metrics, benefiting public health initiatives.
11. **Recycling Sorters**: Create introductory computer vision projects for sorting recyclables to aid municipal recycling programs.
12. **Environmental Data Analysis**: Conduct basic projects on environmental data sets to identify trends and propose solutions for urban planning and conservation efforts.
13. **Chemical Analysis Programs**: Create Python programs to support educational institutions.
14. **Electronic Circuits for Physics**: Develop circuits to aid students in experiments.
15. **Engineering Mathematics Tools**: Design simulation tools to assist in academic research.

Education and Mentorship

1. **Tutoring and Mentorship**: Provide tutoring and mentorship to underprivileged children.
2. **Day Camps**: Organize and run day camps for low-income children during weekends.
3. **Educational Opportunities for Incarcerated Individuals**: Volunteer to provide educational programs and mentorship to incarcerated individuals.
4. **Skill Development Workshops**: Conduct workshops to teach various skills to children based on students' expertise.

Community Service and Development

1. **Local Charities and Community Projects**: Volunteer with local charities to support community development projects.
2. **Entrepreneurship Initiatives**: Help villagers improve their livelihood through entrepreneurship initiatives.
3. **Women Empowerment Programs**: Empower women through skill enhancement, awareness programs, and entrepreneurship training.
4. **Digital Awareness Programs**: Conduct programs on cybersecurity and social media safety to protect against digital frauds.

Cultural and Traditional Skills

1. **Traditional Skills Learning**: Spend time with villagers to learn traditional skills such as pottery, carpentry, weaving, etc.
2. **Artisan Marketing Assistance**: Help artisans market their crafts through digital platforms and e-commerce.

Technology for Social Good

1. **Problem-Solving with Technology**: Use technology to solve specific problems faced by certain sections of society, such as developing apps for community support.
2. **Community Development Tools**: Create tools and resources to assist in community development and problem-solving.

Healthcare Domain

1. **Health Awareness Campaigns**: Organize campaigns to raise awareness about hygiene, nutrition, and preventive healthcare.
2. **Medical Camp Assistance**: Volunteer at medical camps to support healthcare delivery in underserved areas.
3. **Mental Health Support**: Conduct workshops and support groups focusing on mental health awareness and assistance.
4. **Telemedicine Services**: Assist in setting up and running telemedicine services for remote communities.

Print Media and Social Platforms

1. **Community Newsletters**: Create and distribute newsletters to share important community news and stories.
2. **Social Media Campaigns**: Run social media campaigns to raise awareness on various social issues and promote community initiatives.

Other Possible Domains

1. **Environmental Conservation**: Participate in tree planting drives, clean-up campaigns, and conservation projects.
2. **Disaster Relief Support**: Assist in disaster relief efforts, providing aid and support to affected communities.
3. **Animal Welfare**: Volunteer at animal shelters, support animal rescue operations, and promote animal welfare initiatives.
4. **Cultural Preservation**: Work on projects to preserve and promote local cultural heritage and traditions.

3. Documentation and Proof of Engagement

- Students must provide relevant proofs in the form of video clips and day-wise photographs.
- Maintain a logbook detailing the hours spent and activities undertaken.

4. Reporting and Presentation

- Prepare a detailed report on the engagement activities.
- Create a 10-minute video demonstrating the overall engagement, learning experiences, and impact.
- The video should include testimonials from beneficiaries showcasing the outcomes and benefits.

Evaluation Criteria:

The evaluation of the VAC will be based on the following rubrics, totaling 100 marks:

Criteria	**Marks**
Relevant Proofs (video clips, day-wise photographs)	20
Detailed Report	30
Video Presentation (10 minutes)	50
- Demonstration of overall engagement	
- Learning experiences	
- Initiative impact on society	
- Testimonials from beneficiaries	

Rubrics for Evaluation:

Evaluation Criteria	**Excellent (10)**	**Good (7-9)**	**Satisfactory (5-6)**	**Needs Improvement (1-4)**
Relevant Proofs	Comprehensive and well-documented	Adequately documented	Basic documentation provided	Inadequate or missing documentation

Evaluation Criteria	**Excellent (10)**	**Good (7-9)**	**Satisfactory (5-6)**	**Needs Improvement (1-4)**
Detailed Report	Thorough, well-structured, insightful	Clear and informative	Basic structure and content	Lacks detail and structure
Video Presentation	Highly engaging and impactful	Engaging and informative	Basic engagement and clarity	Lacks engagement and clarity
Demonstration of Engagement	Clearly demonstrates active and meaningful engagement	Shows active involvement	Demonstrates some involvement	Lacks clear demonstration of involvement
Learning Experiences	Profound insights and reflections	Clear insights and reflections	Basic reflections	Lacks depth in reflections
Initiative Impact on Society	Significant positive impact shown	Evident positive impact	Some positive impact	Minimal or unclear impact
Testimonials from Beneficiaries	Strong and compelling testimonials	Clear and supportive testimonials	Basic testimonials	Lack of or weak testimonials

Implementation Plan:

1. **Orientation Session:** Introduce students to the VAC and explain the objectives and expectations.
2. **Activity Selection:** Students select their preferred engagement activities.
3. **Engagement Phase:** Students actively participate in the chosen activities, documenting their involvement.
4. **Reporting Phase:** Students prepare their detailed report and video presentation.
5. **Evaluation:** Faculty evaluates students based on the provided rubrics.
6. **Feedback Session:** Provide constructive feedback to students for continuous improvement.

Conclusion:

This Value-Added Course aims to instill a sense of social responsibility in engineering students, encouraging them to apply their skills for the betterment of society. By engaging in various social service activities, students will gain valuable experiences that complement their technical education, fostering holistic development and community engagement.

Student Report Template

Title Page:

- Course Title: Community Engagement Service (VAC-II)
- Student Name:
- Enrollment Number:
- Semester: II
- Program: B.Tech (CSE) including all Specializations, BCA, B.Sc
- Date:

1. Introduction:
   - Overview of the Course: Provide a brief overview of the Community Engagement Service (VAC II) course, highlighting its purpose and importance.
   - Importance of Social Service in Engineering Education: Discuss why incorporating social service into engineering education is crucial for developing well-rounded professionals.
   - Expectations and Requirements: Outline the course expectations, including participation, documentation, and reporting requirements.
2. Chosen Activity:
   - Activity Name: State the name of the chosen social service activity.
   - Description of the Activity: Provide a detailed description of the activity.
   - Objectives and Goals: List the objectives and goals of the activity.
3. Methodology:
   - Steps Taken: Describe the steps taken to complete the activity.
   - Tools and Techniques Used: Mention any tools or techniques used, such as mobile apps, web-based platforms, etc.
   - Duration of Engagement: Specify the duration of the engagement (at least 30 hours).
4. Implementation:
   - Detailed Description of Engagement Activities: Provide a detailed log of the engagement activities, including day-wise descriptions.
   - Proof of Engagement: Include video clips, photographs, and other relevant proofs of engagement.
5. Impact Analysis:
   - Impact on Society: Analyze the impact of the activity on society.
   - Benefits to the Community: Discuss the benefits provided to the community.
   - Testimonials from Beneficiaries: Include testimonials from beneficiaries showcasing the outcomes and benefits.
6. Learning Experiences:
   - Skills and Knowledge Gained: Detail the skills and knowledge gained through the activity.
   - Reflections on the Experience: Reflect on the overall experience.
   - Challenges Faced and Overcome: Describe any challenges faced and how they were overcome.

7. Ethical Considerations:
    - Ethical Issues Encountered: Discuss any ethical issues encountered during the activity.
    - Solutions and Best Practices: Provide solutions and best practices for addressing these ethical issues.
    - Reflections on Social Responsibility: Reflect on the importance of social responsibility.
8. Conclusions:
    - Summary of the Experience: Summarize the overall experience.
    - Personal Growth and Development: Discuss personal growth and development resulting from the activity.
    - Future Recommendations: Provide recommendations for future engagements.
9. Appendices:
    - Additional Documents and Proofs: Include any additional supporting documents, such as logbook entries and extra photographs.
    - Video Presentation Link: Provide a link to the video presentation.

Summer Internship-I

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Summer Internship-I	**Course Code**	**L-T-P**	**Credits**

ENSI251	0-0-0	2

Type of Course: INT-1

Pre-requisite(s), if any: NA

Duration:

The internship will last for **six weeks**. It will take place after the completion of the 2nd semester and before the commencement of the 3rd semester.

Internship Options:

Students can choose from the following options:

1. **Industry Internship (Offline):**
    1. Students must produce a joining letter at the start and a relieving letter upon completion.
2. **Global Certifications:**
    1. Students can opt for globally recognized certification programs relevant to their field of study.
3. **Research Internship:**
    1. Students can engage in a research internship under the mentorship of a faculty member for six weeks.
4. **On-Campus Industry Internship Programs:**
    1. The university will offer on-campus internships in collaboration with industry partners.
5. **Internships at Renowned Institutions:**
    1. Students can pursue summer internships at esteemed institutions such as IITs, NITs, Central Universities, etc.

Report Submission and Evaluation:

1. **Report Preparation:**
    1. Students must prepare a detailed report documenting their internship experience and submit it to the department. A copy of the report will be kept for departmental records.

2. **Case Study/Project/Research Paper:**
    1. Each student must complete one of the following as part of their internship outcome:
        1. A case study
        2. A project
        3. A research paper suitable for publication
3. **Presentation:**
    1. Students are required to present their learning outcomes and results from their summer internship as part of the evaluation process.

Evaluation Criteria for Summer Internship (Out of 100 Marks)

1. Relevance to Learning Outcomes (30 Marks)

1. **Case Study/Project/Research Paper Relevance (15 Marks):**
    - Directly relates to core subjects: 15 marks
    - Partially relates to core subjects: 10 marks
    - Minimally relates to core subjects: 5 marks
    - Not relevant: 0 marks
2. **Application of Theoretical Knowledge (15 Marks):**
    - Extensive application of theoretical knowledge: 15 marks
    - Moderate application of theoretical knowledge: 10 marks
    - Minimal application of theoretical knowledge: 5 marks
    - No application of theoretical knowledge: 0 marks

2. Skill Acquisition (30 Marks)

1. **New Technical Skills Acquired (15 Marks):**
    - Highly relevant and advanced technical skills: 15 marks
    - Moderately relevant technical skills: 10 marks
    - Basic technical skills: 5 marks
    - No new skills acquired: 0 marks
2. **Professional and Soft Skills Development (15 Marks):**
    - Significant improvement in professional and soft skills: 15 marks
    - Moderate improvement in professional and soft skills: 10 marks
    - Basic improvement in professional and soft skills: 5 marks

- No improvement: 0 marks

3. Report Quality (20 Marks)

- **Structure and Organization (10 Marks):**
    - Well-structured and organized report: 10 marks
    - Moderately structured report: 7 marks
    - Poorly structured report: 3 marks
    - No structure: 0 marks
- **Clarity and Comprehensiveness (10 Marks):**
    - Clear and comprehensive report: 10 marks
    - Moderately clear and comprehensive report: 7 marks
    - Vague and incomplete report: 3 marks
    - Incomprehensible report: 0 marks

4. Presentation (20 Marks)

- **Content Delivery (10 Marks):**
    - Clear, engaging, and thorough delivery: 10 marks
    - Clear but less engaging delivery: 7 marks
    - Somewhat clear and engaging delivery: 3 marks
    - Unclear and disengaging delivery: 0 marks
- **Visual Aids and Communication Skills (10 Marks):**
    - Effective use of visual aids and excellent communication skills: 10 marks
    - Moderate use of visual aids and good communication skills: 7 marks
    - Basic use of visual aids and fair communication skills: 3 marks
    - No use of visual aids and poor communication skills: 0 marks

Total: 100 Marks

Course Outcomes:

By the end of this course, students will be able to:

- **Apply Theoretical Knowledge:**
    - Integrate and apply theoretical knowledge gained during coursework to real-world industry or research problems.
- **Develop Technical Skills:**

    - Acquire and demonstrate advanced technical skills relevant to the field of computer science and engineering through practical experience.
- **Conduct Independent Research:**
    - Execute independent research projects, including problem identification, literature review, methodology design, data collection, and analysis.
- **Prepare Professional Reports:**
    - Compile comprehensive and well-structured reports that document the internship experience, project details, research findings, and conclusions.
- **Enhance Problem-Solving Abilities:**
    - Develop enhanced problem-solving and critical thinking skills by tackling practical challenges encountered during the internship.
- **Improve Professional and Soft Skills:**
    - Exhibit improved professional and soft skills, including communication, teamwork, time management, and adaptability in a professional setting.
- **Present Findings Effectively:**
    - Deliver clear and engaging presentations to effectively communicate project outcomes, research findings, and acquired knowledge to peers and faculty members.
- **Pursue Lifelong Learning:**
    - Demonstrate a commitment to lifelong learning by engaging in continuous skill development and staying updated with emerging trends and technologies in the field.

Learning Experiences

Classroom Learning Experience

1. **Orientation Sessions**: Introduce internship objectives and expectations through interactive presentations.
2. **Skill Development Workshops**: Cover essential skills like communication, teamwork, and time management.

3. **Project Planning**: Guide students in developing project proposals aligned with internship goals.
4. **Group Discussions**: Facilitate discussions on challenges and experiences in workplace settings.
5. **Guest Speakers**: Invite industry professionals to share insights and best practices.
6. **Continuous Feedback**: Implement regular check-ins and peer reviews to assess progress and learning.

Outside Classroom Learning Experience

1. **Internship Placement**: Engage students in real-world work environments to apply learned skills.
2. **Reflective Journals**: Encourage students to document their experiences and lessons learned during the internship.
3. **Project Implementation**: Work on assigned projects and tasks within the organization.
4. **Networking Opportunities**: Create platforms for students to connect with industry professionals.
5. **Self-Assessment**: Provide tools for students to evaluate their performance and growth.
6. **Final Presentations**: Organize sessions for students to present their internship experiences and outcomes.

COMPETITIVE CODING BOOTCAMP-I

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **COMPETITIVE CODING BOOTCAMP-I**	**Course Code**	**L-T-P** 2-0-0	**Credits** 0
Type of Course:	AUDIT-1		
Contact Hours	30		
Version			

Course Outcomes

CO1	**Understanding** problem-solving strategies and techniques relevant to competitive programming
CO2	**Analyzing** the efficiency of algorithms in terms of time and space complexity using asymptotic notations
CO3	**Applying** core programming concepts such as functions, recursion, and dynamic memory allocation to solve computational problems
CO4	**Implementing** solutions for problems involving arrays and strings, utilizing efficient operations and algorithms

Course Outline:

Unit Number: 1	**Title: Foundations of Competitive Programming**	**No. of hours: 8**

Content: Introduction to Competitive Programming Platforms ▪ Overview of major platforms: Codeforces, LeetCode, HackerRank etc. ▪ Setting up accounts and environment for competitive programming. ▪ Solving introductory problems to get familiar with the platforms. Problem-Solving Strategies ▪ Techniques for solving problems ▪ Greedy Algorithms: Understanding local optimality leading to global solutions. ▪ Divide and Conquer: Solving problems by breaking them into subproblems (with examples like Merge Sort). ▪ Brute Force: Iterative approach to solve problems when constraints are small.		
Unit Number: 2	**Title:** Time and Space Complexity of Algorithms	**No. of hours: 8**
Content: Time and Space Complexity: ▪ Big O Notation: Definition, examples, and practical importance. ▪ Common Complexities: O(1), O(log n), O(n), O(n log n), O(n^2), etc. ▪ Impact of time and space complexity on algorithm performance. ▪ Asymptotic notations ▪ Best, Average and worst case analysis of Algorithms		
Unit Number: 3	**Title: Core Programming Concepts**	**No. of hours: 8**
Content: **Functions:** Definition and Declaration, Function Overloading, Recursion and Backtracking **Pointers:** Basics of Pointers and References, Pointer Arithmetic, Dynamic Memory Allocation (malloc, free, new, delete)		

Files: File I/O Operations (Reading/Writing), File Handling in C++/Java/Python, **Vectors (in C++/ArrayLists in Java):** Declaration, Initialization, and Operations, Dynamic Resizing		
Unit Number: 4	**Title: Arrays and Strings**	**No. of hours: 6**
Content: Arrays: Operations, Manipulations Strings: Operations, Substrings, Pattern Matching Operations on arrays: Insertion, deletion, and traversal. String operations: Concatenation, substring search. Key Problems: Rotating arrays, reversing strings, finding longest substrings without repeating characters		

Experiment List

Problem Statement	Mapped COs
1. Two Sum: Find two numbers that add up to a specific target.	CO1
2. Best Time to Buy and Sell Stock: Maximize profit from stock prices.	CO1
3. Valid Parentheses: Check if a string contains valid parentheses.	CO1
4. Greedy Algorithm: Jump Game - Can you reach the end of the array?	CO1
5. Divide and Conquer: Merge Sort implementation to sort an array.	CO1
6. Brute Force: Find all subsets of a given set.	CO1
7. Greedy Algorithm: Minimum Number of Platforms Required for Trains	CO1

Problem Statement	Mapped COs
8. Divide and Conquer: Maximum Subarray (Kadane's Algorithm)	CO1
9. Brute Force: Count number of occurrences of a substring in a string.	CO1
10. Greedy Algorithm: Coin Change Problem (Minimum Coins)	CO1
11. Time Complexity: Check if a number is prime using $O(\sqrt{n})$ complexity.	CO2
12. Sorting: QuickSort algorithm with $O(n \log n)$ complexity.	CO2
13. Big O Notation: Analyze time complexity of an algorithm.	CO2
14. Space Complexity: Fibonacci with $O(n)$ space complexity.	CO2
15. Time Complexity: Find first duplicate element in an array with $O(n)$ time.	CO2
16. Time Complexity: Search an element in a rotated sorted array in $O(\log n)$ time.	CO2
17. Complexity Analysis: Binary Search Tree operations with complexity $O(\log n)$.	CO2
18. Analyze best, average, and worst case for Insertion Sort.	CO2
19. Time and Space Complexity: Check the complexity of an algorithm (recurrences).	CO2
20. Time Complexity: Compute factorial recursively with complexity analysis.	CO2
21. Recursion: Generate all permutations of a string.	CO3
22. Dynamic Memory Allocation: Implement a dynamic array (vector) from scratch.	CO3
23. Backtracking: Solve the N-Queens problem using recursion.	CO3
24. Pointers: Swap two numbers using pointers in C++.	CO3
25. File Handling: Read and write data to a file in Python/C++/Java.	CO3

Problem Statement	Mapped COs
26. Function Overloading: Implement overloaded functions for adding integers and floats.	CO3
27. Dynamic Memory Allocation: Use malloc and free to manage memory in C.	CO3
28. Recursion: Solve Tower of Hanoi using recursion.	CO3
29. Arrays: Rotate an array to the right by k steps.	CO4
30. Strings: Find the longest substring without repeating characters.	CO4

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures**: Introduce competitive coding concepts and strategies using PPTs and coding demos.
2. **Algorithm Workshops**: Cover key algorithms and data structures essential for coding competitions.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on solving competitive coding problems.
4. **Mock Contests**: Organize timed coding contests to simulate competition environments.
5. **Group Discussions**: Facilitate discussions on problem-solving techniques and optimization strategies.
6. **Continuous Feedback**: Implement peer reviews and performance assessments after practice sessions.

Outside Classroom Learning Experience

1. **Practice Assignments**: Assign coding problems from various online platforms for independent practice.
2. **Online Competitions**: Encourage participation in external coding competitions and hackathons.
3. **Question Bank**: Provide a repository of practice problems and resources for self-assessment.

4. **Online Forums**: Create platforms for students to discuss coding challenges and share solutions.
5. **Self-Study Resources**: Recommend books and online courses for further learning on algorithms and data structures.
6. **Collaborative Projects**: Organize group projects to develop coding applications or solve larger problems together.

Textbooks:

- "Introduction to Algorithms" by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein.
- "Algorithm Design" by Jon Kleinberg and Éva Tardos.

Online Resources:

- LeetCode (https://leetcode.com/)
- HackerRank (https://www.hackerrank.com/)
- GeeksforGeeks (https://www.geeksforgeeks.org/)

List of Suggested Competitive Programming Courses:

1. Algorithms and Data Structures by MIT OpenCourseWare
2. Introduction to Competitive Programming by NPTEL
3. Competitive Programming by HackerRank
4. The Bible of Competitive Programming & Coding Interviews

All students must complete one online course from the suggested programs.

Web References

- https://www.geeksforgeeks.org/competitive-programming-a-complete-guide/
- https://www.geeksforgeeks.org/must-do-coding-questions-for-companies-like-amazon-microsoft-adobe/
- https://github.com/parikshit223933/Coding-Ninjas-Competitive-Programming
- https://www.hackerearth.com/getstarted-competitive-programming/

References to Interview Questions

- https://www.simplilearn.com/coding-interview-questions-article
- https://www.csestack.org/competitive-coding-questions/

- https://www.geeksforgeeks.org/a-competitive-programmers-interview/

SEMESTER: IV

ANALYSIS AND DESIGN OF ALGORITHMS

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **Analysis and Design of Algorithms**	**Course Code**	**L-T-P**	**Credits**
	ENCS202	4-0-0	4
Type of Course:	**Major-13**		

Pre-requisite(s), if any: - basic understanding of Data Structure and any programming language

Course Perspective The course provides a comprehensive introduction to the fundamental concepts of algorithm analysis and design, essential for various fields such as computer science, engineering, data science, and artificial intelligence. This course equips students with the tools to understand, analyze, and develop efficient algorithms for solving complex computational problems. By covering both theoretical foundations and practical applications, the course ensures a balanced approach to learning. The course is divided into 5 modules:

a) Introduction and Complexity Analysis

b) Divide and Conquer, Greedy Algorithms, and Dynamic Programming

c) Graph Algorithms

d) Advanced Algorithms and Techniques

e) Advanced Topics and Implementation Techniques

The Course Outcomes (COs).

COs	Statements

CO 1	**Understanding** fundamental algorithmic concepts and analyze their complexities.
CO 2	**Analyzing and evaluating** the performance of various algorithms.
CO 3	**Designing** efficient algorithms considering both time and space complexities.
CO 4	**Applying** algorithmic problem-solving strategies to a variety of computational problems.
CO 5	**Developing** skills to implement and optimize algorithms for real-world applications.

Course Outline:

Unit Number: 1	**Title:** Introduction and Complexity Analysis	**No. of hours: 10**

Content Summary:

Introduction to Algorithms: Definition, importance, specification and role in problem-solving.

Algorithm Analysis: RAM computational models, Time and space complexity, Asymptotic Notations, best, average, and worst-case analysis, Performance measurement of algorithms, rate of growth of algorithms

Recurrence Relations: Solving recurrences using substitution, recursion tree, and master theorem.

Sorting: Analysis of Time complexities of comparison and Linear sorting Algorithms

Unit Number: 2	**Title:** Divide and Conquer, Greedy Algorithms, and Dynamic Programming	**No. of hours: 10**

Content Summary:

Divide and Conquer: General method, Merge Sort, Quick Sort, Binary Search, Strassen's Matrix Multiplication, finding maximum and minimum.

Greedy Algorithms: Concept and characteristics, Fractional Knapsack, Activity Selection, Huffman Coding.

Dynamic Programming: General Method, Longest Common Subsequence, 0/1 Knapsack problem, Matrix Chain Multiplication, Travelling salesman problem.

Unit Number: 3	**Title:** Graph Algorithms	**No. of hours: 10**

Content Summary:

Graph Representation: Adjacency matrix, adjacency list.

Graph Traversal Algorithms: Depth First Search (DFS), Breadth First Search (BFS), Applications of graph (Topological sorting).

Shortest Path Algorithms: Dijkstra's algorithm, Bellman-Ford algorithm, Floyd-Warshall algorithm.

Minimum Spanning Tree Algorithms: Kruskal's algorithm, Prim's algorithm.

Unit Number: 4	**Title**: Advanced Algorithms and Techniques	**No. of hours: 10**

Content Summary:

Backtracking: Concept, examples (N-Queens problem, Sum of subsets).

Branch and Bound: Concept, examples (Traveling Salesman Problem, 0/1 Knapsack Problem).

String Matching Algorithms: Naive algorithm, Rabin-Karp algorithm, String matching with finite automata, Knuth-Morris-Pratt (KMP) algorithm.

Introduction to NP-Completeness: The class P and NP, Polynomial time, NP-complete and NP-hard.

Introduction to Approximation Algorithms and Randomized Algorithms

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key concepts in algorithm design and analysis using PPTs and examples.

2. **Conceptual Understanding**: Cover fundamental topics like complexity analysis, recursion, and algorithmic paradigms.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on designing and analyzing various algorithms.
4. **Case Studies**: Analyze real-world algorithms and their applications in different fields.
5. **Group Work**: Collaborate on projects that involve implementing and optimizing algorithms.
6. **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding and application of concepts.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects requiring analysis and design of algorithms for practical problems.
2. **Lab Projects**: Facilitate hands-on programming tasks to implement and test algorithms.
3. **Question Bank**: Provide practice problems and resources for self-assessment on algorithm concepts.
4. **Online Forums**: Create platforms for discussing algorithm challenges and sharing solutions.
5. **Self-Study for Case Studies**: Encourage independent research on advancements in algorithm design and analysis.
6. **Collaborative Projects**: Organize group projects focused on solving complex problems using algorithms.

Text Books

1. Introduction to Algorithms, 4TH Edition, Thomas H Cormen, Charles E Lieserson,

   Ronald L Rivest and Clifford Stein, MIT Press/McGraw-Hill.
2. Fundamentals of Algorithms – E. Horowitz et al.

Additional Readings:

Online Learning Resources :

I) **MIT OpenCourseWare - Introduction to Algorithms (6.006)**

a. A comprehensive resource from MIT covering fundamental and advanced algorithms.
b. Link: MIT OpenCourseWare - Introduction to Algorithms

II) **HackerRank - Algorithms Practice**

a. Provides a platform to practice and compete in coding challenges related to algorithms.
b. Link: HackerRank - Algorithms Practice

III) **LeetCode - Algorithm Problems**

a. A platform offering a vast array of problems to practice algorithms and data structures.
b. Link: LeetCode - Algorithm Problems

ANALYSIS AND DESIGN OF ALGORITHMS LAB

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **Analysis and Design of Algorithms Lab**	**Course Code**	**L-T-P**	**Credits**
	ENCS256	0-0-2	1
Type of Course:	Major-15		

Pre-requisite(s), if any: - Data Structure

Defined Course Outcomes

COs	
CO 1	**Analyzing** the time and space complexity of algorithms, demonstrating an understanding of asymptotic notations and performance metrics.
CO 2	**Implementing** and compare sorting algorithms, such as bubble sort and insertion sort, and apply the divide and conquer technique to algorithms like merge sort and quick sort
CO 3	**Solving** optimization problems using greedy and dynamic programming algorithms, such as the fractional knapsack problem and longest common subsequence
CO 4	**Developing** graph algorithms for traversal, shortest path, and minimum spanning tree, applying techniques like DFS, BFS, Dijkstra's, and Kruskal's algorithms
CO 5	**Implementing** advanced algorithms for problems like N-Queens, traveling salesman, and string matching using backtracking, branch and bound, and pattern matching techniques

Lab Experiments

S. N	Lab Task	Mapped CO/COs
1	Conduct a case study on the efficiency of different sorting algorithms (e.g., Insertion Sort, Bubble Sort, Merge Sort, Quick Sort, Counting Sort, Radix sort, Bucket sort).	CO1
2	Develop a tool in your preferred programming language to measure the performance of various algorithms.	CO1
3	Develop a resource allocation system for a fictional company using greedy algorithms.	CO2
4	Implement solutions for the Longest Common Subsequence, 0/1 Knapsack Problem, and Matrix Chain Multiplication.	CO2
5	Create a file compression tool using the Huffman Coding algorithm. Allow users to input a text file and generate the corresponding Huffman tree and encoded output	CO2
6	Create a program to represent a social network using both adjacency matrix and adjacency list representations.	CO3
7	Develop a application that uses Depth First Search (DFS) and Breadth First Search (BFS) algorithms.	CO3
8	Design a city navigation system that calculates the shortest path between locations using Dijkstra’s algorithm, Bellman-Ford algorithm, and Floyd-Warshall algorithm.	CO3

9	Implement a solution to the N-Queens problem using backtracking. Allow the user to input the size of the chessboard (N) and display all possible solutions	CO4
10	Write a program to find all subsets of a given set of positive integers that sum up to a given value using the backtracking technique.	CO4
11	Develop the simulation of various string matching algorithms and compare their runtime complexities. Display their time complexity graphs	CO4
12	Implement the Branch and Bound technique to solve the Traveling Salesman Problem (TSP) and compare it with brute-force solutions.	CO4

DATABASE MANAGEMENT SYSTEMS

<table>
<tr><td>Program Name</td><td colspan="3">Bachelor of Technology (CSE) with specialization in AI & ML</td></tr>
<tr><td rowspan="2">Course Name:
Database Management System</td><td>Course Code</td><td>L-T-P</td><td>Credits</td></tr>
<tr><td>ENCS204</td><td>3-1-0</td><td>4</td></tr>
<tr><td>Type of Course:</td><td colspan="3">Major -14</td></tr>
<tr><td colspan="4">Pre-requisite(s), if any: Nil</td></tr>
</table>

Course Perspective. This course provides a comprehensive introduction to the fundamental concepts and advanced techniques of database management systems (DBMS). It is designed to equip students with the knowledge and skills required to design, implement, and manage databases effectively. The course covers a broad range of topics, including database architecture, data models, SQL, transaction management, concurrency control, database recovery, and security.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements

CO 1	**Understanding** the fundamental concepts and architecture of database management systems, including data models and ER modeling.
CO 2	**Utilizing** Structured Query Language (SQL) and relational algebra for effective database querying and manipulation.
CO 3	**Applying** database design principles, including normalization and integrity constraints, to develop well-structured databases.
CO 4	**Analyzing** storage structures, transaction processing, concurrency control, and recovery protocols in databases.
CO 5	**Implementing** security measures and explores advanced database concepts such as distributed databases, data warehousing, and data mining.

Course Outline:

Unit Number : 1	**Title: Introduction**	**No. of hours: 12**
Content: **Introduction to DBMS**: Overview, benefits, and applications. **Database System Architecture**: Schemas, Instances, Data abstraction, data models (network model, relational model, object-oriented data model), Three schema architecture and data independence **Entity-Relationship Model**: Entity Types, Entity Sets, Attributes, and Keys, Relationship Types, Relationship Sets, ER diagrams, Naming Conventions, Design issues. **Integrity Constraints**: Primary key, foreign key, unique, not null, check constraints.		
Unit Number : 2	**Title: Relational Query Languages**	**No. of hours: 8**

Content: Relational Database Design, Relational query languages, Relational algebra, Tuple and domain relational calculus. **SQL**: DDL (Data Definition Language), DML (Data Manipulation Language), DCL (Data Control Language). **Query Processing and Optimization**: Evaluation of relational algebra expressions, query equivalence, join strategies, query optimization algorithms. **Database Design**: Functional dependencies, normalization (1NF, 2NF, 3NF, BCNF, 4NF), dependency preservation, lossless decomposition. **Open Source and Commercial DBMS**: Overview of MySQL, Oracle, DB2, SQL Server.		
Unit Number : 3	**Title: Transaction Processing and Storage Strategies**	**No. of hours: 12**
Content: **Transaction Management**: ACID properties, transaction states, serializability, conflict and view serializability. **Concurrency Control**: Lock-based protocols, timestamp-based protocols, multi-version concurrency control, deadlock handling. **Database Recovery**: Recovery concepts, recovery techniques (log-based recovery, shadow paging), checkpoints. **Storage Strategies**: File organization, indexing (single-level, multi-level), B-tree, B+ tree, hashing (static and dynamic).		
Unit Number : 4	**Title: Advanced Topics and Database Security**	**No. of hours: 8**
Content: **Open-Source DBMS:** Hands-on experience with MySQL and PostgreSQL, Installation, Configuration, and Basic Operations, Hands-on: Create and manage databases, tables, and user privileges, Perform queries, and data manipulations, and use built-in functions **Fundamentals of MongoDB:** Introduction to MongoDB, document-oriented storage, CRUD operations, indexing, aggregation framework, Implement CRUD operations, design		

schemas, and optimize queries in MongoDB

Advanced Database Topics: Object-oriented databases, object-relational databases, logical databases, web databases.

Distributed Databases: Concepts, architecture, data fragmentation, replication, distributed query processing.

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key concepts in database management using PPTs and case studies.
2. **Conceptual Understanding**: Cover fundamental topics like data modeling, normalization, and SQL.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on database design and query optimization.
4. **Case Studies**: Analyze real-world database systems and their architectures.
5. **Group Work**: Collaborate on projects that involve designing and implementing databases.
6. **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding of database concepts.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects requiring the application of database management principles.
2. **Lab Projects**: Facilitate hands-on tasks to create, manipulate, and query databases using DBMS software.
3. **Question Bank**: Provide practice problems and resources for self-assessment on database topics.
4. **Online Forums**: Create platforms for discussing database challenges and sharing solutions.
5. **Self-Study for Case Studies**: Encourage independent research on current trends and technologies in database management.
6. **Collaborative Projects**: Organize group projects focused on developing database solutions for real-world problems.

Textbooks

1. R. Elmasri and S.B. Navathe, 2000, Fundamentals of Database Systems, 3rd Ed, AW.
2. C.J. Date, 2000, An Introduction to Database Systems, 7th ED., Addison-Wesley.
3. Database System Concepts", 6th Edition by Abraham Silberschatz, Henry F. Korth, S. Sudarshan, McGraw-Hill.

Additional Readings:

Online Learning Resources for "Database Management Systems"

1. NPTEL-Database Management System
   -
2. **MIT OpenCourseWare - Database Systems (6.830)**
   - Advanced course materials from MIT covering database system internals and advanced topics.
   - Link: MIT OpenCourseWare - Database Systems
3. **Oracle - Database 2-Day Developer's Guide**
   - Official documentation and guide for Oracle database developers.
   - Link: Oracle - Database 2-Day Developer's Guide
4. **SQLBolt - Learn SQL with interactive exercises**
   - Interactive SQL tutorials and exercises to practice database querying.
   - Link: SQLBolt - Learn SQL

DATABASE MANAGEMENT SYSTEMS LAB

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Database Management System Lab	**Course Code**	L-T-P	Credits
	ENCS254	0-0-2	1
Type of Course:	Major-16		

Defined Course Outcomes

COs	
CO1	**Designing and implementing** database schemas using both open-source and commercial DBMS, defining tables, relationships, and integrity constraints
CO2	**Developing and analyzing** Entity-Relationship diagrams, relational schemas, and enforce normalization techniques to ensure database efficiency and integrity
CO3	**Understanding** the Write and execute SQL queries for data definition, manipulation, and complex data retrieval, demonstrating proficiency in relational algebra and transaction processing
CO4	**Implementing** advanced database concepts including indexing, concurrency control, recovery techniques, security features, and distributed database processing

Lab Experiments

Ex. No	Lab Task	Mapped CO/COs
1	Analyze and document the benefits of using a DBMS over a traditional file system for managing data. Use a case study of a small retail business to highlight the advantages of a DBMS in handling inventory, sales, and customer data.	CO1
2	Design a three-schema architecture for a university management system. Create the internal schema, conceptual schema, and external schema. Illustrate how data independence is achieved and provide examples of each schema with specific details.	CO1
3	Design and implement an ER model for a university course registration system. The system should include entities such as Students, Courses, Professors, and Enrollments. Define relationships, attributes, and keys	CO1
4	Design a relational database schema for an e-commerce platform that manages Customers, Products, Orders, Order Details, and Payments.	CO2

	Define the entities, their attributes, and the relationships between them, ensuring the schema is normalized to at least 3NF. Use this schema to create an ER diagram and specify primary and foreign keys.	
5	Write SQL scripts to create the e-commerce platform database schema using Data Definition Language (DDL). Create tables for Customers, Products, Orders, Order Details, and Payments, and enforce primary keys, foreign keys, unique constraints, not null constraints, and check constraints. Ensure the database structure supports data integrity and consistency	CO2
6	Populating, Querying, and Securing the E-commerce Database using SQL DML, DCL, and Relational Algebra/Calculus	CO2
7	Design and implement a banking transaction management system that demonstrates the ACID properties. The system should handle various transaction states, ensuring serializability and data integrity. Implement features for depositing, withdrawing, and transferring funds, and simulate scenarios to showcase conflict and view serializability.	CO3
8	Develop a concurrency control mechanism for an e-commerce platform to manage simultaneous transactions, such as placing orders and updating inventory. Implement lock-based protocols, timestamp-based protocols, and multi-version concurrency control. Simulate scenarios where deadlock handling techniques are required to ensure smooth operation	CO3
9	Design and implement a data warehousing solution for a financial analytics platform. Create a data warehouse to store historical financial data and perform OLAP operations for data analysis. Implement data preprocessing techniques and apply data mining algorithms to discover patterns and insights from the financial data. Simulate various analytical queries and demonstrate how the data warehouse and mining techniques enhance decision-making and business intelligence.	CO4

MACHINE LEARNING AND PATTERN RECOGNITION

Program Name	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Machine Learning and Pattern Recognition	Course Code	L-T-P	Credits
	ENSP202	4-0-0	4
Type of Course:	IDC-4		
Pre-requisite(s), if any: Basic knowledge of the Statistics & Python.			

Course Perspective. A course on Machine Learning and Pattern Recognition is designed to equip students with the knowledge and skills needed to understand and apply algorithms for data analysis and predictive modeling. This course delves into the principles of machine learning, focusing on the development and application of algorithms to identify patterns and make predictions based on data. Students will explore various types of learning, including supervised, unsupervised, and reinforcement learning, and gain hands-on experience with algorithms such as decision trees, support vector machines, clustering techniques, and neural networks. The course is divided into 4 modules:

a) Introduction

b) Important concepts of machine learning

c) Linear Regression

d) Classification

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements

CO 1	Explaining the use of Machine Learning Models in business and understand machine learning models can be used to solve business problems.
CO 2	Comparing machine learning algorithms such as supervised, unsupervised, and reinforcement learning models
CO 3	Identifying the performance of different machine learning models and compare them to optimize the results
CO 4	Making use continuous and discrete data set to fit regression and classification models

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Introduction**	**No. of hours: 8**
Content Summary: Learning systems, real world applications of machine learning, why machinelearning, variable types and terminology, function approximation **Types of machine learning:** Supervised learning, unsupervised learning, Reinforcement learning		
Unit Number: 2	**Title: Important concepts of machine learning**	**No. of hours: 8**
Content Summary: Parametric vs non-parametric models, the trade-off between prediction accuracy andmodel interpretability, the curse of dimensionality, measuring the quality of fit, bias-variance trade off, overfitting, model selection, no free lunch theorem.		
Unit Number: 3	**Title: Linear Regression**	**No. of hours: 8**
Content Summary: Linear regression, estimating the coefficients, accessing the accuracy of coefficient estimates, accessing the accuracy of the model, multiple linear regression, qualitativepredictors		
Unit Number: 4	**Title: Classification**	**No. of hours: 8**

Content Summary:
Logistic regression, estimating regression coefficients, making predictions, multiple logistic regressions, linear discriminant analysis, bayes' theorem of classification, LDA for p=1, LDA for p>1, quadratic discriminant analysis

Learning Experiences

1. Interactive Lectures:
   - Presentations that introduce core concepts of machine learning and pattern recognition, supplemented with real-world examples to illustrate applications.
2. Hands-on Coding Sessions:
   - Practical lab sessions where students implement algorithms such as linear regression, logistic regression, and decision trees using Python libraries (e.g., Scikit-learn).
3. Group Discussions:
   - Facilitate discussions on the implications of different machine learning approaches (e.g., supervised vs. unsupervised learning) and their applications in various industries.
4. Workshops on Model Evaluation:
   - Conduct workshops focused on evaluating the performance of machine learning models, including metrics such as accuracy, precision, recall, and F1-score.
5. Case Study Analysis:
   - Analyze real-world case studies of successful machine learning implementations, allowing students to identify challenges and solutions.
6. Collaborative Problem-Solving:
   - Group activities where students work on specific problems or datasets, applying the concepts learned to propose and evaluate solutions.
7. Quizzes and Assessments:
   - Use quizzes and in-class assessments to reinforce understanding of key concepts and algorithms, providing timely feedback.

Outside Classroom Learning

1. Independent Research Projects:
   - Assign research projects where students investigate specific algorithms, tools, or case studies in machine learning and present their findings.
2. Online Courses and Tutorials:

- Encourage students to enroll in online courses (e.g., Coursera, edX) that complement the curriculum, focusing on specific machine learning techniques or frameworks.

3. Self-Directed Datasets Projects:
   - Allow students to select datasets from platforms like Kaggle to analyze independently, applying machine learning methods and presenting their results.
4. Discussion Forums:
   - Set up online discussion boards for students to ask questions, share insights, and collaborate on topics related to machine learning and pattern recognition.

Mapping /Alignment of C0s with POs

PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PSO1	PSO2	PSO3	PSO4	PSO5
CO1	-	3	-		3	3	3	3	3	2	2	-
CO2	-	3	-		3	3	3	3	3	2	2	3
CO3	-	-	-	3	3	-	3	3	-	2	2	-
CO4	-	3	3	3	3	3	3	2	2	1	1	2

- indicate no co-relation between CO and PO/PSO,

1 indicates the strength of co-relation between CO and PO/PSO is Weak/low,

2= strength of co-relation between CO and PO/PSO is Moderate/Medium,

3= strength of co-relation is Strong/High.

References

1. Machine Learning by Tom M. Mitchell - McGraw Hill Education; First edition
2. Pattern Recognition and Machine Learning (Information Science and Statistics) by Christopher M. Bishop - Springer; 1st ed. 2006. Corr. 2nd printing 2011 edition
3. The Elements of Statistical Learning: Data Mining, Inference, and Prediction by Trevor Hastie, Robert Tibshirani, Jerome Friedman - Springer; 2nd ed. 2009,
4. Corr. 9th printing 2017 edition

Additional Readings:

Online Learning Resources for " Machine Learning and Pattern Recognition "

- https://www.coursera.org/learn/machine-learning

Machine Learning Practical with Python, Scikit-learn, Matplotlib, TensorFlow

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Machine Learning Practical with Python, Scikit-learn, Matplotlib, TensorFlow Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP252	0-0-2	1
Type of Course:	IDC-5		
Pre-requisite(s), if any: Basic knowledge of the Statistics & Python			

Course Perspective. This course is designed to provide students with a robust foundation in the fundamentals of machine learning, covering both supervised and unsupervised learning. Through hands-on experience, students will use Python and libraries like Scikit-learn for implementing algorithms, Matplotlib for data visualization, and TensorFlow for deep learning applications. The course emphasizes real-world applications, allowing students to tackle problems in various domains such as finance, healthcare, and marketing.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	Implementing and apply various regression techniques such as Lasso, Ridge, and multiple logistic regression to analyze and interpret data, making accurate predictions.
CO 2	Designing and implementing classification algorithms to solve real-world problems, such as predicting fraudulent transactions, diabetes diagnosis, and default on credit card payments.
CO 3	Analyzing and evaluating the performance of models using techniques like principal component analysis, correlation matrices, ROC curves, k-fold validation, and subset selection.
CO 4	Analyzing and visualizing data using advanced techniques and tools such as clustering, decision trees, and graphical representations, enhancing data interpretation and decision-making processes

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Experiment List

Ex. No	Experiment Title	Mapped CO/COs
1	Lasso and Ridge regression implementation	CO4
2	Principal component analysis implementation	CO4
3	Making predictions using multiple Logistics regression	CO2, CO3
4	Implementation of correlation matrix, correlation matrix , ROC curve	CO3
5	Faudulent transaction using classification algorithm	CO2
6	Predict whether a patient have diabetes	CO2
7	Improve sales of product of a company	CO1, CO2
8	Finance - Predict whether a credit card user will default on monthly credit card payment based on annual income and monthly credit card balance	CO1, CO2
9	HR - Predict the Baseball major league player salary based on career and previous season statistics	CO1, CO2
10	Write a program to implement bootstrap	CO4
11	Write a program to implement k-fold validation	CO3
12	Write a program to implement subset selection	CO3
13	Write a program for Ridge Regression	CO4
14	Write a program for lasso regression	CO4
15	Write a program to analyze and solve zero values	CO3
16	Write a program to analyze the categorical values	CO3
17	Write a program for graphical representation of data.	CO4

18	Write a program for principal component analysis.	CO4
19	Write a program to implement clustering.	CO2
20	Write a program to implement decision tree.	CO2
21	Write a program to implement markov model.	CO2

References

1. Machine Learning by Tom M. Mitchell - McGraw Hill Education; First edition
2. Pattern Recognition and Machine Learning (Information Science and Statistics) by Christopher M. Bishop - Springer; 1st ed. 2006. Corr. 2nd printing 2011 edition
3. The Elements of Statistical Learning: Data Mining, Inference, and Prediction by
4. Trevor Hastie, Robert Tibshirani, Jerome Friedman - Springer; 2nd ed. 2009, Corr. 9th printing 2017 edition.

Additional Readings:

Online Learning Resource:

1. **Machine Learning by Andrew Ng**: Coursera Link
2. **Deep Learning Specialization by Andrew Ng**: Coursera Link
3. **Applied Data Science with Python Specialization**: Coursera Link

R PROGRAMMING FOR DATA SCIENCE AND DATA ANALYTICS LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **R Programming for Data Science and Data Analytics Lab**	**Course Code**	**L-T-P**	**Credits**
	SEC039	0-0-4	2
Type of Course:	SEC-4		
Pre-requisite(s), if any: Basic knowledge of the programming and statistics			

Course Perspective. The course is designed to equip students with the essential skills needed to manipulate, analyze, and visualize data using R. Students will learn fundamental programming concepts, data structures, and functions in R, while also exploring advanced topics such as statistical analysis, machine learning, and data visualization techniques. Through hands-on lab sessions, students will gain practical experience by working on real-world datasets, allowing them to apply theoretical knowledge to solve complex data problems. This course aims to build a solid foundation in R, preparing students for careers in data science, analytics, and related fields, where they can leverage their skills to derive insights and make data-driven decisions. The course is divided into 4 modules:

a) Getting Started with R and R Workspace

b) Basic Objects and Basic Expressions

c) Working with Basic Objects and Strings

d) Working with Data –Visualize and Analyze Data

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	Experimenting with basic R code, including creating variables, data types, and functions.
CO 2	Examining and manipulating data using R, including importing and exporting data, subsetting data, merging data sets, and cleaning data

CO 3	Building visualizations using R, including basic and advanced plots, graphs, and charts
CO 4	Examining data with the help of statistical analysis using R, including descriptive statistics, regression analysis

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Ex. No	Experiment Title	Mapped CO/COs
1.	Installation of R and Rstudio	CO1
2.	Create Matrix in R and perform following operations	CO1, CO2
3.	Create a matrix A and fill with values from 1 to 12.	CO1, CO2
4.	Create a matrix B and fill with values from 1 to 12.	CO1, CO2
5.	Find the transpose of matrix A and matrix B.	CO1, CO2
6.	Find the multiplication of matrix A and matrix B.	CO1, CO2
7.	Find the addition of matrix A and matrix B.	CO1, CO2
8.	Find the substation of matrix A and matrix B.	CO1, CO2
9.	Subsetting a Matrix.	CO1, CO2
10.	Create a matrix A and fill with values from 4 to 16	CO1, CO2
11.	Get first 2 rows	CO1, CO2
12.	Subset top 2 row and left 2 columns	CO1, CO2
13.	Subset 3 row and 2 column	CO1, CO2
14.	Write a R Program to create a list a_list that contains numbers, strings, logical value, and vectors	CO1, CO2
15.	Add names to the list a_list	CO1, CO2
16.	Add an element at the end of the list a_list	CO1, CO2

17.	Create a function calc This function will accept three arguments that include two numeric vectors x and y and one character vector type. The character vector type will define the kind s operation, the user wants to perform.	CO1, CO2
18.	Write a R program to find the numbers between 1000 and 1100 that satisfy (i ^ 2) %% 11 equals (i ^ 3) %%17, where ^ is a power operator and %% (modulo	CO1, CO2
	operator) returns the remainder of a division.	
19.	Develop a function that can behave differently according to the type of input object.	CO1, CO2
20.	Create scatter plot using more than one dataset usevarious point styles and colors.	CO1, CO3
21.	Create multi-period line plot, with mix different linetypes and create Multi Series Chart with Legend.	CO1, CO3
22.	Create bar chart, pie chart and histogram with random data	CO1, CO3

Text Books/Reference Books:

I) "R in Action: Data Analysis and Graphics with R" by Robert I. Kabacoff

II) "R for Data Science" by Hadley Wickham and Garrett Grolemund

Additional Readings:

Online Learning Resources for " R PROGRAMMING FOR DATA SCIENCE AND DATA ANALYTICS "

1. **Coursera**

   "R Programming" by Johns Hopkins University: Part of the Data Science Specialization, this course covers the basics of R programming.

   Link- https://www.coursera.org/learn/r-programming

Communication & Personality Development

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Life Skills for Professionals - II	**Course Code**	**L-T-P**	**Credits**
	AEC007	3-0-0	3
Type of Course:	AEC-2		
Contact Hours	36		
Pre-requisite(s), if any:			

Course Perspective. The course enhances public speaking and presentation skills, helps students confidently convey ideas, information & build self-reliance and competence needed for career advancement. Personality assessments like the Johari Window and Myers & Briggs Type Indicator (MBTI) provide frameworks to enhance self-understanding, helps people increase their self-awareness, understand and appreciate differences in others and apply personality insights to improve their personal and professional effectiveness. Interpersonal skills included in the course deal with important topics like communication, teamwork and leadership, vital for professional success.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Improving** public speaking and presentation abilities to confidently convey ideas and information.
CO 2	**Understanding** the framework of Communication to augment oratory skills and written English communication, professional writing, and persuasive communication.
CO 3	**Cultivating** essential soft skills required at the different workplaces.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Developing self and others**	**No. of hours: 8**

Content Summary: Self Awareness, Personality Concepts (Personality Assessments - Johari Window, Myers & Brigg), Self-Management, Self Esteem, Self-Efficacy, Interpersonal skills, mindset, grit and working in teams.

Unit Number: 2	**Title: Enhancing Reading and Writing Skills**	**No. of hours: 6**

Content Summary: Speed reading and its importance in competitive examinations, techniques for speed reading, note-taking, and critical analysis. Paragraph Writing, Essay and Summary writing, Business Letter, Email writing

Unit Number: 3	**Title: Effective Communication and Public Speaking**	**No. of hours: 7**

Content Summary: Communication Framework, barriers & overcoming these barriers, Group Discussions, Extempore & Public Speaking drills, to manage stage fright and anxiety. Structuring and organizing a presentation (Oral & PPT), Etiquettes, Grooming, Body Language and Conversation starters, TMAY.

Unit Number: 4	**Title: Career Guide and readiness**	**No. of hours: 15**

Content Summary: Cover Letter, ATS friendly resume, Elevator Pitch, Video Resume (Visume), Networking, Group Discussion, Mock Interviews. Capstone Project

Learning Experiences

1. **Interactive Lectures**: Introduce advanced life skills concepts using PPTs and real-life scenarios.

2. **Conceptual Understanding**: Cover topics like emotional intelligence, conflict resolution, and leadership.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on real-world workplace challenges.
4. **Group Discussions**: Facilitate discussions on ethics, diversity, and effective communication in professional settings.
5. **Guest Speakers**: Invite industry leaders to share insights and experiences related to life skills.
6. **Continuous Feedback**: Implement quizzes and peer reviews to assess application of life skills.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign reflective essays on personal development and career aspirations.
2. **Workshops**: Facilitate hands-on sessions for practicing negotiation, networking, and public speaking skills.
3. **Question Bank**: Provide resources for self-assessment on advanced life skills development.
4. **Online Forums**: Create platforms for discussing personal growth and professional challenges.
5. **Self-Study for Case Studies**: Encourage independent research on successful professionals and their life skills.
6. **Collaborative Projects**: Organize group projects focused on community engagement and leadership initiatives.

Textbooks

I) Aggarwal, R. S. (2014). Quantitative aptitude (Revised edition).

II) Gladwell, M. (2021). Talking to strangers.

III) Scott, S. (2004). Fierce conversations.

References

- "The 7 Habits of Highly Effective People" by Stephen R. Covey
- "Presentation Skills: The Essential Guide for Students" by Joan van Emden and Lucinda Becker

- "Emotional Intelligence: Why It Can Matter More Than IQ" by Daniel Goleman
- "Arithmetic for Competitive Examinations" by R.S. Aggarwal
- "The Art of Public Speaking" by Dale Carnegie

MINOR PROJECT-II

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Minor Project-II	**Course Code**	**L-T-P**	**Credits**
	ENSI252	---	2
Type of Course:	Proj-2		
Pre-requisite(s), if any: NA			

Duration:

The minor project will last for **three** months.

Project Requirements:

1. **Understanding of Societal Problems:**
    - Students must have a basic understanding of societal problems, the concerned domain, and relevant issues.
2. **Critical Thinking and Problem Formulation:**
    - Students are expected to think critically about formulated problems and review existing solutions.
3. **Data Gathering and ETL Activities:**
    - Students should gather relevant data and perform ETL (Extract, Transform, Load) activities to prepare the data for analysis.
4. **Innovation and Entrepreneurship Focus:**
    - Students should develop innovative ideas or entrepreneurial solutions to address the identified problems.

5. **Implementation (Optional):**
    - While implementation of the proposed solutions is encouraged, it is not strictly required. The focus should be on idea development.

Guidelines:

1. **Project Selection:**
    - Choose a societal problem relevant to the field of computer science and engineering.
    - Ensure the problem is specific and well-defined.
2. **Literature Review:**
    - Conduct a thorough review of existing literature and solutions related to the problem.
    - Identify gaps in existing solutions and potential areas for further investigation.
3. **Data Gathering and ETL:**
    - Collect relevant data from various sources.
    - Perform ETL activities to clean, transform, and load the data for analysis.
4. **Analysis and Critical Thinking:**
    - Analyze the problem critically, considering various perspectives and implications.
    - Evaluate the effectiveness and limitations of current solutions.
5. **Innovation and Idea Development:**
    - Develop innovative ideas or entrepreneurial solutions to address the identified problem.
    - Focus on the feasibility, impact, and potential of the proposed solutions.
6. **Documentation:**
    - Document the entire process, including problem identification, literature review, data gathering, ETL activities, analysis, and ideas.
    - Use appropriate formats and standards for documentation.
7. **Presentation:**
    - Prepare a presentation summarizing the problem, existing solutions, data analysis, and proposed ideas.
    - Ensure the presentation is clear, concise, and well-structured.

Evaluation Criteria for Minor Project (Out of 100 Marks):

1. **Understanding of Societal Problems (15 Marks):**

- Comprehensive understanding of the problem: 15 marks
- Good understanding of the problem: 12 marks
- Basic understanding of the problem: 9 marks
- Poor understanding of the problem: 5 marks
- No understanding of the problem: 0 marks

2. **Critical Thinking and Analysis (20 Marks):**
    - Exceptional critical thinking and analysis: 20 marks
    - Good critical thinking and analysis: 15 marks
    - Moderate critical thinking and analysis: 10 marks
    - Basic critical thinking and analysis: 5 marks
    - Poor critical thinking and analysis: 0 marks
3. **Data Gathering and ETL Activities (20 Marks):**
    - Comprehensive and effective ETL activities: 20 marks
    - Good ETL activities: 15 marks
    - Moderate ETL activities: 10 marks
    - Basic ETL activities: 5 marks
    - Poor ETL activities: 0 marks
4. **Innovation and Idea Development (25 Marks):**
    - Highly innovative and feasible ideas: 25 marks
    - Good innovative ideas: 20 marks
    - Moderate innovative ideas: 15 marks
    - Basic innovative ideas: 10 marks
    - Poor innovative ideas: 5 marks
    - No innovative ideas: 0 marks
5. **Documentation Quality (10 Marks):**
    - Well-structured and detailed documentation: 10 marks
    - Moderately structured documentation: 7 marks
    - Poorly structured documentation: 3 marks
    - No documentation: 0 marks
6. **Presentation (10 Marks):**
    - Clear, concise, and engaging presentation: 10 marks
    - Clear but less engaging presentation: 7 marks

- Somewhat clear and engaging presentation: 3 marks
- Unclear and disengaging presentation: 0 marks

Total: 100 Marks

Course Outcomes:

By the end of this course, students will be able to:

1. **Understand Societal Issues:**
   - Demonstrate a basic understanding of societal problems and relevant issues within the concerned domain.
2. **Critical Thinking:**
   - Think critically about formulated problems and existing solutions.
3. **Data Management:**
   - Gather relevant data and perform ETL activities to prepare the data for analysis.
4. **Innovation and Entrepreneurship:**
   - Develop innovative ideas or entrepreneurial solutions to address identified problems.
5. **Literature Review:**
   - Conduct comprehensive literature reviews and identify gaps in existing solutions.
6. **Documentation:**
   - Document findings and analysis in a well-structured and appropriate format.
7. **Presentation Skills:**
   - Present findings and analysis effectively, using clear and concise communication skills.
8. **Problem Analysis:**
   - Analyze problems from various perspectives and evaluate the effectiveness of existing solutions.
9. **Professional Development:**
   - Develop skills in research, analysis, documentation, and presentation, contributing to overall professional growth.

Learning Experiences

Classroom Learning Experience

1. **Project Kickoff**: Introduce project objectives and expectations through orientation

sessions.

2. **Research Methodology**: Cover essential techniques for conducting research and project planning.
3. **Problem-Solving Sessions**: Conduct workshops focused on overcoming project-related challenges.
4. **Progress Presentations**: Facilitate sessions for students to present their project updates and receive feedback.
5. **Group Collaboration**: Encourage teamwork to enhance project development and idea exchange.
6. **Continuous Feedback**: Implement peer reviews and mentor check-ins to assess progress and learning.

Outside Classroom Learning Experience

1. **Independent Research**: Assign tasks that require in-depth research and exploration of project topics.
2. **Hands-On Implementation**: Facilitate practical application of project concepts in real-world scenarios.
3. **Documentation**: Encourage students to maintain detailed project logs and documentation.
4. **Online Collaboration Tools**: Create platforms for students to communicate and share resources effectively.
5. **Self-Assessment**: Provide tools for students to evaluate their contributions and project outcomes.
6. **Final Presentation**: Organize sessions for students to present their completed projects to peers and faculty.

COMPETITIVE CODING BOOTCAMP-II

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: COMPETITIVE CODING BOOTCAMP-II	**Course Code**	**L-T-P**	**Credits**
		2-0-0	0
Type of Course:	AUDIT-2		
Contact Hours	30		
Version			

Course Outcomes

CO1	**Understanding** fundamental tree structures, including AVL trees, and their balancing mechanisms.
CO2	**Applying** graph representations (adjacency matrix and adjacency list) to solve basic graph traversal problems.
C	**Implementing** shortest path algorithms such as Dijkstra's algorithm and

O3	Bellman-Ford.
CO4	**Exploring** dynamic programming concepts, including memoization and tabulation, to solve classic problems.

<table>
<tr><td>Unit Number: 1</td><td>Title: Object-Oriented Programming Concepts</td><td>No. of hours: 8</td></tr>
<tr><td colspan="3">Content:
OOP Basics: Encapsulation, Inheritance, Polymorphism, Class Design and Object Creation
C++ OOP Concepts: Classes and Objects, Constructors/Destructors, Operator Overloading, Inheritance, Virtual Functions
Java OOP Concepts: Classes and Objects, Constructors, Method Overloading, Inheritance, Polymorphism, Abstract Classes, Interfaces
Python OOP Concepts: Classes and Objects, Constructors, Method Overloading (via default arguments), Inheritance, Polymorphism, Multiple Inheritance</td></tr>
<tr><td>Unit Number: 2</td><td>Title: Linked Lists, Stacks and Queues</td><td>No. of hours: 8</td></tr>
<tr><td colspan="3">Content:
Linked Lists
▪ Singly and doubly linked lists: Creation, insertion, deletion, traversal.
▪ Key Problems: Reversing a linked list (iterative and recursive), detecting cycles using Floyd’s cycle-finding algorithm.
Stacks and Queues :
▪ Stack operations: Push, pop, top, isEmpty.
▪ Queue operations: Enqueue, dequeue, front, isEmpty.
▪ Applications: Parentheses matching, queue-based problems (LeetCode challenge -</td></tr>
</table>

sliding window problems).		
Unit Number: 3	**Title: Sorting & Searching**	**No. of hours: 8**

Content

Basic Sorting Algorithms

- Implementing Bubble Sort, Selection Sort, Insertion Sort.
- Understanding the time complexities and use cases of each algorithm.
- Key Problems: Sorting small arrays, finding the median, custom sorting based on conditions (frequent LeetCode challenge).

Advanced Sorting Algorithms

- Implementing Merge Sort, Quick Sort, Heap Sort.

Binary Search

- Implementing binary search for sorted arrays.
- Applications: Finding an element in a sorted array, finding the position to insert an element, LeetCode challenges like searching for ranges.

Unit Number: 4	**Title: Trees**	**No. of hours: 6**

Content:

Basic Tree Concepts

- Introduction to tree terminology and operations.
- Tree Traversals: Preorder, inorder, postorder.
- Key Problems: Printing all elements in a tree, finding the depth of a tree.

Binary Trees

- Basic operations on binary trees: Insertion, deletion, searching.
- Key Problems: Finding the height of a binary tree, counting leaf nodes, lowest common ancestor (common LeetCode challenges).

Binary Search Trees

Understanding BST properties: Every left subtree is smaller, and every right subtree

is larger.

Learning Experiences:

Classroom Learning Experience

1. **Advanced Lectures**: Introduce complex algorithms and techniques using PPTs and coding demonstrations.
2. **Algorithm Workshops**: Cover advanced topics like dynamic programming, graph algorithms, and optimization strategies.
3. **Intensive Problem-Solving Sessions**: Conduct in-class exercises focused on challenging competitive coding problems.
4. **Mock Contests**: Organize timed coding competitions to simulate real contest environments.
5. **Group Strategy Discussions**: Facilitate discussions on effective problem-solving strategies and approaches.
6. **Continuous Feedback**: Implement performance assessments and code reviews to enhance skills.

Outside Classroom Learning Experience

1. **Practice Assignments**: Assign challenging coding problems for independent practice from various platforms.
2. **Online Competitions**: Encourage participation in external coding contests and hackathons.
3. **Question Bank**: Provide a repository of advanced practice problems for self-assessment.
4. **Online Collaboration**: Create forums for students to discuss problems and share solutions.
5. **Self-Study Resources**: Recommend books and online courses focused on advanced algorithms and techniques.
6. **Collaborative Projects**: Organize group projects that involve developing coding solutions for complex problems.

Textbooks:

- "Introduction to Algorithms" by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein
- "Data Structures and Algorithms Made Easy" by Narasimha Karumanchi

Online References:

1. **GeeksforGeeks**:
   - Offers articles on advanced data structures like self-balancing trees, segment trees, tries, and more[1].
   - Link to GeeksforGeeks
2. **Coursera**:
   - Various data structures and algorithms courses available online.
   - Examples include "Data Structures and Algorithms" from the University of California San Diego and "Algorithms, Part I" from Princeton University[3].
   - Link to Coursera
3. **Princeton University References**:
   - Provides a list of seminal papers and advanced resources.
   - Includes textbooks like "Algorithms, 4th Edition" by Robert Sedgewick and Kevin Wayne[4].

Lab Experiments

Problem Statement	Mapped COs
Object-Oriented Programming Concepts	
1. Design a Parking Lot System using OOP concepts (Classes, Objects,	CO1

Problem Statement	Mapped COs
Inheritance, Polymorphism).	
2. Implement a Student Management System with Classes and Objects.	CO1
3. Create a Banking System with Constructors and Destructors.	CO1
4. Implement Method Overloading and Overriding in a chosen language.	CO1
5. Demonstrate Multiple Inheritance with a practical example.	CO1
6. Design a Library Management System with OOP principles.	CO1
7. Use Virtual Functions to implement polymorphism.	CO1
8. Implement Abstract Classes and Interfaces for a Payment System.	CO1
9. Create a simple calculator with Operator Overloading.	CO1
10. Build a Polymorphic class hierarchy (e.g., Shapes) to showcase polymorphism.	CO1
Linked Lists, Stacks, and Queues	
11. Reverse a Linked List (Iterative and Recursive).	CO2
12. Detect a cycle in a Linked List using Floyd's Cycle-Finding Algorithm.	CO2
13. Implement basic operations on a Singly Linked List (Insertion, Deletion).	CO2
14. Implement and traverse a Doubly Linked List.	CO2
15. Implement Stack operations (Push, Pop, Top) using arrays or linked lists.	CO2
16. Implement Queue operations (Enqueue, Dequeue, Front) using arrays or linked lists.	CO2
17. Solve the Parentheses Matching problem using Stack.	CO2
18. Implement Sliding Window Maximum using Deque.	CO2

Problem Statement	Mapped COs
19. Check for balanced parentheses using Stack.	CO2
20. Design a Circular Queue using linked list or array.	CO2
Sorting & Searching	
21. Implement Bubble Sort and analyze its time complexity.	CO3
22. Implement Merge Sort to sort an array of integers.	CO3
23. Find the Kth largest element in an array using Quick Sort.	CO3
24. Perform Binary Search to find an element in a sorted array.	CO3
25. Implement Heap Sort to sort a list of elements.	CO3
26. Find the position to insert an element in a sorted array using Binary Search.	CO3
27. Implement a custom sort based on frequency of elements.	CO3
28. Compare sorting results using Insertion Sort and Bubble Sort.	CO3
Trees	
29. Perform Preorder, Inorder, and Postorder Traversal on a Binary Tree.	CO4
30. Find the Lowest Common Ancestor in a Binary Search Tree.	CO4

Problem Statement	Mapped COs
Trees	
31. Implement an algorithm to check if a Binary Tree is balanced.	CO4
32. Determine if two Binary Trees are identical.	CO4

Problem Statement	Mapped COs
33. Find the maximum path sum in a Binary Tree.	CO4
34. Convert a Binary Search Tree to a Greater Tree (where each node's value is replaced by the sum of all greater values).	CO4
35. Count the number of nodes in a complete Binary Tree.	CO4
36. Flatten a Binary Tree to a linked list using preorder traversal.	CO4
37. Serialize and deserialize a Binary Tree.	CO4
38. Find the diameter of a Binary Tree (the longest path between any two nodes).	CO4
39. Check if a Binary Tree is a subtree of another Binary Tree.	CO4
40. Find the level order traversal of a Binary Tree (Breadth-First Search).	CO4

Semester: 5

THEORY OF COMPUTATION

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Theory of Computation	**Course Code**	**L-T-P**	**Credits**
	ENCS301	3-1-0	4
Type of Course:	Major-17		
Pre-requisite(s), if any: NA			

Course Perspective. The course provides a comprehensive foundation in the theoretical aspects of computer science, essential for understanding the underlying principles of various computational processes and languages. This course delves into the formalization and analysis of computation, encompassing finite automata, pushdown automata, context-free grammars, Turing machines, and the Chomsky hierarchy. The course is divided into 4 modules:

a) Introduction to Finite Automata
b) Pushdown Automata and Context-Free Languages
c) Chomsky Hierarchy and Turing Machines
d) Code Generation and Optimization

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Remembering** the fundamental concepts and terminology of automata theory.
CO 2	**Understanding** the relationships and equivalences between various computational models.

CO 3	**Applying** conversion techniques between different forms of automata and grammars.
CO 4	**Analyzing** the properties and limitations of formal languages using theoretical tools.
CO 5	**Evaluating** the decidability and complexity of computational problems.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Introduction to Finite automata**	**No. of hours: 10**

Content:

Finite Automata: Review of Automata, Description of Finite automata, representation of FA,

Deterministic Finite Automata(DFA), Non-deterministic Finite Automata(NFA),Equivalence of NFA and DFA Finite Automata with Epsilon Transitions, Minimization of Deterministic Finite Automata

Finite Automata with output: - Moore machine and Mealy Machine, Conversion of Moore machine to Mealy Machine & Vice-Versa

Applications of Finite Automata

Unit Number: 2	**Title: Regular Expression and Languages**	**No. of hours: 10**

Content:

Regular Expressions: Introduction, Identities of Regular Expressions, Arden’s theorem state and prove

Finite Automata and Regular Expressions: Converting from DFA's to Regular Expressions and Vive-Versa

Pumping Lemma for Regular Sets: Introduction, Applications of the pumping lemma-Proving languages not to be regular, Closure properties of regular languages

Introduction to Formal languages: Definition of a Grammar, Derivations and the Language Generated by a Grammar, Chomsky Classification of Languages

Unit Number: 3	**Title: Context-Free Languages and Pushdown Automata (PDA)**	**No. of hours: 12**

Content:

Context Free Grammar (CFG): Properties of context free grammar, Derivations using a grammar, Parse Trees, Ambiguity in context free grammar

Simplification of Context Free grammar: Reduced grammar, Removal of useless Symbols and unit production

Normal Forms of CFG: Chomsky Normal Form (CNF), Greibach Normal Form (GNF)

Pumping lemma for CFG.

Push down Automata (PDA): Definition, acceptance by PDA, Types of PDA: Deterministic PDA, Non-Deterministic PDA

Equivalence of CFL and PDA, interconversion

Unit Number: 4	**Title: Turing Machine and Undecidability**	**No. of hours: 8**

Content Summary:

Turing Machines: Definition, types, and language acceptors, Design of Turing Machines

Universal Turing Machine and its implications

Decidability and Undecidability

Halting problem of Turing Machine, Post-Correspondence Problem.

Properties of Recursive and Recursively Enumerable Languages

`

Learning Experience

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key concepts in the theory of computation using PPTs and examples.
2. **Conceptual Understanding**: Cover topics like automata theory, formal languages, and Turing machines.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on designing automata and proving language properties.
4. **Case Studies**: Analyze real-world applications of computational theory in computer science.
5. **Group Discussions**: Facilitate discussions on complexity theory and computability.
6. **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding of theoretical concepts.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign take-home projects that require applying theoretical concepts to practical problems.
2. **Lab Projects**: Facilitate hands-on tasks involving simulations of automata and formal languages.
3. **Question Bank**: Provide practice problems and resources for self-assessment on computation theory topics.
4. **Online Forums**: Create platforms for discussing challenges and sharing solutions related to computation theory.
5. **Self-Study for Case Studies**: Encourage independent research on advancements and applications in computation theory.
6. **Collaborative Projects**: Organize group projects focused on exploring complex theoretical concepts and their implications.

Text Books/Reference Books:

III) Hopcroaft J.E., Ullman, J.D., and Rajiv Motwani, 2001, Introduction to Automata Theory, Language & Computations, 3rdEd.,AW.

IV) Mishra K.L.P.& N. Chandrasekaran, 2000, Theory of Computer Science Automata, Languages and Computation,5th Ed. , 2000, PHI

V) H.R. Lewis and C.H. Papadimitriou, “Elements of the theory of Computation”, Second Edition, Pearson Education.

VI) Peter Linz, 2001, Introduction to formal Languages & Automata, 3rd Ed., NarosaPubl.

VII) J. Martin, “Introduction to Languages and the Theory of computation” Third Edition, Tata Mc Graw Hill.

Additional Readings:

Online Learning Resources for "Theory of Computation"

2. **NPTEL - Theory of Computation**
    - Online course by IITs on the fundamentals of the theory of computation.
    - Link: NPTEL-Theory of Computation
3. **Coursera - Automata Theory by Stanford University**
    - This course covers the fundamentals of automata theory, including finite automata, regular expressions, and Turing machines.
    - Link: Coursera - Automata Theory
4. **MIT OpenCourseWare - Theory of Computation**
    - Advanced course materials from MIT covering various topics in the theory of computation.
    - Link: MIT OpenCourseWare - Theory of Computation

OPERATING SYSTEMS

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: OPERATING SYSTEMS	**Course Code**	**L-T-P**	**Credits**
	ENCS303	3-1-0	4
Type of Course:	Major-18		

Pre-requisite(s), if any: Basics of programming

Course Perspective. This course provides a comprehensive introduction to the fundamental principles and practices of operating systems. It covers essential concepts such as process management, memory management, file systems, and I/O systems, as well as more advanced topics like distributed operating systems and concurrent systems. Through this course, students will gain a deep understanding of how operating systems function, how they manage hardware resources, and how they provide services to applications. The course also emphasizes practical skills in implementing and managing operating system components and handling challenges such as process synchronization, deadlocks, and system security. By the end of the course, students will be well-equipped to apply these concepts in designing and optimizing operating systems in various computing environments. The course is divided into 4 modules:

a) Introduction to Operating Systems and Process
b) Memory & File Management
c) Process Synchronization, Deadlocks & I/O Systems

d) Distributed Operating Systems & Concurrent Systems

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the fundamental concepts of operating systems, including their structure and types.
CO 2	**Analyzing** process scheduling algorithms and their impact on system performance.
CO 3	**Implementing** and manage memory allocation, paging, and virtual memory techniques.
CO 4	**Examining** process synchronization mechanisms and handle deadlocks in an operating system environment.
CO 5	**Developing** distributed operating systems and concurrent systems with a focus on fault tolerance and recovery mechanisms.

Course Outline:

Unit Number: 1	**Title: Introduction to Operating System, Process and CPU Scheduling**	**No. of hours: 10**

Introduction: Definition, Role, Types of Operating System, Batch Systems, multi programming, time–sharing, parallel, distributed and real-time systems, Operating system structure, Operating system components and services, System calls, System programs, Virtual machines.

Processes: Process Concept, Process Scheduling, Operation on Processes, Cooperating Processes, Threads.

CPU Scheduling: Basic Concepts, Scheduling Criteria, Scheduling Algorithms, Multiple Processor Scheduling, Real-Time Scheduling.

Unit Number:	**Title: Threads, Synchronization, Deadlock and Memory Management**	**No. of hours: 10**

2

Threads: overview, Benefits of threads, User and kernel threads, Multithreaded Models, Precedence Graph, Fork-Join, Cobegin-Coend construct.

Inter-process Communication and Synchronization: Background, The Critical-Section Problem, Synchronization Hardware, Semaphores, Classical Problems of Synchronization, Critical Regions, Monitors, Message Passing.

Deadlocks: System Model, Deadlock Characterization, Methods for Handling Deadlocks, Deadlock Prevention, Deadlock Avoidance, Deadlock Detection, Recovery from Deadlock.

Memory Management: Background, Logical vs. Physical Address space, swapping, Contiguous allocation, Paging, Segmentation, Segmentation with Paging.

Unit Number: 3	**Title: Virtual Memory, Device Management and Secondary-Storage Structure**	**No. of hours: 10**

Virtual Memory: Demand Paging and its performance, Page-replacement Algorithms, Allocation of Frames, Thrashing, page size and other Considerations, Demand Segmentation.

Device Management: Techniques for Device Management, Dedicated Devices, Shared Devices, Virtual Devices, Independent Device Operation, Buffering, Device Allocation Consideration.

Secondary-Storage Structure: Disk Structure, Disk Scheduling, Disk Management, Swap Space Management, Disk Reliability.

Unit Number: 4	**Title: File-System Interface, implementation and Security**	**No. of hours: 10**

File-System Interface: File Concept, Access Methods, Directory Structure.

File-System Implementation: Introduction, File-System Structure, Basic File System, Allocation Methods, Free-Space Management, Directory Implementation.

Security: Security problems, Goals of protection, Access matrix, Authentication, Program threats, System threats, Intrusion detection.

Learning Experiences

- **Interactive Lectures**: Introduce key concepts of operating systems using PPTs and real-world examples.
- **Conceptual Understanding**: Cover topics like process management, memory management, and file systems.
- **Problem-Solving Sessions**: Conduct in-class exercises focused on system calls and scheduling algorithms.
- **Case Studies**: Analyze real-world operating systems and their architectures.
- **Group Work**: Collaborate on projects that involve designing and implementing operating system components.
- **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding of operating system principles.

- **Outside Classroom Learning Experience**
- **Theory Assignments**: Assign take-home projects requiring application of operating system concepts to practical scenarios.
- **Lab Projects**: Facilitate hands-on tasks involving system programming and OS simulations.
- **Question Bank**: Provide practice problems and resources for self-assessment on operating system topics.
- **Online Forums**: Create platforms for discussing operating system challenges and sharing solutions.
- **Self-Study for Case Studies**: Encourage independent research on current trends and technologies in operating systems.
- **Collaborative Projects**: Organize group projects focused on solving real-world problems using operating system concepts.

Textbooks

1. Operating System Concepts, Abraham Silberschatz, Peter Baer Galvin, Greg Gagne, Wiley
2. Modern Operating Systems, Andrew S. Tanenbaum and Herbert Bos, Pearson, 4th Edition, 2014.

3. Operating Systems: Internals and Design Principles, William Stallings, Pearson, 9th Edition, 2017.
4. Operating Systems: Three Easy Pieces, Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau Arpaci-Dusseau Books, 1st Edition, 2018

References

I) MukeshSinghal and N. G. Shivaratri, "Advanced Concepts in Operating Systems", McGrawHill, 2000

II) Abraham Silberschatz, Peter B. Galvin, G. Gagne, "Operating System Concepts", Sixth Addison Wesley Publishing Co., 2003.

III) Andrew S. Tanenbaum, "Modern Operating Systems", Second Edition, Addison Wesley, 2001.

IV) Tannenbaum, "Operating Systems", PHI, 4th Edition.

Additional Readings:

Online Learning References :

I) **MIT OpenCourseWare - Operating System Engineering**

   a. Advanced course materials from MIT covering various topics in operating system design and implementation.
   b. Link: MIT OpenCourseWare - Operating System Engineering

II) **NPTEL - Operating System by IITs**

   a. Online course by IITs providing in-depth coverage of operating system principles and practices.
   b. Link: NPTEL - Operating System

OPERATING SYSTEM LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: OPERATING SYSTEMS LAB	**Course Code**	**L-T-P**	**Credits**
	ENCS351	0-0-2	1
Type of Course:	Major-19		

Pre-requisite(s), if any: Basics of programming

Proposed Lab Experiments

Defined Course Outcomes

COs	
CO 1	**Implementing** and analyze process creation, management, and CPU scheduling algorithms, demonstrating the ability to simulate an operating system environment.
CO 2	**Developing** and evaluate multithreaded applications, demonstrating synchronization, deadlock handling, and memory management techniques.
CO 3	**Simulating** virtual memory management, device management, and disk scheduling algorithms, showcasing the application of operating system concepts.
CO 4	**Designing** and implement secure file systems, demonstrating file operations, directory management, and access control mechanisms.

List of Experiments

Experiment Title	Mapped CO/COs
Implement a program that simulates system calls for basic operations such as process creation, file manipulation, and device management. Demonstrate how system calls interact with the operating system components and services.	CO1
Develop a process scheduling simulation that demonstrates different CPU scheduling algorithms (FCFS, SJF, Round Robin, Priority Scheduling). Compare the performance of each algorithm based on scheduling criteria such as turnaround time, waiting time, and response time.	CO1
Create a multi-threaded application to illustrate process operations, including creation, termination, and inter-process communication. Implement thread management to demonstrate the concept of cooperating processes and the benefits of threading.	CO1
Implement a multi-threaded program to demonstrate the benefits of threads over single-threaded processes. Use different multithreading models such as user-level and kernel-level threads and simulate various thread operations.	CO2
Design and implement solutions for classical synchronization	CO

problems such as the Producer-Consumer problem, Readers-Writers problem, and Dining Philosophers problem using semaphores, critical regions, and monitors.	2
Create a simulation to detect and handle deadlocks in a system. Implement deadlock prevention, avoidance, and detection algorithms. Demonstrate recovery from deadlock scenarios.	CO 2
Develop a memory management simulator that demonstrates different memory allocation techniques such as contiguous allocation, paging, and segmentation. Implement swapping and address translation between logical and physical address spaces.	CO 2
Implement a demand paging system to simulate virtual memory management. Evaluate the performance of different page-replacement algorithms (FIFO, LRU, Optimal) and analyze the effects of thrashing.	CO 3
Create a simulation for device management that includes buffering, device allocation, and handling dedicated, shared, and virtual devices. Demonstrate techniques for independent device operation and management.	CO 3
Develop a disk scheduling simulator to compare the performance of different disk scheduling algorithms (FCFS, SSTF, SCAN, C-SCAN). Implement disk management techniques and swap space management.	CO 3
Implement a file system simulator to demonstrate different file access methods (sequential, direct, indexed). Design a directory structure and simulate file operations such as creation, deletion, reading, and writing.	CO 4
Develop a program to simulate file system implementation techniques, including different file allocation methods (contiguous, linked, indexed) and free-space management techniques. Implement a basic file system and directory structure.	CO 4

NATURAL LANGUAGE PROCESSING

<table>
<tr><td>Department:</td><td colspan="3">Bachelor of Technology (CSE) with specialization in AI & ML</td></tr>
<tr><td rowspan="2">Course Name:
Natural languageprocessing</td><td>Course Code</td><td>L-T-P</td><td>Credits</td></tr>
<tr><td>ENSP302</td><td>4-0-0</td><td>4</td></tr>
<tr><td>Type of Course:</td><td colspan="3">IDC-6</td></tr>
<tr><td colspan="4">Pre-requisite(s), if any: Strong programming skills, particularly in Python.</td></tr>
</table>

Course Perspective. The course covers fundamental concepts and techniques used in the processing and analysis of natural language data. Students will explore key topics such as text preprocessing, tokenization, syntactic parsing, semantic analysis, and machine translation. Advanced topics may include deep learning approaches for NLP, sentiment analysis, and language generation. Through hands-on projects and real-world applications, students will gain practical experience in building NLP models and using NLP libraries and tools. The course aims to equip students with the skills necessary to develop applications that can understand, interpret, and generate human language, preparing them for careers in data science, artificial intelligence, and related fields. The course is divided into four modules:

a) Introduction to NLP

b) Text Representation

c) Information Extraction

d) NLP for social media

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	Understanding of the fundamental concepts and approaches to Natural Language Processing, including the ability to build and implement NLP models through the eight steps of model development.
CO 2	Analyzing and applying various text representation techniques, such as Bag of Words, TF-IDF, and word embeddings, to effectively transform and visualize textual data for classification tasks.
CO 3	Implementing information extraction techniques, including named entity recognition and relationship extraction, and develop chatbots using dialog systems and tools like Rasa NLU.
CO 4	Evaluating the application of NLP in social media and e-commerce, addressing challenges such as sentiment analysis, identifying misinformation, and optimizing search and recommendation systems.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Introduction to NLP**	**No. of hours: 8**
Content Summary: Natural Language Processing in real world, What is language, Approached to NLP, **Build NLP model:** Eights Steps for building NLP Model, Web Scrapping		
Unit Number: 2	**Title: Text Representation**	**No. of hours: 8**
Content Summary: Basic Vectorization, One-Hot Encoding, Bag of Words, Bag of N Grams, TF-IDF, Pre-trained Word Embedding, Custom Word Embeddings, Vector Representations via averaging, Doc2Vec Model, Visualizing Embeddings using TSNW and Tensorbaord **Text Classification:** Application of Text Classification, Steps for building textclassification system, Text classification using Naïve Bayes Classifier, Logistic Regression, and Support Vector Machine, Neural embedding for Text Classification, text classification using deep learning, interpret text classification model		

<table>
<tr><td>Unit Number: 3</td><td>Title: Information Extraction</td><td>No. of hours: 8</td></tr>
<tr><td colspan="3">Content Summary:
Applications of Information Extraction, Processes for Information Extraction. Key phrase Extraction, Named Entity Recognition, Disambiguation and linking of named entity, Relationship extraction
Chatbot: Real life applications of chatbot, Chatbot Taxonomy, Dialog Systems, Process of building a dialog, Components of Dialog System, End to End Approach, Rasa NLU</td></tr>
<tr><td>Unit Number: 4</td><td>Title: NLP for social media</td><td>No. of hours: 8</td></tr>
<tr><td colspan="3">Content Summary:
Application of NLP in social media, challenges with social media, Natural Language Processing for Social Data, Understanding Twitter Sentiments, Identifying memes and Fake News
NLP for E-Commerce: E-commerce catalog, Search in E-Commerce, How to buildan e-commerce catalog, Review and Sentiment Analysis, Recommendations for E-Commerce</td></tr>
</table>

Learning Experience

1. **Interactive Lectures:**
   - Deliver lectures on foundational NLP concepts, model-building processes, and applications in real-world scenarios, using slides and video demonstrations.
2. **Hands-on Coding Workshops:**
   - Conduct practical sessions where students write and run Python code for various text representation techniques (e.g., Bag of Words, TF-IDF) and information extraction tasks.
3. **Group Discussions:**
   - Facilitate discussions on the ethical considerations of NLP, focusing on topics like bias in algorithms and the impact of misinformation in social media.
4. **Project-Based Learning:**
   - Assign team projects to develop chatbots or text classification systems, guiding students through the process of gathering data, preprocessing, and model implementation.
5. **Case Study Analyses:**
   - Examine case studies of successful NLP applications (e.g., sentiment analysis on Twitter) to identify methodologies and outcomes.

6. **Peer Presentations:**
    - Have students present specific NLP techniques or their project progress, promoting knowledge sharing and collaborative learning.
7. **Quizzes and Assessments:**
    - Conduct regular quizzes to test understanding of key concepts and methodologies, providing immediate feedback to reinforce learning.

Outside Classroom Learning

1. **Independent Research Assignments:**
    - Assign topics for students to explore advancements in NLP (e.g., transformer models), culminating in a presentation or report.
2. **Online Courses and MOOCs:**
    - Encourage enrollment in supplementary online courses focusing on specific NLP tools or frameworks, such as Coursera or edX.
3. **Self-Directed Projects:**
    - Allow students to choose datasets for analysis and classification tasks, applying learned methods and techniques, and presenting findings to the class.
4. **Discussion Forums:**
    - Set up online discussion boards (e.g., on platforms like Slack or Discord) where students can discuss NLP-related topics, share resources, and collaborate on ideas.

References

1. Natural Language Processing with Python by Steven Bird, Ewan Klein and Edward Loper
2. Foundations of Statistical Natural Language Processing by Christopher Manning and Hinrich Schütze

NATURAL LANGUAGE PROCESSING LAB

Department:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Natural Language processing Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP352	0-0-2	1
Type of Course:	IDC-7		
Pre-requisite(s), if any: Basics of programming			

Proposed Lab Experiments

Course Outcomes (Cos)

COs	Statements
CO 1	**Developing** proficiency in implementing various NLPtechniques and algorithms for text preprocessing, feature extraction, and linguistic analysis.
CO 2	**Applying** machine learning and deep learning models to solve real-world NLPtasks such as text classification, sentiment analysis, and named entity recognition.
CO 3	**Designing and building** end-to-end NLP applications, including chatbots orlanguage translation systems, by integrating different NLP components and models.
CO 4	**Evaluating and assessing** the performance of NLP models using appropriate metrics and techniques, and optimize models for enhanced accuracy and efficiency.

Ex. No	Experiment Title	Mapped CO/COs
1.	Write a program to scrap website.	CO1
2.	Write a program to inspect website using developer tool.	CO1
3.	Write a program to request permission to scrap website.	CO1
4.	Write a program to inspect H1 element	CO1
5.	Write a program to inspect table element	CO1

6.	Write a program to create column list	CO1
7.	Write a program to clean column list	CO1
8.	Write a program to word tokenization	CO1
9.	Write a program to implement RegEx for word tokenization	CO1
10.	Write a program to implement stopwords	CO1
11.	Write a program to implement LSTM	CO2, CO3, CO4

BIG DATA ANALYSIS WITH SCALA AND SPARK LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **Big Data Analysis withScala and Spark Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP359	0-0-4	2
Type of Course:	IDC-8		
Pre-requisite(s), if any: NA			

Course Perspective. The course covers the fundamental principles of big data processing and the capabilities of Spark's ecosystem. Students will learn to write efficient and scalable code in Scala and utilize Spark's powerful APIs for distributed data processing, including Spark SQL, DataFrames, and Spark MLlib for machine learning. Hands-on projects and practical exercises will allow students to apply concepts to real-world datasets, performing tasks such as data cleaning, transformation, and advanced analytics. The course aims to equip students with the skills necessary to design and implement robust big data solutions, preparing them for careers in data engineering, data science, and related fields where big data technologies are crucial. The course is divided into five modules:

a) Introduction to Big Data

b) Hadoop and HDFS

c) Hive and Pig

d) Map Reduce

e) Scala and Spark

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the vision of Big Data from global context.
CO 2	To **understanding** and apply Hadoop in Market perspective of Big Data.
CO 3	**Applying** and **analysing** architecture and APIs with use of Devices,Gateways and Data Management in Big data.
CO 4	To **evaluating** the application of Big Data in Industrial and Commercial Building Automation, evaluating Big Data performance using MapReduceand Real-World Design Constraints.
CO 5	**Building** and **creating** state of the art architecture in Big Data. Creating projects and research activities based on Pig, Hive, Pig Latin.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

<table>
<tr><td>Unit Number: 1</td><td>Title: Introduction to Big Data</td><td>No. of hours: 4</td></tr>
<tr><td colspan="3">Content Summary:
Develop an understanding of the complete open-source Hadoop ecosystem and its near term future directions, compare and evaluate the major Hadoop distributions and their ecosystem components both their strengths and their limitations, hands-on experience with key components of various big data ecosystem components and roles in building a complete big data, Future of Big Data. Knowledge of data, How touse Big insight</td></tr>
<tr><td>Unit Number: 2</td><td>Title: Hadoop and HDFS</td><td>No. of hours: 8</td></tr>
</table>

Content Summary:
Why Hadoop? What is Hadoop? Hadoop vs RDBMS, Hadoop vs Big Data, Types of data, Brief history of Hadoop, Problems with traditional large-scale systems, Requirements for a new approach, Anatomy of a Hadoop cluster . Concepts &Architecture, Data Flow (File Read , File Write), Fault Tolerance, Shell Commands,
Java Base API, Data Flow Archives, Coherency, Data Integrity, Role of Secondary NameNode, Zookeeper

Unit Number: 3	Title: Hive and Pig	No. of hours: 8
Content Summary: List the characteristics of representative data file formats including flat/text files CSV XML JSON and YAML, Architecture, Installation, Configuration, Hive vs RDBMS, Tables, DDL & DML, Partitioning & Bucketing, Hive Web Interface, Why Pig, Use case of Pig, Pig Components, Data Model, Pig Latin. Implementation of Real-world case study Using Real Data. List the characteristics of representative data file formats including flat/text files CSV XML JSON and YAML.		
Unit Number: 4	**Title: Map Reduce**	**No. of hours: 8**
Content Summary: Describe the MapReduce model v1 • List the limitations of Hadoop 1 and MapReduce 1 • Review the Java code required to handle the Mapper class the • Reducer class and the program driver needed to access MapReduce • Describe the YARN model • Compare Hadoop 2/YARN with Hadoop 1 ,• Understand the nature and purpose of Apache Spark in the Hadoop ecosystem • List and describe the architecture and components of the Spark unified stack • Describe the role of a Resilient Distributed Dataset (RDD) • Understand the principles of Spark programming • List and describe the Spark libraries • Launch and use Spark's Scala and Python shells		
Unit Number: 5	**Title : Scala and Spark**	**No. of hours: 4**
Content Summary: Explain the use and advantages of Scala Programming • Explain types of variables in Scala, Functions, Flow Control Statements. • Implement programs to experience hands on. Understand the need and use of Spark. • Explain the Spark Unified Stack. • Explain the Spark Runtime Architecture.		

Mapping /Alignment of C0s with POs

PO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PSO1	PSO2	PSO3	PSO4	PSO5
CO1	2	1	3	1	2	2	2	2	1	3	1	2
CO2	1	2	2	2	1	3	1	1	2	2	2	1
CO3	2	2	3	2	2	2	2	2	2	3	2	2
CO4	3	2	2	1	3	1	3	3	2	2	1	3
CO5	3	1	1	3	1	2	1	3	1	1	3	3

- indicate no co-relation between CO and PO/PSO,1 indicates the strength of co-relation between CO and PO/PSO is Weak/low, 2= strength of co-relation between CO and PO/PSO is Moderate/Medium, 3= strength of co-relation is Strong/High.

References

1. Gelman, Andrew, and Jennifer Hill. Data Analysis Using Regression and
2. Multilevel/Hierarchical Models. 1st ed. Cambridge, UK: Cambridge University Press,2006. ISBN:9780521867061.
3. Gelman, Andrew, John B. Carlin, Hal S. Stern, and Donald B. Rubin. Bayesian Data Analysis. 2nd ed. New York, NY: Chapman & Hall, 2003. ISBN:9781584883883
4. Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data" by EMC Education Services

Proposed Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	Implement the following file management tasks in Hadoop: • Adding files & directories • Retrieving files • Deleting files	CO1
2	Install & Run Hive then use Hive to create, alter, & drop databases, tables, joins.	CO1
3	Implement Hive Partitioning & Bucketing with data set.	CO2
4	Install & Run Pig then write Pig Latin scripts to sort, group, join & filter your data	CO3
5	Run a basic Word Count MapReduce program to understand MapReduce Paradigm with data set.	CO2
6	Working with Jupyter Notebooks. Working with notebooks. Creating Notebooks. Using Notebooks with watson studio	CO2
7	Implement Hbase commands with data set.	CO3
8	Data transactions with SQOOP	CO2
9	Create an external table using any data set and load the data in the hive.	CO1
10	Implement an internal table in the hive with the loadingof data.	CO2
11	Manipulating Data in Hive a) Data Structures in Hive b) Creating Tables in Hive c) Handling CSV files in Hive d) Bucketing Tables	CO2
12	Implement Pig Latin for the following queries: • How do you load data into Pig from a file in HDFS? • What are the different data types supported in Pig Latin? • How do you filter and transform data using Pig Latin operations like FILTER, FOREACH, and GENERATE?	CO2
13	Create sample bucket with column names such as first_name, job_id, department, salary and country and creat 4 buckets over here. and load the data into 4 buckets	CO2

14	Implement Static partition in hive and Filtering Results with Hive.	CO3
15	Implement below commands in hive using any dataset in hive. 1. Create an internal table using tab-separated data 2. Insert 3. Group by 4. Order by 5. Drop	CO3
16	Perform following task for Dynamic Partitions: You need to create a database in which you want to perform the operation of the creation of a table and enable the dynamic partition, Create any table with a suitable table name to store the data ,load the data	CO3
17	Implement the following commands using Pig Apply Flatten and Tokenize command using Pig Latin	CO3
18	Implement importing commands using Sqoop and MySQL (Using any data sets) a) Creating database and table in MySQL b) Inserting data into MySQL Table c) Importing data from MySQL to HDFS	CO2
19	Implement Exporting commands using Sqoop and MySQL(Using any data sets) Creating database and empty table in MySQL	CO1
20	What are the options for grouping and aggregating datain Pig Latin? Perform the steps.	CO1
21	Exporting data from HDFS and put into MySQL .Writethe commands.	CO1

22	Create a partitioned table ,load the data from the first table to the partitioned table and Viewing of the table	CO2
23	Implementing the basic commands of LINUX Operating System: • File/Directory Creation • Deletion	CO2
24	• Cat Command-Creating of empty file, adding data into file, append and viewing of data • Touch command-creating of file, adding data into file, and viewing of data	CO3
25	• VI Editor command and its mode- creating of file, adding data into file and viewing of data	CO3

Projects to be covered: (at least 4-5 projects). Please provide objectives of the project

☐ Medical insurance fraud detection.

☐ Data warehouse design for an E-Commerce site.

- Tourist behaviour analysis.
- Crime Detection.

DATA SCIENCE - TOOLS AND TECHNIQUES LAB

<table>
<tr><td>Program Name:</td><td colspan="3">Bachelor of Technology (CSE) with specialization in AI & ML</td></tr>
<tr><td rowspan="2">Course Name:
Data Science - Tools And Techniques Lab</td><td>Course Code</td><td>L-T-P</td><td>Credits</td></tr>
<tr><td>SEC040</td><td>0-0-4</td><td>2</td></tr>
<tr><td>Type of Course:</td><td colspan="3">SEC-5</td></tr>
<tr><td colspan="4">Pre-requisite(s), if any: General understanding of Scala 2. Experience withJava (preferred), Python, or another object oriented language 3. General understanding of machine learning</td></tr>
</table>

Course Perspective. The "Data Science - Tools and Techniques Lab" course provides a hands-on, practical approach to learning the essential tools and techniques used in data science. Students will explore various data science tools, including programming languages like Python and R, data manipulation libraries such as pandas and dplyr, and visualization tools like Matplotlib and ggplot2. The course also covers key data science techniques, including data cleaning, exploratory data analysis, statistical modeling, and machine learning algorithms. The course is divided into four modules:

a) Scala Language

b) Variables, Data Types, Conditional Statements

c) Code Blocks, Functions, Collections

d) Loops, Packages, Classes and Exceptional Handling

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	Learning to leverage the integration of Apache Spark™ and Scala.
CO 2	Analysing Spark’s machine learning pipelines to fit models and

CO 3	searching for optimal hyperparameters using Scala in a Spark cluster.
CO 4	Understanding the advantages of using Apache Spark as a Big Data analytics platform

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

UNIT WISE DETAILS		
Unit Number:1	**Title: Scala Language**	**No. of hours: 8**
Content Summary: **Scala Language:** Getting to know Scala programming language, Scala and Java, Statically typed language, Apache Spark and Scala, Scala Performance Benefits, Installing Scala, Using Scala REPL/Shell, getting help from Scala shell, Hello World, Paste mode, retrieving history, auto-complete feature, exiting from Scala REPL		
Unit Number:2	**Title: Variables, Data Types, Conditional Statements**	**No. of hours: 8**
Content Summary: **Variables, Data Types, Conditional Statements:** Immutability of variables, define mutable and immutable variables, mutability and type safety, Specifying types for variables, Scala Identifier rules, naming conventions, Scala datatypes, Boolean types, string type, multiline strings, string operations, string concatenation, string interpolation, length of string, splitting string, extracting part of string, index of character of strings, the ANY type, type casting, Boolean expressions, conditional statement in Scala, nested IF/ELSE statement, pattern matching		
Unit Number:3	**Title: Code Blocks, Functions, Collections**	**No. of hours: 8**

Content Summary: **Code Blocks, Functions, Collections:** Code Blocks in Scala, Why use functions in Scala, understanding functions in Scala, define and invoke a function, functions with multiple parameters, positional parameters, functions with no argument, single-line function, passing function as argument, anonymous function, Collections in Scala, Understanding List, list size, convert list to string, iterating over list, map function and collection, foreach, reduce operation, list equality, create set, indexing map, manipulating maps, understanding tuples, indexing tuples, mutable collections, nested collections		
Unit Number:4	**Title: Loops, Packages, Classes and Exceptional Handling**	**No. of hours: 8**
Content Summary: **Loops, Packages, Classes and Exceptional Handling:** For loop, While loop, Breaking Loop iteration, classes and objects in Scala, Create classes and objects, singleton objects, case classes, equality checks, classes and packages, avoid name space collusion, importing package, fundamental of exception handling, type inferences and exception handling, try, catch, finally, Scala built tool (SBT), Compile Scala applications		

Learning Experiences

Classroom Learning Experience

1. **Lectures and Demonstrations:**
   - Interactive lectures covering core concepts of Scala, Apache Spark, and data science techniques.
   - Live coding demonstrations to illustrate programming practices and tool usage.
2. **Hands-on Coding Sessions:**
   - Practical exercises in Scala programming, including data manipulation and visualization tasks.
   - Guided sessions on using Scala REPL and executing Spark jobs.
3. **Group Projects:**
   - Collaborative projects where students apply learned concepts to solve data science problems.
   - Team presentations to share findings and methodologies.

4. **Workshops and Lab Sessions:**
    - Structured lab sessions focusing on specific topics (e.g., machine learning pipelines, exception handling).
    - Problem-solving workshops that reinforce concepts through practical application.
5. **Quizzes and Assessments:**
    - Regular quizzes to assess understanding of theoretical concepts and practical skills.
    - Hands-on assessments requiring coding solutions to given problems.

Outside Classroom Learning

1. **Self-Directed Learning:**
    - Online resources (tutorials, documentation, videos) for additional practice and exploration of advanced topics.
    - Recommended reading materials on data science methodologies and Scala programming.
2. **Online Forums and Communities:**
    - Participation in coding forums and communities (e.g., Stack Overflow, GitHub) to seek help and share knowledge.
    - Engaging in discussions on best practices and current trends in data science.
3. **Real-World Projects:**
    - Opportunities to work on real-world data science projects or internships that require the application of Scala and Spark.

Proposed Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1.	Write a program to install Scala	CO1
2.	Write a program to use Scala REPL/Shell	CO1
3.	Write a program to implement Hello World in Scala	CO1
4.	Write a program to define mutable and immutable functionsin Scala	CO1
5.	Write a program to define Scala Data types	CO1
6.	Write a program to implement string operations in scala	CO1
7.	Write a program to illustrate Boolean expressions in Scala	CO1
8.	Write a program to define and invoke a function	CO1
9.	Write a program to implement Collections in Scala.	CO3
10.	Write a program to implement Loops in Scala	CO3
11.	Write a program to create classes and objects	CO1, CO4
12.	Write a program to implement exceptional handling	CO1, CO4

Summer Internship-II

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Summer Internship-II	**Course Code**	**L-T-P**	**Credits**
	ENSI351	0-0-0	2
Type of Course:	INT-2		
Pre-requisite(s), if any: NA			

Duration:

The internship will last for six weeks. It will take place after the completion of the 4th semester and before the commencement of the 5th semester.

Internship Options:

Students can choose from the following options:

- **Industry Internship (Offline) or Internship in Renowned Institutions (Offline):**
  - Students must produce a joining letter at the start and a relieving letter upon completion.

Report Submission and Evaluation:

1. **Report Preparation:**
   - Students must prepare a detailed report documenting their internship experience and submit it to the department. A copy of the report will be kept for departmental records.
2. **Case Study/Project/Research Paper:**
   - Each student must complete one of the following as part of their internship outcome:
     1. A case study
     2. A project

3. A research paper suitable for publication

3. **Presentation:**
   - Students are required to present their learning outcomes and results from their summer internship as part of the evaluation process.

Evaluation Criteria for Summer Internship (Out of 100 Marks):

1. **Relevance to Learning Outcomes (30 Marks)**
   - **Case Study/Project/Research Paper Relevance (15 Marks):**
     1. Directly relates to core subjects: 15 marks
     2. Partially relates to core subjects: 10 marks
     3. Minimally relates to core subjects: 5 marks
     4. Not relevant: 0 marks
   - **Application of Theoretical Knowledge (15 Marks):**
     1. Extensive application of theoretical knowledge: 15 marks
     2. Moderate application of theoretical knowledge: 10 marks
     3. Minimal application of theoretical knowledge: 5 marks
     4. No application of theoretical knowledge: 0 marks
2. **Skill Acquisition (40 Marks)**
   - **New Technical Skills Acquired (20 Marks):**
     1. Highly relevant and advanced technical skills: 20 marks
     2. Moderately relevant technical skills: 15 marks
     3. Basic technical skills: 10 marks
     4. No new skills acquired: 0 marks
   - **Professional and Soft Skills Development (20 Marks):**
     1. Significant improvement in professional and soft skills: 20 marks
     2. Moderate improvement in professional and soft skills: 15 marks
     3. Basic improvement in professional and soft skills: 10 marks

    4. No improvement: 0 marks

3. **Report Quality (15 Marks)**
    - **Structure and Organization (8 Marks):**
        1. Well-structured and organized report: 8 marks
        2. Moderately structured report: 6 marks
        3. Poorly structured report: 3 marks
        4. No structure: 0 marks
    - **Clarity and Comprehensiveness (7 Marks):**
        1. Clear and comprehensive report: 7 marks
        2. Moderately clear and comprehensive report: 5 marks
        3. Vague and incomplete report: 2 marks
        4. Incomprehensible report: 0 marks
4. **Presentation (15 Marks)**
    - **Content Delivery (8 Marks):**
        1. Clear, engaging, and thorough delivery: 8 marks
        2. Clear but less engaging delivery: 6 marks
        3. Somewhat clear and engaging delivery: 3 marks
        4. Unclear and disengaging delivery: 0 marks
    - **Visual Aids and Communication Skills (7 Marks):**
        1. Effective use of visual aids and excellent communication skills: 7 marks
        2. Moderate use of visual aids and good communication skills: 5 marks
        3. Basic use of visual aids and fair communication skills: 2 marks
        4. No use of visual aids and poor communication skills: 0 marks

Total: 100 Marks

Course Outcomes:

By the end of this course, students will be able to:

1. **Apply Theoretical Knowledge:**
   - Integrate and apply theoretical knowledge gained during coursework to real-world industry or research problems.
2. **Develop Technical Skills:**
   - Acquire and demonstrate advanced technical skills relevant to the field of computer science and engineering through practical experience.
3. **Conduct Independent Research:**
   - Execute independent research projects, including problem identification, literature review, methodology design, data collection, and analysis.
4. **Prepare Professional Reports:**
   - Compile comprehensive and well-structured reports that document the internship experience, project details, research findings, and conclusions.
5. **Enhance Problem-Solving Abilities:**
   - Develop enhanced problem-solving and critical thinking skills by tackling practical challenges encountered during the internship.
6. **Improve Professional and Soft Skills:**
   - Exhibit improved professional and soft skills, including communication, teamwork, time management, and adaptability in a professional setting.
7. **Present Findings Effectively:**
   - Deliver clear and engaging presentations to effectively communicate project outcomes, research findings, and acquired knowledge to peers and faculty members.
8. **Pursue Lifelong Learning:**
   - Demonstrate a commitment to lifelong learning by engaging in continuous skill development and staying updated with emerging trends and technologies in the field.

Learning Experiences

Classroom Learning Experience

1. **Orientation Sessions**: Introduce internship objectives, expectations, and assessment criteria.
2. **Skill Development Workshops**: Cover essential professional skills such as communication, teamwork, and time management.
3. **Project Planning Guidance**: Assist students in developing project proposals aligned with internship goals.
4. **Progress Presentations**: Facilitate sessions for students to present updates and receive constructive feedback.
5. **Group Discussions**: Encourage sharing of challenges and solutions experienced during internships.
6. **Continuous Feedback**: Implement regular check-ins and mentor evaluations to assess student progress.

Outside Classroom Learning Experience

1. **Internship Placement**: Engage students in real-world work environments to apply learned skills.
2. **Reflective Journals**: Encourage students to document experiences, challenges, and lessons learned throughout the internship.
3. **Project Implementation**: Work on assigned projects and tasks within the organization, applying theoretical knowledge.
4. **Networking Opportunities**: Create opportunities for students to connect with industry professionals and peers.
5. **Self-Assessment**: Provide tools for students to evaluate their performance and contributions during the internship.
6. **Final Presentations**: Organize sessions for students to showcase their internship experiences and project outcomes to faculty and peers.

Arithmetic & Reasoning Skills

Programme Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**			
Course Name: Arithmetic & Reasoning Skills	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	AEC008	3-0-0	3	24
Type of Course:	AEC-3			
Pre-requisite(s), if any:				

Course Perspective: The course aims to improve basic arithmetic skills, speed, and accuracy in mental calculations, and logical reasoning. These abilities are essential for a strong math foundation, helping students succeed in academics and various practical fields.

The Course Outcomes (COs): On completion of the course the participants will be able to:

COs	Statements
CO 1	**Understanding** arithmetic algorithms required for solving mathematical problems.
CO 2	**Applying** arithmetic algorithms to improve proficiency in calculations.
CO 3	**Analyzing** cases, scenarios, contexts and variables, and understanding their inter-connections in a given problem.
CO 4	**Evaluating** & deciding approaches and algorithms to solve mathematical & reasoning problems.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title:** Mathematical Essentials	**No. of hours: 15**

Content: Table chart, Line graph, Bar graph, Pie chart

Unit Number: 2	**Title:** Fundamentals of Logical Reasoning	**No. of hours: 6**

Content: Blood Relations, Direction Sense, Coding Decoding

Unit Number: 3	**Title:** Elementary Quantitative Skills	**No. of hours: 18**

Content: Simple and Compound Interest, Average, Partnership, Time and Work, Time Speed & Distance

Unit Number: 4	**Title:** Advanced Quantitative Skills	**No. of hours: 6**

Content:

Permutation & Combination, Probability

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce advanced life skills concepts using PPTs and real-life scenarios.
2. **Conceptual Understanding**: Cover topics like strategic thinking, resilience, and ethical decision-making.

3. **Problem-Solving Sessions**: Conduct in-class exercises focused on real-world workplace dilemmas and solutions.
4. **Group Discussions**: Facilitate discussions on leadership styles, team dynamics, and conflict management.
5. **Guest Speakers**: Invite industry experts to share insights on personal and professional development.
6. **Continuous Feedback**: Implement quizzes and peer reviews to assess application of life skills in professional contexts.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign reflective essays on personal experiences and career aspirations.
2. **Workshops**: Facilitate hands-on sessions for practicing advanced communication and negotiation skills.
3. **Question Bank**: Provide resources for self-assessment on life skills development and application.
4. **Online Forums**: Create platforms for discussing personal growth challenges and sharing experiences.
5. **Self-Study for Case Studies**: Encourage independent research on successful professionals and their life skills.
6. **Collaborative Projects**: Organize group projects focused on community engagement and leadership initiatives.

References

R 1. Aggarwal, R. S. (2014). Quantitative aptitude (Revised edition).

R 2. Gladwell, M. (2021). Talking to strangers.

R 3. Scott, S. (2004). Fierce conversations.

COMPETITIVE CODING BOOTCAMP-III

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **COMPETITIVE CODING BOOTCAMP-III**	**Course Code**	**L-T-P**	**Credits**
		2-0-0	0
Type of Course:	AUDIT-3		
Contact Hours	30		
Version			

Course Outcomes

CO1	Analyzing and writing SQL queries to retrieve, modify, and optimize data in relational databases.
CO2	Implementing efficient tree-based data structures like AVL, B Trees, and Splay Trees to solve computational problems.
CO3	Developing solutions for optimization problems using greedy algorithms and dynamic programming approaches.

CO4 Evaluating graph algorithms for traversing, searching, and finding shortest paths in complex graph structures.

<table>
<tr><td>Unit Number: 1</td><td>Title: SQL & PL/SQL</td><td>No. of hours: 8</td></tr>
<tr><td colspan="3">Content:
Introduction to Databases and SQL:
○ Understand relational databases, tables, and SQL queries.
○ Practice SELECT, INSERT, UPDATE, DELETE statements.
Joins and Subqueries:
○ Master INNER JOIN, LEFT JOIN, RIGHT JOIN, and self-joins.
○ Learn about subqueries and correlated subqueries.
Indexes and Query Optimization:
○ Explore indexing techniques (B-tree, hash indexes).
○ Optimize SQL queries for performance.
PL/SQL Basics:
○ Introduce PL/SQL (Procedural Language/Structured Query Language).
○ Write basic PL/SQL blocks, loops, and conditional statements</td></tr>
<tr><td>Unit Number:</td><td>Title: Height Balanced Tree Concepts</td><td>No.</td></tr>
</table>

<table>
<tr><td>2</td><td></td><td>of
ho
ur
s:
8</td></tr>
<tr><td colspan="3">Content:
AVL Trees: Definition and Properties, Rotations , AVL Tree Operations (Insertion, Deletion, Lookup), Complexity Analysis

B Trees : Definition and Properties, B Tree Operations, Complexity Analysis, Applications in Databases and File Systems
B+ Trees: Definition and Properties, B+ Tree Operations, Complexity Analysis

Splay Trees: Definition and Properties, Splaying Operation, Splay Tree Operations (Insertion, Deletion, Lookup), Complexity Analysis

Applications of Height Balanced Trees: Use in Databases (Indexing), Use in Memory Management (Allocators)</td></tr>
<tr><td>Unit Number: 3</td><td>Title: Greedy Design Strategy and Dynamic Programming</td><td>No. of ho ur s: 8</td></tr>
<tr><td colspan="3">Content:
Greedy Algorithms: Definition and Characteristics, Greedy Choice</td></tr>
</table>

Property, Optimal Substructure **Dynamic Programming:** Definition and Characteristics, Optimal Substructure, Overlapping Subproblems, Comparison with Greedy Algorithms **Greedy Algorithms** **Basic Greedy Algorithms:** Activity Selection Problem, Huffman Coding, Kruskal's Algorithm, Prim's Algorithm, Fractional Knapsack Problem Complexity Analysis: Time Complexity, Proof of Optimality **Dynamic Programming** **Basic Dynamic Programming Problems:** Fibonacci Sequence (Memoization vs. Tabulation), 0/1 Knapsack Problem, Longest Common Subsequence (LCS), Matrix Chain Multiplication		
Unit Number: 4	**Title: Graph Algorithms**	**No. of hours: 6**
Content: **Graph Representations:** ▪ Representing graphs using adjacency matrix and adjacency list. ▪ Solving basic graph traversal problems. **Breadth-First Search (BFS):** ▪ Implementing BFS for finding the shortest path in unweighted graphs. ▪ Applications include finding connected components. **Depth-First Search (DFS):** ▪ Implementing DFS for tasks like topological sorting and cycle detection. **Shortest Path Algorithms**		

Dijkstra's Algorithm:

- Implementing Dijkstra's algorithm for finding shortest paths in weighted graphs.
- Using priority queues for efficient computation.

Bellman-Ford Algorithm:

- Handling negative weights with the Bellman-Ford algorithm.
- Detecting negative weight cycles.

Lab Experiments

Problem Statement	Mapped COs
SQL & PL/SQL	
1. Write an SQL query to find the department with the highest average salary.	CO1
2. Write a query to find all employees who earn more than their managers.	CO1
3. Retrieve the top three salaries from the "employees" table.	CO1
4. Write an SQL query to find employees who have been in the company for more than 5 years.	CO1
5. Write a query to delete duplicate rows from a table without using temporary tables.	CO1
6. Fetch all records where the customer ordered more than once from the "orders" table.	CO1
7. Find the name of departments with more than 10 employees using a JOIN between "employees" and "departments".	CO2

Problem Statement	Mapped COs
8. Write a query to retrieve all customers who ordered more than the average number of orders using subqueries.	CO2
9. Write a query to find the second highest salary of employees using a subquery.	CO2
10. Optimize the performance of a query that fetches all orders placed in the last 30 days from the "orders" table using indexing.	CO3
11. Use indexing to speed up searches on the "products" table and compare the execution time before and after indexing.	CO3
12. Write a PL/SQL block to display the Fibonacci sequence up to a given number using loops.	CO4
13. Create a PL/SQL block that calculates the factorial of a number using recursion.	CO4
Height Balanced Tree Concepts	
14. Implement an AVL Tree and insert a series of elements into it. Ensure the tree remains balanced after each insertion.	CO5
15. Write a function to check if a given AVL Tree is height-balanced.	CO5
16. Perform AVL Tree deletion and ensure rebalancing using rotations.	CO5
17. Implement an AVL Tree lookup operation and calculate its time complexity.	CO5
18. Implement insertion operations in a B Tree and verify the tree’s structure after each insertion.	CO6
19. Write a function to search for an element in a B Tree and trace the steps of the search.	CO6
20. Implement deletion operations in a B Tree and verify rebalancing	CO6

Problem Statement	**Mapped COs**
after each deletion.	
21. Perform insertion and deletion operations in a B+ Tree and trace the changes in the tree structure.	CO6
22. Demonstrate the application of B Trees in database indexing with a small dataset.	CO6
23. Implement a splay tree and observe the behavior of nodes being splayed to the root after lookups.	CO6
24. Compare the performance of AVL Trees, B Trees, and Splay Trees for a series of random insertions.	CO6
Greedy Design Strategy and Dynamic Programming	
25. Solve the Activity Selection Problem using a greedy algorithm.	CO7
26. Write a function to implement Huffman coding for a string of characters and display the encoded output.	CO7
27. Implement Kruskal’s algorithm for finding the minimum spanning tree of a graph.	CO7
28. Solve the Fractional Knapsack Problem using a greedy approach.	CO7
29. Solve the 0/1 Knapsack Problem using dynamic programming.	CO8
30. Write a program to find the nth Fibonacci number using memoization and compare it with the iterative approach.	CO8
31. Implement dynamic programming to solve the Longest Common Subsequence (LCS) problem.	CO8
32. Solve the Matrix Chain Multiplication problem using dynamic programming and analyze the time complexity.	CO8

Problem Statement	Mapped COs
Graph Algorithms	
33. Implement a graph using an adjacency list and perform Depth-First Search (DFS) to detect cycles.	CO9
34. Implement Breadth-First Search (BFS) to find all connected components in an unweighted graph.	CO9
35. Solve a shortest-path problem in an unweighted graph using BFS.	CO9
36. Write a program to implement Dijkstra's algorithm to find the shortest path in a weighted graph.	CO10
37. Implement Bellman-Ford algorithm to find shortest paths in a graph with negative weights.	CO10
38. Detect negative weight cycles in a graph using the Bellman-Ford algorithm.	CO10
39. Perform topological sorting of a directed graph using DFS.	CO9
40. Solve a shortest-path problem using Dijkstra's algorithm with priority queues and analyze the time complexity.	CO10

Semester: VI

COMPUTER ORGANIZATION & ARCHITECTURE

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Computer Organization & Architecture	**ENCS302**	3-1-0	4
Type of Course:	Major-20		

Pre-requisite(s), if any: Concepts of Digital Electronics

Course Perspective. This course provides a foundational understanding of computer organization and architecture. It covers essential concepts such as computer components, memory hierarchy, and processor design. Students will explore data representation, caching strategies, and I/O systems, focusing on practical applications and performance optimization. By combining theoretical knowledge with hands-on learning, the course aims to equip students with the skills necessary to understand and improve computer systems.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements

CO 1	**Understanding** the basics of instructions sets and their impact on processor design.
CO 2	**Demonstrating** an understanding of the design of the functional units of a digital computer system
CO 3	**Evaluating** cost performance and design trade-offs in designing and constructing a computer processor including memory.
CO 4	**Designing** a pipeline for consistent execution of instructions with minimum hazards
CO 5	**Manipulating** representations of numbers stored in digital computers using I/O devices and store them into memory

Course Outline:

Unit Number: 1	**Title: Introduction to Computer Architecture**	**No. of hours: 10**

Content Summary:

Basics of Computer Architecture: Von Neumann architecture, CPU, memory, and I/O subsystems.

Instruction Set Architecture (ISA): Registers, instruction execution cycle, addressing modes.

Data Representation: Number systems (binary, octal, decimal, hexadecimal), Arithmetic operations (addition, subtraction), Floating point representation (IEEE 754 standard).

Instruction Set Types: Introduction to RISC and CISC architectures.

Unit Number: 2	**Title: Memory Hierarchy and I/O Systems**	**No. of hours: 10**

Content Summary:

Memory Hierarchy: RAM, ROM, Cache, and secondary storage.

Cache Memory: Direct-mapped, set-associative, fully associative caches, Write-through vs. write-back caches.

Storage: Introduction to magnetic disks, Flash memory (NAND and NOR flash).

I/O Techniques: Programmed I/O, Interrupt-driven I/O, Direct Memory Access (DMA).

Unit Number: 3	Title: Processor Design	No. of hours: 10

Content Summary:

Processor Basics: Building a simple datapath, single-cycle and multi-cycle processor designs.

Pipelining: Introduction, stages, hazards (data, control) and mitigation strategies.

Clocking Methodology: Basics of clocking, Amdahl's Law**.**

Instruction Level Parallelism: Concept and basic strategies for parallelism**.**

Unit Number: 4	Input/Output Systems and Advanced Topics	No. of hours: 10

Content Summary:

I/O Systems: Memory-mapped vs. I/O-mapped I/O, DMA.

Advanced Memory Concepts: Memory interleaving, processor-cache interactions.

Storage Technologies: Disk scheduling algorithms, flash memory structure**.**

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key concepts in computer organization using PPTs and diagrams.
2. **Conceptual Understanding**: Cover topics like CPU architecture, memory hierarchy, and I/O systems.
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on assembly language and instruction set architecture.
4. **Case Studies**: Analyze real-world computer architectures and their performance metrics.

5. **Group Work**: Collaborate on projects that involve designing and simulating computer systems.
6. **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding of organizational concepts.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign projects that require application of concepts in computer organization and architecture.
2. **Lab Projects**: Facilitate hands-on tasks involving hardware simulations and assembly programming.
3. **Question Bank**: Provide practice problems and resources for self-assessment on architecture topics.
4. **Online Forums**: Create platforms for discussing challenges and sharing solutions related to computer architecture.
5. **Self-Study for Case Studies**: Encourage independent research on current trends in computer organization.
6. **Collaborative Projects**: Organize group projects focused on building and optimizing computer architectures.

Textbooks

- Stallings, W. (2016). *Computer Organization and Architecture* (10th ed.). Pearson.
- Patterson, D. A., & Hennessy, J. L. (2020). *Computer Organization and Design: The Hardware/Software Interface* (5th ed.). Morgan Kaufmann.
- Mano, M. M. (2017). *Computer System Architecture* (3rd ed.). Pearson.
- Tanenbaum, A. S., & Austin, T. (2013). *Structured Computer Organization* (6th ed.). Pearson.
- Hennessy, J. L., & Patterson, D. A. (2017). *Computer Architecture: A Quantitative Approach* (6th ed.). Morgan Kaufmann.

Additional Readings:

Online Learning References

a) **MIT OpenCourseWare - Computer System Engineering**
   a. **Link:** MIT OCW
   b. **Description:** This course provides a deep dive into computer system architecture, exploring processor design, memory systems, and parallel processing.

b) **GeeksforGeeks - Computer Organization and Architecture**
   a. **Link:** [GeeksforGeeks]
   b. **Description:** GeeksforGeeks provides detailed tutorials on various topics in computer organization and architecture, such as instruction sets, pipelining, and memory hierarchy.

c) **NPTEL - Computer Architecture**
   a. **Link:** [NPTEL]
   b. **Description:** This course from NPTEL covers the principles of computer architecture, including instruction sets, CPU design, and memory systems, with a focus on practical applications.

COMPUTER NETWORKS

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **Computer Networks**	**Course Code**	**L-T-P**	**Credits**
	ENCS304	4-0-0	4
Type of Course:	Major-21		
Pre-requisite(s), if any:			

Course Perspective. The Computer Networks course is designed to provide students with a comprehensive understanding of network systems and their components. The course explores the fundamental principles of data communication, network architectures, and protocols essential for designing and managing modern network systems. Emphasis is placed on both theoretical concepts and practical applications, including network topologies, data link layer protocols, and network layer functionalities. Students will gain insights into network performance metrics, error control mechanisms, and network security practices. Through a combination of lectures, hands-on labs, and project-based assignments, students will develop the skills necessary to analyze, implement, and troubleshoot network systems effectively. The course aims to equip students with the knowledge and skills required to succeed in the field of network engineering and administration.

The Course Outcomes (COs). On completion of the course the participants will be able to:

COs	Statements
CO 1	**Understanding** the fundamental concepts and principles of computer

	networks.
CO 2	**Applying** knowledge of network hardware and software components.
CO 3	**Developing** skills in network administration and management.
CO 4	**Identifying** appropriate protocol for desired communication service.

Course Outline:

Unit Number: 1	**Title: Evolution of Computer Networking**	**No. of hours: 10**
Content Summary: **Introduction to Computer Networks**: Overview, Evolution, and Trends **Data Communication Components**: Representation of data, data flow, and network elements **Network Topologies:** Star, Mesh, Bus, and Ring **Networking Models:** OSI Model and TCP/IP Model **Protocols and Standards:** Key protocols (e.g., Ethernet, IP) and standards organizations **Physical Media:** Transmission media (copper, fiber, wireless) **Network Architectures:** Circuit switching, packet switching, and network of networks **Performance Metrics:** Packet delay, loss, and end-to-end throughput		
Unit Number: 2	**Title: Data Link Layer**	**No. of hours: 10**

Content Summary: **Data Link Layer Overview**: Functions and services **Error Detection and Correction**: Techniques like Block Coding, Hamming Code, CRC **Flow Control and Error Control Protocols**: Stop-and-Wait, Go-Back-N, Selective Repeat ARQ, Sliding Window Protocols **Medium Access Control (MAC) Protocols:** Pure ALOHA, Slotted ALOHA, CSMA/CD, and CDMA/CA **Link Layer Technologies:** Ethernet, PPP, and Frame Relay		
Unit Number: 3	**Title: Introduction to Network Layer and Transport Services**	**No. of hours: 10**
Content Summary: **Network Layer Functions**: Routing, switching, and logical addressing **Addressing Schemes**: IPv4, IPv6, and Address Resolution Protocols (ARP, RARP) **Dynamic Address Assignment:** BOOTP and DHCP **Routing Protocols**: Distance Vector (RIP), Link State (OSPF), and Path Vector (BGP) **Transport Layer Protocols:** UDP, TCP, SCTP **Congestion Control and Quality of Service (QoS):** Techniques like Leaky Bucket, Token Bucket, and congestion management.		
Unit Number: 4	**Title: Application Layer**	**No. of hours: 10**
Content Summary: **Application Layer Protocols**: DNS, DHCP, TELNET, FTP, HTTP **Email Protocols:** SMTP, IMAP, POP3 **Web Technologies:** WWW, HTML, and HTTP **Network Security Basics:** Firewalls, Introduction to Cryptography		

(encryption methods and security protocols)

Bluetooth Technology: Basics of Bluetooth and its applications

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures**: Introduce key concepts in computer networks using PPTs and network diagrams.
2. **Conceptual Understanding**: Cover topics like network protocols, topologies, and architecture models (OSI, TCP/IP).
3. **Problem-Solving Sessions**: Conduct in-class exercises focused on troubleshooting network issues and configuring devices.
4. **Case Studies**: Analyze real-world network designs and their performance implications.
5. **Group Work**: Collaborate on projects that involve designing and simulating network setups.
6. **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding of networking concepts.

Outside Classroom Learning Experience

1. **Theory Assignments**: Assign projects that require practical application of networking principles.
2. **Lab Projects**: Facilitate hands-on tasks involving network configuration and management tools.
3. **Question Bank**: Provide practice problems and resources for self-assessment on networking topics.
4. **Online Forums**: Create platforms for discussing networking challenges and sharing solutions.
5. **Self-Study for Case Studies**: Encourage independent research on current trends and technologies in computer networking.
6. **Collaborative Projects**: Organize group projects focused on developing efficient network solutions for real-world scenarios.

Textbooks:

T1: " Data Communication and Networking", 5th Edition, Behrouz A. Forouzan, McGraw-Hill, 2012.

T2: "Computer Networks", Andrew S. Tanenbaum and David J. Wetherall, Pearson, 5th Edition, 2010.

T3: "Computer Networking A Top-Down Approach". 5th Edition, James F. Kurose-Keith W. Ross (Pearson).

Additional Readings:

Online Learning References

I) **MIT OpenCourseWare - Computer Networks**
   a. **Link:** MIT OCW
   b. **Description:** This course provides a thorough exploration of computer networks, focusing on network design, protocol layers, and network management.

II) **GeeksforGeeks - Computer Networks**
   a. **Link:** GeeksforGeeks
   b. **Description:** GeeksforGeeks provides detailed tutorials on various aspects of computer networks, such as the OSI model, data link layer, network layer, and transport layer protocols.

III) **NPTEL - Computer Networks**
   a. **Link:** NPTEL
   b. **Description:** This course from NPTEL provides a comprehensive overview of computer networking, including topics like error detection, IP addressing, and routing protocols.

COMPUTER NETWORKS LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Computer Networks Lab	**Course Code**	**L-T-P**	**Credits**
	ENCS352	0-0-2	1
Type of Course:	Major-23		
Pre-requisite(s), if any:			

Proposed Lab Experiments

Defined Course Outcomes

COs	
CO1	**Applying** fundamental networking concepts and techniques to develop and analyze network topologies, protocols, and error detection mechanisms.
CO2	**Designing** and implement network protocols and architectures for efficient data communication and management in various environments.
CO3	**Utilizing** advanced networking techniques to implement, monitor, and optimize communication systems for real-time and multimedia applications.
CO	**Integrating** IoT devices and develop smart systems using networking principles for automation and efficient data

4	management.	

Experiment Title	Mapped CO/COs
Design and simulate a simple computer network using various connection topologies (bus, star, ring, mesh). Compare the advantages and disadvantages of each topology in terms of data flow and network efficiency.	CO1
Create a network simulation to demonstrate packet switching and circuit switching. Compare the performance and efficiency of both methods by simulating a series of data transmission scenarios.	CO1
Develop a network simulator to analyze packet delay, loss, and end-to-end throughput. Implement various routing	CO

algorithms and measure their impact on network performance under different traffic conditions.	1
Implement error detection and correction mechanisms using block coding and CRC. Simulate a communication system that demonstrates how errors are detected and corrected during data transmission.	CO2
Design and simulate flow control and error control protocols such as Stop and Wait, Go-Back-N ARQ, and Selective Repeat ARQ. Compare their performance in terms of throughput and efficiency under varying network conditions.	CO2
Develop a simulation to demonstrate multiple access protocols such as Pure ALOHA, Slotted ALOHA, CSMA/CD, and CSMA/CA. Analyze the performance of each protocol in handling network collisions and maximizing data transmission efficiency	CO2
Implement a sliding window protocol with piggybacking for efficient data transmission and error control. Simulate data transfer between two nodes and visualize the window movements and acknowledgments.	CO2
Create a simulation to demonstrate logical addressing using IPv4 and IPv6. Implement address mapping techniques such as ARP, RARP, BOOTP, and DHCP to show how devices acquire and resolve network addresses.	CO3
Implement a transport layer simulation to demonstrate process-to-process communication using UDP, TCP, and SCTP. Compare the protocols in terms of connection establishment, data transmission, and congestion control.	CO3
Implement a DNS and DDNS simulation to demonstrate domain name resolution and dynamic updates. Create a	CO

simple client-server application that queries and updates the DNS records.	4
Create a web server simulation to demonstrate the workings of HTTP and WWW. Implement basic HTTP request and response handling, and simulate a simple web browsing session.	CO4

INTRODUCTION OF NEURAL NETWORK AND DEEP LEARNING

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Neural Networks and Deep Learning	**Course Code**	**L-T-P**	**Credits**
	ENSP310	4-0-0	4
Type of Course:	Major-22		
Pre-requisite(s), if any: Fundamentals of AI and Machine Learning, Programming knowledge			

Course Perspective. This course provides a comprehensive introduction to the fundamental concepts of neural networks and deep learning, which are crucial for the development of intelligent systems and advanced data analysis. Students will explore core topics including the basic structure and function of neural networks, feedforward neural networks, deep learning techniques, and probabilistic neural networks. The course emphasizes both theoretical understanding and practical implementation, preparing students to tackle real-world problems using advanced neural network models.

The Course Outcomes (COs). On completion of the course the participants will be able to:

COs	**Statements**
CO 1	**Understanding** key concepts of neural networks and deep learning, including ANNs and their biological equivalents.
CO 2	**Implementing** basic neural network models and training algorithms using popular deep learning frameworks

CO 3	**Comparing** and **contrasting** different deep learning architectures, such as CNNs, RNNs, and GANs, and their applications in various domain
CO 4	**Assessing and optimizing** deep learning models by applying regularization techniques, tuning hyperparameters, and evaluating performance metrics.

Course Outline:

Unit Number: 1	**Title: Introduction to Neural Networks**	**No. of hours: 10**
Content: ▪ Human Brain and Artificial Neuron Models ▪ Biological vs. Artificial Neural Networks ▪ Evolution and Characteristics of Neural Networks ▪ Learning Methods: Supervised, Unsupervised, Reinforcement ▪ Taxonomy of Neural Network Architectures		
Unit Number: 2	**Title: Supervised and Unsupervised Neural Networks**	**No. of hours: 10**
Content: **Supervised learning:** - Supervised Learning Networks, Perceptron Networks, Adaptive Linear Neuron, Back-propagation Network. Associative Memory Networks. Training Algorithms for pattern association. **Unsupervised learning**: - Introduction, Fixed Weight Competitive Nets, Maxnet, Hamming Network, Kohonen Self-Organizing Feature Maps, Learning Vector Quantization, Counter Propagation Networks, Adaptive Resonance Theory Networks.		
Unit Number: 3	**Title: Deep learning and Regularization for Deep Learning**	**No. of hours: 10**

Content: **Introduction to Deep Learning**: Historical Trends and Development, Deep Feed-Forward Networks, Gradient-Based Learning, Architecture Design and Hidden Units, Back-Propagation and Differentiation Algorithms **Regularization Techniques**: Parameter Norm Penalties, Data Augmentation and Noise Robustness, Semi-Supervised Learning and Multi-Task Learning, Early Stopping, Sparse Representations, Bagging and Ensemble Methods		
Unit Number: 4	**Title: Optimization for Train Deep Models**	**No. of hours: 10**
Content: **Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM), Generative Adversarial Networks (GANs);** **Optimization Techniques:** Parameter Initialization Strategies, Adaptive Learning Rates (Adam, RMSprop), Regularization Techniques (Dropout, Batch Normalization), Advanced Optimization Algorithms (Stochastic Gradient Descent, Mini-Batch Gradient Descent) **Applications and Trends:** Large-Scale Deep Learning Applications, Computer Vision, Speech Recognition, Natural Language Processing, Recent Trends and Research in Deep Learning		

Learning Experiences:

Inside Classroom Learning

1. **Lectures and Interactive Discussions:**
    - Instructor-led lectures covering key concepts of neural networks, deep learning architectures, and optimization techniques.

- Interactive discussions to clarify complex topics and encourage student engagement.

2. **Hands-on Coding Labs:**
   - Practical sessions where students implement neural network models using popular deep learning frameworks (e.g., TensorFlow, PyTorch).
   - Guided exercises to reinforce theoretical knowledge through coding.
3. **Group Projects:**
   - Collaborative projects that require students to design, implement, and present deep learning models for specific applications.
   - Peer reviews and group presentations to foster teamwork and critical feedback.
4. **Workshops:**
   - Specialized workshops on specific topics such as CNNs, RNNs, or GANs, focusing on practical implementation and problem-solving strategies.
5. **Quizzes and Assessments:**
   - Regular quizzes to test understanding of theoretical concepts and practical skills.
   - Assignments that involve building and optimizing neural network models.

Outside Classroom Learning

1. **Self-Directed Learning:**
   - Access to online resources, tutorials, and videos to deepen understanding of neural networks and deep learning concepts.
   - Recommended readings from textbooks and research papers.
2. **Online Communities and Forums:**
   - Participation in forums such as Stack Overflow and GitHub to seek help, share knowledge, and collaborate on projects.
   - Engaging with data science and machine learning communities to discuss trends and advancements.
3. **Real-World Applications and Projects:**

   - Opportunities to work on real-world data science projects or internships that apply deep learning techniques.
   - Involvement in hackathons or competitions (e.g., Kaggle) to gain practical experience.
4. **Networking and Professional Development:**
   - Attendance at industry conferences, webinars, or meetups focused on deep learning and AI to connect with professionals and researchers.
   - Engaging with guest speakers from industry to gain insights into current applications and future trends.

Textbooks:

- Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
- Aggarwal, C. C. (2018). Neural Networks and Deep Learning: A Textbook. Springer.
- Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.
- Shanmugamani, R. (2018). Deep Learning for Computer Vision. Packt Publishing.
- Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media.

Online Learning References

1. **Deep Learning by Ian Goodfellow, Yoshua Bengio, and Aaron Courville**
   - Deep Learning Book
2. **Neural Networks and Deep Learning by Michael Nielsen**
   - Neural Networks and Deep Learning
3. **CS231n: Convolutional Neural Networks for Visual Recognition by Stanford University**
   - CS231n Course
4. **Deep Learning Specialization by Andrew Ng on Coursera**
   - Deep Learning Specialization

5. Introduction to Deep Learning by MIT OpenCourseWare

- MIT OpenCourseWare

DEEP LEARNING PRACTICAL WITH PYTHON, TENSORFLOW AND KERAS

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Deep Learning Practical with Python, TensorFlow and Keras	**Course Code**	**L-T-P**	**Credits**
	ENCS360	0-0-2	1
Type of Course:	Major-24		
Pre-requisite(s), if any:			

Course Perspective. This course provides a solid foundation in deep learning concepts and techniques, starting from the basics of neural networks to advanced architectures. Students will gain a thorough understanding of how deep learning models work, enabling them to grasp more complex topics in the future. The course is divided into 4 modules:

a) Moving beyond gradient descent

b) Convolutional Neural Network

c) Embedding and Representation Learning

d) Models for Sequence Analysis

Defined Course Outcomes

COs	
CO1	**Recalling and describing** the fundamental concepts of deep learning, including neural networks, backpropagation, and activation functions
CO2	**Explaining** the architecture and functioning of different types of neural networks, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).
CO3	**Applying** TensorFlow and Keras libraries to implement and train deep learning models for image classification and natural language processing tasks.
CO4	**Analyzing** the performance of deep learning models by interpreting accuracy metrics, confusion matrices, and loss curves.

Course Outline

Unit Number:1	Title: Moving beyond gradient descent	No. of hours: 8
Content Summary: Local minima vs global minima vs saddle, model identifiability, correcting gradientpoints in wrong directions, Momentum based optimization, second order methods, learning rate adaption, adagrad, rmsprop, adam		
Unit Number:2	**Title: Convolutional Neural Network**	**No. of hours: 8**
Content Summary: Convolution operation, filters and feature maps, motivation, sparse interactions, parameter sharing and equivariant representation, padding and stride, max pooling, full architectural description of convolutional network, build cnn using dataaugmentation, using pretrained convnet, visualize what convnet learn		
Unit Number:3	**Title: Embedding and Representation Learning:**	**No. of hours: 8**
Content Summary: Principle component analysis, working with text data, one-hot encoding of words and characters, word embedding, autoencoder architecture, denoising, sparsity, Word2vec framework, Skip-Gram architecture		
Unit Number:4	**Title: Models for Sequence Analysis**	**No. of hours: 8**
Content Summary: Analysing Variable-length inputs, Seq2seq with neural n-gram, part of speech tagger, dependency parse, syntaxnet, recurrent neural network, challenges with vanishinggradients, long short term memory units		

Textbooks:

1. Deep Learning with Python by Francois Chollet - Manning Publications; 1 edition

2. Deep Learning by Ian Goodfellow, Yoshua Bengio, Aaron Courville, Francis Bach - MIT Press (3 January 2017)

CO-PO Mapping

- indicate no co-relation between CO and PO/PSO,

1 indicates the strength of co-relation between CO and PO/PSO is Weak/low,

2= strength of co-relation between CO and PO/PSO is Moderate/Medium,

3= strength of co-relation is Strong/High.

Proposed Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1.	Implement fence row & Column Transformation.	CO1
2.	Build tensorflow estimator and data pipeline	CO2
3.	Build a regression model on a real dataset (the Boston housing price dataset)	CO2
4.	Build a classification model on a real dataset (Titanic dataset)	CO2
5.	Build deep neural networks for single and multipleinputs.	CO2
6.	Installation of keras and simple keras program	CO2
7.	MNIST using keras- build data pipeline, plot training and validation accuracy	CO2
8.	Write a program to load data set with parameter split, shuffle_files, with_info=True, as_supervised=True	CO2
9.	Write a program to convert tf.data.Dataset objects to pandas.DataFrame with tfds.as_dataframe	CO2
10.	Write a program to Build Training Pipeline using ds.map, ds.cache, ds.shuffle, ds.batch, ds.prefetch	CO2
11.	Write a program to build and train CNN	CO3
12.	Write a program to access overfitting and underfittingin CNN	CO3
13.	Write a program to implement padding, stride and max pooling	CO3
14.	Write a program to implement one-hot encoding	CO3
15.	Write a program to implement word embedding	CO3
16.	Write a program to implement Word2vec	CO3
17.	Write a program to implement RNN	CO3
18.	Write a program to implement LSTM	CO3

(DEPARTMENT ELECTIVE-I)
IMAGE PROCESSING & COMPUTER VISION

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Image Processing & Computer Vision	**Course Code**	**L-T-P**	**Credits**
	ENSP304	4-0-0	4
Type of Course:	DSE-1		
Pre-requisite(s), if any: (1) Linear Algebra and (2) programming in python			

Course Perspective. This course introduces students to the essential concepts and techniques of image processing and computer vision. It bridges theoretical knowledge with practical applications, enabling students to understand and implement various algorithms for image enhancement, restoration, segmentation, and object recognition. The course is designed to equip students with the skills required to tackle real-world challenges in fields such as robotics, medical imaging, surveillance, and multimedia applications.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Remembering** key concepts, definitions, and algorithms in image processing and computer vision.
CO 2	**Understanding** the principles and applications of essential techniques like edge detection and feature extraction.
CO 3	**Applying** Use image processing methods to enhance digital images and analyze the effects.
CO 4	**Evaluating** the effectiveness of various computer vision models in different contexts
CO 5	**Designing** and implement projects that integrate multiple image processing algorithms to solve complex problems.

Course Outline:

Unit Number: 1	Title: Title: Introduction to Basic Concepts of Image Formation	No. of hours: 10

Content Summary:

Fundamentals and Applications of Image Processing:

- Overview of Image Processing and its applications.
- Components of Image Processing Systems.

Image Sensing and Acquisition:

- Image sensing techniques.
- Image acquisition methods.

Sampling and Quantization:

- Concept of sampling and quantization.
- Neighbors of pixel adjacency and connectivity.

- Regions and boundaries.
- Distance measures.

Image Enhancement:

- Frequency and Spatial Domain techniques.
- Contrast Stretching.
- Histogram Equalization.
- Low pass and High pass filtering.

Unit Number: 2	Title: Image Restoration and coloring	No. of hours: 12

Content Summary:

Image Restoration:

- Model of the Image Degradation/Restoration Process.
- Noise Models.
- Restoration in the presence of noise using spatial filtering.
- Periodic Noise Reduction using frequency domain filtering.
- Linear Position Invariant Degradations.
- Estimation of Degradation Function.
- Inverse Filtering.
- Wiener Filtering.
- Constrained Least Square Filtering.
- Geometric Mean Filter.
- Geometric Transformations.

Color Image Processing:

- Color models and transformations.
- Image Segmentation using color.
- Texture Descriptors.
- Color Features.
- Edge/Boundary detection.

- Object Boundary and Shape Representations.
- Interest or Corner Point Detectors.
- Speeded-Up Robust Features (SURF).
- Saliency detection.

Unit Number: 3	Title: Image Compression and Segmentation	No. of hours: 10

Content Summary:

Image Compression:

- Data Redundancies.
- Image Compression models.
- Elements of Information Theory.
- Lossless and Lossy Compression.
- Huffman Coding.
- Shanon-Fano Coding.
- Arithmetic Coding.
- Golomb Coding.
- LZW Coding.
- Run Length Coding.
- Lossless Predictive Coding.
- Bit Plane Coding.
- Image compression standards.

Image Segmentation and Morphological Image Processing:

- Discontinuity-based segmentation.
- Similarity-based segmentation.
- Edge linking and boundary detection.
- Thresholding.
- Region-based Segmentation.
- Introduction to Morphology.

- Dilation and Erosion.
- Basic Morphological Algorithms.

Unit Number: 4	Title: Object Representation and Computer Vision Techniques	No. of hours: 8

Content Summary:

Representation and Description:

- Introduction to Morphology.
- Basic Morphological Algorithms.
- Representation Techniques.
- Boundary Descriptors.
- Regional Descriptors.
- Chain Code.
- Structural Methods.

Computer Vision Techniques:

- Review of Computer Vision Applications.
- Artificial Neural Networks for Pattern Classification.
- Convolutional Neural Networks (CNNs).
- Machine Learning Algorithms and their Applications

Learning Experiences:

1. **Lectures and Interactive Discussions:**
    - Instructor-led lectures covering fundamental concepts of image processing and computer vision.
    - Interactive discussions to deepen understanding of complex topics and encourage student engagement.
2. **Hands-on Programming Labs:**
    - Practical sessions where students implement image processing algorithms using Python.

- Guided exercises on using libraries such as OpenCV and PIL for image manipulation and analysis.

3. **Group Projects:**
    - Collaborative projects that require students to apply multiple image processing techniques to solve real-world problems.
    - Peer reviews and presentations to develop communication skills and critical thinking.
4. **Workshops:**
    - Specialized workshops focused on specific techniques, such as edge detection or object recognition, including hands-on coding activities.
5. **Quizzes and Assessments:**
    - Regular quizzes to assess understanding of theoretical concepts and practical skills.
    - Assignments involving the application of image processing methods to enhance or analyze images.

Outside Classroom Learning

1. **Self-Directed Learning:**
    - Access to online resources, tutorials, and video lectures for additional study on image processing topics.
    - Recommended readings from textbooks, research papers, and online articles.
2. **Online Communities and Forums:**
    - Participation in forums such as Stack Overflow or GitHub to seek help and share knowledge with peers and professionals.
    - Engagement in discussions about current trends and advancements in image processing and computer vision.
3. **Real-World Applications and Projects:**
    - Opportunities to work on internships or research projects that apply image processing techniques in real-world contexts.
    - Participation in hackathons or competitions (e.g., Kaggle) focused on image analysis challenges.

Textbooks

1. Gonzalez Rafael C. and Woods Richard E., Digital Image Processing, New Delhi: Prentice– Hall of India.
2. Computer Vision: Algorithms and Applications by Richard Szeliski

Online Learning References:

I) **Fast.ai: Practical Deep Learning for Coders**
   a. Practical course on deep learning, including applications in image processing and computer vision.
   b. Link: Fast.ai - Practical Deep Learning for Coders

II) **Deep Learning for Computer Vision with Python by Adrian Rosebrock**
   a. A comprehensive book on deep learning techniques for computer vision applications.
   b. Link: PyImageSearch - Deep Learning for Computer Vision with Python

III) **GitHub: OpenCV Projects and Tutorials**
   a. Repository of projects and tutorials on OpenCV, a popular library for computer vision.
   b. Link: GitHub - OpenCV

IV) **Towards Data Science: Image Processing Tutorials**
   a. A collection of tutorials on various image processing techniques and applications.
   b. Link: Towards Data Science - Image Processing

V) **IEEE Xplore Digital Library: Image Processing and Computer Vision Papers**
   a. Access to research papers and articles on the latest developments in image processing and computer vision.
   b. Link: IEEE Xplore Digital Library

IMAGE PROCESSING & COMPUTER VISION LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **Image Processing & Computer Vision Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP354	0-0-2	1
Type of Course:	DSE-1		

Pre-requisite(s), if any: (1) Linear Algebra and (2) programming in python

Defined Course Outcomes

COs	**Statements**
CO1	**Applying** image processing techniques using Python libraries.
CO2	**Analyzing** and evaluate the effectiveness of different image enhancement algorithms.
CO3	**Implementing** image restoration algorithms and evaluate their performance in the presence of noise.
CO4	**Developing** image compression algorithms and analyze their impact on image quality.

CO5	**Formulating** computer vision techniques such as object detection and tracking, gesture recognition, and facial expression recognition using Python.

Lab Experiments

Experiment

Implement a program to acquire and display an image. Demonstrate the process of image sensing and acquisition, and explain the components involved in an image processing system

Develop a program to perform sampling and quantization on a given image. Visualize the effects of different sampling rates and quantization levels on image quality.

Implement image enhancement techniques in the spatial domain, including contrast stretching and histogram equalization. Compare the results before and after enhancement.

Apply frequency domain filtering to an image. Implement both low pass and high pass filters, and demonstrate their effects on the image in terms of noise reduction and edge enhancement.

Implement a noise model to simulate different types of noise (Gaussian, salt-and-pepper) in an image. Apply spatial filtering techniques to restore the image from the noisy version.

Develop a program to perform periodic noise reduction using frequency domain filtering. Implement and compare inverse filtering and Wiener filtering for image restoration

Implement geometric transformations on an image, such as rotation, scaling, and translation. Demonstrate how these transformations affect the image quality and geometry.

Perform color image processing by converting an image from RGB to another color model (e.g., HSV, YCbCr). Implement and visualize color-based image segmentation techniques.

Implement Huffman Coding and Run Length Coding for image compression. Compare the compression ratios and efficiency of these techniques on different types of images.

Develop a program to perform image segmentation using edge detection techniques. Implement edge linking and boundary detection algorithms to segment objects within an image.

Apply morphological operations, such as dilation and erosion, on a binary image. Implement basic morphological algorithms to enhance the structure and features of the objects in the image.

Implement threshold-based and region-based segmentation techniques. Compare their effectiveness in segmenting different types of images and objects.

Implement boundary and regional descriptors to represent the shape and features of objects in an image. Use chain codes and structural methods to describe object

Develop a basic convolutional neural network (CNN) for image classification. Train the CNN on a dataset and evaluate its performance in recognizing different classes of images.

Implement a motion estimation algorithm to track moving objects in a video sequence. Demonstrate object tracking by visualizing the trajectories of the tracked objects.

Create a system for face and facial expression recognition using machine learning algorithms. Implement feature extraction techniques and train a classifier to recognize different expressions and identities.

INTRODUCTION TO GENERATIVE AI

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Introduction to Generative AI	**Course Code**	**L-T-P**	**Credits**
	ENSP306	4-0-0	4
Type of Course:	DSE-1		
Pre-requisite(s), if any:			

Course Perspective. This course provides an in-depth introduction to the principles, techniques, and applications of Generative Artificial Intelligence (AI). Generative AI is a rapidly evolving field that has the potential to transform numerous industries by enabling machines to create content, predict outcomes, and enhance decision-making processes. This course will cover foundational concepts, the latest advancements in generative models, and practical applications, ensuring students gain a comprehensive understanding of the subject.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** generative AI principles, recent advancements, and applications.
CO 2	**Applying** probability theory and statistics in generative AI tasks.
CO 3	**Designing** deep learning models for generative tasks..
CO 4	**Evaluating** generative models for specific applications.
CO 5	Assessing ethical implications and propose solutions for generative AI.

Course Outline:

Unit Number: 1	**Title: Foundations of Generative AI**	**No. of hours: 10**

Content: ☐ **Introduction to Generative AI:** Definition, working principles, and recent advancements. ☐ **Generative Modeling:** • Overview of Generative vs. Discriminative Models. • Introduction to Probabilistic Generative Models. • Naive Bayes as a basic generative model. • Challenges in Generative Modeling. • Introduction to Representation Learning.		
Unit Number: 2	**Title: Deep Learning**	**No. of hours: 10**
Content: ☐ Overview of Structured and Unstructured Data. ☐ Introduction to Deep Neural Networks using Keras and TensorFlow. ☐ Building and Training Deep Neural Networks: • Loading, building, compiling, training, and evaluating models. • Techniques to improve models: Convolutional layers, Batch normalization, and Dropout layers. ☐ Introduction to Autoencoders: • Building and understanding Variational Autoencoders (VAEs). • Using VAEs for tasks like face generation.		
Unit Number: 3	**Generative Adversarial Networks**	**No. of hours: 10**
Content: ☐ **Introduction to GANs:** • Roles of the Discriminator and Generator. • Training processes and challenges in GANs. ☐ **Advanced GAN Techniques:** • Wasserstein GAN (WGAN) and WGAN-GP.		

☐ **Evaluation of GANs:** • Qualitative and Quantitative methods. ☐ **GAN Architectures and Applications.**.		
Unit Number: 4	**Applications and Future Directions**	**No. of hours: 10**
Content: ☐ **Real-World Applications of Generative AI:** • Image synthesis, data augmentation, healthcare, gaming, and art. ☐ **Ethical Considerations:** • Addressing bias, fairness, deepfakes, and responsible AI practices. ☐ **Emerging Trends:** • Overview of reinforcement learning, meta-learning, and tools like OpenAI's DALL-E.		

Learning Experiences:

Inside Classroom Learning

1. **Lectures and Interactive Discussions:**
   - Instructor-led sessions covering foundational principles and recent advancements in Generative AI.
   - Interactive discussions that encourage students to ask questions, share insights, and deepen their understanding of key concepts.
2. **Hands-on Programming Labs:**
   - Practical labs focused on implementing deep learning models using frameworks like Keras and TensorFlow.
   - Activities such as building Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs) to solidify theoretical knowledge through hands-on experience.
3. **Group Projects:**

- Collaborative projects where students work in teams to design and implement generative AI solutions for specific applications.
- Peer presentations and feedback sessions to enhance communication skills and foster teamwork.

4. **Workshops:**
    - Specialized workshops on advanced topics, such as GAN training techniques and ethical considerations in Generative AI.
    - Hands-on coding sessions where students can experiment with different generative models and techniques.
5. **Quizzes and Assessments:**
    - Regular quizzes to evaluate understanding of theoretical concepts and practical applications.
    - Assignments that involve analyzing and evaluating generative models based on provided datasets.

Outside Classroom Learning

1. **Self-Directed Learning:**
    - Encouragement to explore online resources, tutorials, and research articles to deepen knowledge of Generative AI concepts.
    - Access to MOOCs (Massive Open Online Courses) on related topics for additional learning.
2. **Online Communities and Forums:**
    - Participation in platforms like GitHub and Stack Overflow for coding assistance, project collaboration, and community support.
    - Engagement in discussions about Generative AI trends and sharing experiences with peers.
3. **Real-World Applications and Projects:**
    - Opportunities for internships or research projects that involve implementing generative AI solutions in industry settings.
    - Encouragement to enter competitions or hackathons focused on Generative AI challenges.

Textbooks

1. Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play by David Foster
2. Hands-On Generative Adversarial Networks with Keras: Create Powerful Neural Networks for Real-World Applications by Rafael Valle

References

1. Generative Deep Learning, by David Foster, 2nd Edition, O'Reilly Media, Inc.
2. Deep Learning by Ian Goodfellow, Yoshua Bengio and Aaron Courville, The MIT Press
3. Pattern recognition and machine learning by Christopher M. Bishop

INTRODUCTION TO GENERATIVE AI LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Introduction to Generative AI Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP356	C	1

-
0
-
2

Type of Course: DSE-1

Pre-requisite(s), if any: NA

Defined Course Outcomes

COs	
CO1	**Designing** develop, and evaluate deep neural networks for complex image classification tasks using advanced model improvement techniques.
CO2	**Implementing** and analyze variational autoencoders (VAEs) for anomaly detection in network traffic data, focusing on security threat identification.
CO3	**Developing** and enhance generative adversarial networks (GANs) for realistic image synthesis, emphasizing model architecture and performance improvement.
CO4	**Utilizing** generative AI techniques for data augmentation in healthcare and other creative fields, assessing the impact on model performance and addressing ethical considerations.

Lab Experiments

S. N	Experiments	COs
1	Implement a basic probabilistic generative model using the Naive Bayes classifier. Train the model on a simple dataset and evaluate its performance. Discuss the challenges faced during the implementation and how they were addressed.	CO1
2	Develop a representation learning model to learn and visualize the latent features of a given dataset. Implement the model using a suitable machine learning framework and analyze the results.	CO1
3	Create a generative modeling framework to generate synthetic data from a given dataset. Compare the performance of generative versus discriminative modeling approaches on the same dataset.	CO1
4	Implement a simple generative model to generate new data points from a probabilistic distribution. Discuss the working principles of generative AI and recent advancements in the field.	CO1
5	Build and train a deep neural network using Keras and TensorFlow to classify images from the MNIST dataset. Evaluate the model's performance and improve it using techniques like convolutional layers, batch normalization, and dropout layers.	CO2
6	Implement an autoencoder and a variational autoencoder (VAE) to compress and reconstruct images from a given dataset. Analyze the differences between	CO2

	the autoencoder and VAE in terms of performance and latent space representation.	
7	Design and implement a convolutional neural network (CNN) to classify images from the CIFAR-10 dataset. Train the model, evaluate its performance, and use techniques like data augmentation to improve its accuracy.	CO3
8	Implement a basic GAN to generate synthetic images. Train the GAN on a simple image dataset, and evaluate the quality of the generated images using both qualitative and quantitative methods.	CO3
9	Develop a Wasserstein GAN (WGAN) and train it on a dataset of images. Compare the performance of the WGAN with a standard GAN in terms of stability and quality of generated images.	CO3
10	Implement and compare different GAN architectures and loss functions. Train each GAN on the same dataset and evaluate their performance using qualitative and quantitative methods.	CO3
11	Implement a generative AI model for image synthesis. Train the model on a dataset of images and evaluate the quality of the synthesized images. Discuss the potential applications of image synthesis in various fields.	CO4
12	Develop a generative AI model for data augmentation and data generation. Train the model on a dataset with limited data and analyze how data augmentation improves the performance of a classifier trained on the augmented dataset.	CO4
1	Create a generative AI application for healthcare, such	C

3	as generating synthetic medical images for training purposes. Discuss the ethical considerations and challenges associated with using generative AI in healthcare.	O4

TRANSFER LEARNING

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Transfer Learning	**Course Code**	**L-T-P**	**Credits**
	ENSP308	4-0-0	4
Type of Course:	DSE-1		
Pre-requisite(s), if any:			

Course Perspective. This course introduces students to the fundamental concepts and advanced techniques of transfer learning, an essential area in machine learning and deep learning. Transfer learning focuses on leveraging knowledge gained from one domain to improve learning in another domain. This course covers theoretical foundations, practical implementations, and applications across various domains, providing students with the skills necessary to apply transfer learning to real-world problems.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the theoretical foundations, motivations, and applications of transfer learning.
CO 2	**Implementing** transfer learning algorithms proficiently using Python and deep learning libraries.
CO 3	**Applying** fine-tuning, feature extraction, and model adaptation techniques effectively across different domains and tasks.

CO 4	**Evaluating** transfer learning models using standard metrics and methodologies.
CO 5	**Analyzing** real-world applications and ethical implications of transfer learning for inclusive and meaningful solutions.

Course Outline:

Unit Number: 1	Title: Foundations of Transfer Learning	No. of hours: 10

Content Summary:

Introduction to Transfer Learning:

- Overview of AI, Machine Learning, and Transfer Learning
- Relationship to Existing Machine Learning Paradigms
- Fundamental Research Issues and Applications

Instance-Based Transfer Learning:

- Instance-Based Noninductive Transfer Learning
- Instance-Based Inductive Transfer Learning

Feature-Based Transfer Learning:

- Minimizing Domain Discrepancy
- Learning Universal Features
- Feature Augmentation

Model-Based Transfer Learning:

- Transfer through Shared Model Components
- Transfer through Regularization

Unit Number: 2	Title: Advanced Transfer Learning Techniques	No. of hours: 10

Content Summary:

Relation-Based Transfer Learning:

- Markov Logic Networks (MLNs)
- Relation-Based Transfer Learning with MLNs

Heterogeneous Transfer Learning:

- Problem Definition and Methodologies
- Applications of Heterogeneous Transfer Learning

Adversarial Transfer Learning:

- Generative Adversarial Networks (GANs)
- Transfer Learning with Adversarial Models

Transfer Learning in Reinforcement Learning:

- Inter-task and Inter-domain Transfer Learning

Unit Number: 3	Title: Multi-task and Theoretical Aspects of Transfer Learning	No. of hours: 12

Content Summary:

Multi-task Learning:

- Supervised, Unsupervised, and Semi-supervised Learning
- Active, Reinforcement, and Online Learning
- Multi-view and Distributed Multi-task Learning

Transfer Learning Theory:

- Generalization Bounds for Multi-task Learning
- Generalization Bounds for Supervised and Unsupervised Transfer Learning

Transitive Transfer Learning (TTL):

- TTL over Mixed Graphs
- TTL with Hidden Feature Representations and Deep Neural Networks

AutoTL: Learning to Transfer Automatically:

- The L2T Framework

- Parameterizing and Inferring What to Transfer
- Connections to Other Learning Paradigms

Unit Number: 4	Title: Applications of Transfer Learning	No. of hours: 8

Content Summary:

Specialized Learning Techniques:

- Few-Shot, Zero-Shot, and One-Shot Learning
- Bayesian Program Learning and Poor Resource Learning

Transfer Learning Applications:

- Computer Vision and Medical Image Analysis
- Natural Language Processing and Sentiment Analysis
- Dialogue Systems and Spoken Language Understanding
- Natural Language Generation

Learning Experiences:

Inside Classroom Learning

1. **Lectures and Discussions:**
    - **Content Delivery:** Engaging lectures on the theoretical foundations and motivations of transfer learning, including concepts like instance-based and feature-based transfer learning.
    - **Interactive Q&A:** Opportunities for students to ask questions and discuss concepts, enhancing understanding through peer interaction.
2. **Hands-on Programming Labs:**
    - **Practical Implementation:** Students implement transfer learning algorithms using Python and libraries such as TensorFlow and PyTorch.
    - **Model Fine-tuning:** Exercises on fine-tuning pre-trained models, allowing students to gain practical experience.
3. **Group Projects:**

- **Collaborative Learning:** Team-based projects where students apply transfer learning techniques to real-world datasets or specific problems.
- **Peer Review:** Presentations of projects to classmates for feedback and improvement.

4. **Quizzes and Assessments:**
    - **Knowledge Checks:** Regular quizzes to assess understanding of theoretical concepts and practical skills.
    - **Assignments:** Tasks requiring evaluation of transfer learning models using standard metrics and methodologies.

Outside Classroom Learning

1. **Self-Directed Learning:**
    - **Online Resources:** Encouragement to explore MOOCs, tutorials, and research articles related to transfer learning and machine learning.
    - **Study Groups:** Forming study groups to discuss and review material, fostering a collaborative learning environment.
2. **Real-World Applications:**
    - **Internships and Research Projects:** Opportunities to work on transfer learning projects in industry settings, gaining hands-on experience.

Textbook

1. "Transfer Learning" by Qiang Yang, Yu Zhang, Wenyuan Dai, Sinno Jialin Pan

Reference Books:

1. "Deep Learning" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville
2. "Transfer Learning in Action" by Yuxi (Hayden) Liu
3. "Transfer Learning for Natural Language Processing" by Paul Azunre
4. "Deep Learning for Computer Vision" by Rajalingappaa Shanmugamani
5. "Hands-On Transfer Learning with Python" by Dipanjan Sarkar and Raghav Bali

Additional Readings:

I) **Open Source Projects:**

a. TensorFlow Hub: GitHub Repository

b. PyTorch Hub: GitHub Repository

II) **Community Forums and Discussions:**

a. Reddit: r/MachineLearning

b. Stack Overflow: Transfer Learning Tag

III) **OSSU Link:**

Open Source Society University - Data Science Curriculum

TRANSFER LEARNING LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Transfer Learning Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP358	0-0-2	1
Type of Course:	DSE-1		
Pre-requisite(s), if any: NA			

Defined Course Outcomes

COs	**Statement**
CO1	**Applying** and evaluate basic transfer learning models and techniques to solve image and text classification tasks using pre-trained models.
CO2	**Implementing** and analyze advanced transfer learning techniques, including relation-based, adversarial, and heterogeneous transfer learning, to improve model performance across different domains.
CO3	**Developing** and assess multi-task learning models and theoretical aspects of transfer learning, focusing on improving model generalization and performance in multi-domain tasks.
C	**Utilizing** transfer learning for practical applications, including few-

O 4	shot learning, zero-shot learning, sentiment analysis, and privacy-preserving techniques, to address real-world problems effectively.

Lab Experiments

Experiment	CO
Implement an instance-based transfer learning model using a simple dataset to demonstrate the concepts of noninductive and inductive transfer learning.	CO1
Develop a feature-based transfer learning model to minimize domain discrepancy and learn universal features. Use a dataset with different domains to showcase the effectiveness of feature augmentation.	CO1
Create a model-based transfer learning application by sharing model components and applying regularization techniques. Evaluate the performance of the model on a target task with limited data.	CO1
Compare the performance of a naive machine learning model with a transfer learning model on the same dataset. Discuss the advantages and challenges of transfer learning in this context.	CO1
Implement a relation-based transfer learning model using Markov Logic Networks (MLNs). Apply the model to a dataset	CO

with relational data and evaluate its performance.	2
Develop a heterogeneous transfer learning application that addresses the challenges of transferring knowledge between different feature spaces or distributions. Demonstrate the application on a real-world dataset.	CO2
Create an adversarial transfer learning model using GANs. Train the model on a source domain and evaluate its ability to generate or classify data in a target domain.	CO2
Implement transfer learning in a reinforcement learning context by transferring knowledge from one task to another. Compare the performance of the transfer learning model with a model trained from scratch.	CO2
Develop a multi-task learning model that simultaneously trains on multiple tasks. Evaluate the performance improvements achieved through shared learning compared to individual task training.	CO3
Create a transitive transfer learning model using mixed graphs and hidden feature representations. Apply the model to a dataset with multiple related tasks and analyze the results.	CO3
Implement transfer learning models for various applications, such as computer vision, medical image analysis, and natural language processing. Compare the effectiveness of transfer learning across these different domains	CO4

MINOR PROJECT-III

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Minor Project-III	**Course Code**	**L-T-P**	**Credits**
	ENSI352	---	2
Type of Course:	Proj-3		
Pre-requisite(s), if any: NA			

Duration:

The minor project will last for three months.

Project Requirements:

1. **Problem Identification and Analysis:**
   - Identify a relevant problem in society or industry.
   - Conduct a thorough analysis of the problem, considering various perspectives and implications.
2. **Implementation:**
   - Develop and implement a solution to address the identified problem.
3. **Data Visualization:**
   - Utilize appropriate data visualization techniques to represent the problem, solution, and outcomes effectively.
4. **Presentation of Solutions:**
   - Prepare a comprehensive presentation of the implemented solution, including its development process, outcomes, and impact.
5. **Case Studies:**
   - Conduct case studies related to the problem and solution, analyzing existing examples and drawing relevant insights.

Guidelines:

1. **Project Selection:**
   - Choose a societal or industrial problem relevant to the field of computer science and engineering.
   - Ensure the problem is specific and well-defined.
2. **Literature Review:**
   - Conduct a thorough review of existing literature and solutions related to the problem.
   - Identify gaps in existing solutions and potential areas for further investigation.
3. **Implementation:**
   - Develop a detailed plan for implementing the solution.
   - Execute the implementation using appropriate tools, technologies, and methodologies.
4. **Data Visualization:**
   - Collect relevant data and use visualization techniques to represent the problem, solution, and outcomes.
   - Ensure the visualizations are clear, accurate, and effectively communicate the information.
5. **Documentation:**
   - Document the entire process, including problem identification, literature review, implementation, data visualization, and case studies.
   - Use appropriate formats and standards for documentation.
6. **Presentation:**
   - Prepare a presentation summarizing the problem, existing solutions, implementation process, data visualization, and case studies.
   - Ensure the presentation is clear, concise, and well-structured.

Evaluation Criteria for Minor Project (Out of 100 Marks):

1. **Problem Identification and Analysis (15 Marks):**
   - Comprehensive identification and analysis of the problem: 15 marks
   - Good identification and analysis of the problem: 12 marks

- Basic identification and analysis of the problem: 9 marks
- Poor identification and analysis of the problem: 5 marks
- No identification and analysis of the problem: 0 marks

2. **Implementation (30 Marks):**
    - Successful and thorough implementation: 30 marks
    - Good implementation: 25 marks
    - Moderate implementation: 20 marks
    - Basic implementation: 15 marks
    - Poor implementation: 10 marks
    - No implementation: 0 marks
3. **Data Visualization (20 Marks):**
    - Effective and clear data visualization: 20 marks
    - Good data visualization: 15 marks
    - Moderate data visualization: 10 marks
    - Basic data visualization: 5 marks
    - Poor data visualization: 0 marks
4. **Presentation of Solutions (15 Marks):**
    - Clear, concise, and engaging presentation: 15 marks
    - Clear but less engaging presentation: 12 marks
    - Somewhat clear and engaging presentation: 9 marks
    - Unclear and disengaging presentation: 5 marks
    - No presentation: 0 marks
5. **Case Studies (20 Marks):**
    - Comprehensive and insightful case studies: 20 marks
    - Good case studies: 15 marks
    - Moderate case studies: 10 marks
    - Basic case studies: 5 marks
    - Poor case studies: 0 marks

Total: 100 Marks

Course Outcomes:

By the end of this course, students will be able to:

1. **Identify and Analyze Problems:**
    - Identify relevant societal or industrial problems and conduct a thorough analysis of these problems.
2. **Implement Solutions:**
    - Develop and implement effective solutions to address identified problems using appropriate tools and technologies.
3. **Visualize Data:**
    - Utilize data visualization techniques to represent problems, solutions, and outcomes clearly and effectively.
4. **Present Solutions:**
    - Prepare and deliver comprehensive presentations summarizing the implementation process, outcomes, and impact of their solutions.
5. **Conduct Case Studies:**
    - Conduct case studies related to the problem and solution, analyzing existing examples and drawing relevant insights.
6. **Literature Review:**
    - Conduct comprehensive literature reviews to identify gaps in existing solutions and potential areas for further investigation.
7. **Documentation:**
    - Document the entire process, including problem identification, literature review, implementation, data visualization, and case studies, using appropriate formats and standards.
8. **Professional Development:**
    - Develop skills in research, analysis, implementation, data visualization, documentation, and presentation, contributing to overall professional growth.

COMPETITIVE CODING BOOTCAMP-IV

Program Name:	**COMPETITIVE CODING BOOTCAMP-IV**		
Course Name: **COMPETITIVE CODING BOOTCAMP-IV**	**Course Code**	**L-T-P**	**Credits**
		2-0-0	0
Type of Course:	AUDIT-4		
Contact Hours	30		
Version			

Course Outcomes

CO1	**Understanding** system design principles and identify functional and non-functional requirements for projects.
CO2	**Applying** scaling and load balancing techniques and evaluate caching strategies for system optimization.
CO3	**Designing** efficient database schemas and implement ACID transactions in SQL and NoSQL.
CO4	**Solving** algorithmic problems using advanced techniques and apply string matching algorithms in coding challenges.

<table>
<tr><td>Unit Number: 1</td><td>Title: Foundations and Advanced Concepts in System Design</td><td>No. of hours: 8</td></tr>
<tr><td colspan="3">

Content:

Introduction to System Design

- **Principles of System Design**: Basics of system design: modularity, scalability, and maintainability, **Hands-on**: Design a simple e-commerce or social media system blueprint.
- **Functional vs. Non-Functional Requirements**: Understand key differences, **Hands-on**: Identify functional and non-functional requirements for a simple project.

Scalability and Load Balancing

- **Scaling Techniques**: **Vertical Scaling**: Adding resources to a single server, **Horizontal Scaling**: Distributing load across multiple servers, **Hands-on**: Set up vertical and horizontal scaling scenarios using cloud tools.
- **Load Balancing**: Round-robin, least connections, and IP hashing strategies,

Caching Strategies:

- **Client-Side vs. Server-Side Caching**: Comparison of caching techniques.
- **Eviction Policies**: LRU and LFU policies for cache eviction.

</td></tr>
<tr><td>Unit Number: 2</td><td>Title: Advanced Database concepts</td><td>No. of hours: 8</td></tr>
<tr><td colspan="3">

Content:

Database Indexing: different types of indexes (e.g., B-tree, hash, bitmap), how indexes improve query performance, create indexes and their impact on write operations.

Database Transactions: ACID properties (Atomicity, Consistency, Isolation, Durability), implementing transactions in both SQL and NoSQL databases, scenarios like rollbacks and savepoints**.**

</td></tr>
</table>

Database Sharding: partitioning techniques, sharding, sharding strategies for scalability. **Data Modeling:** entity-relationship diagrams (ERDs), Normalize data models based on business requirements, Design efficient schemas for different use cases.		
Unit Number: 3	**Title: Advanced Concepts**	**No. of hours: 8**
Content: **Bit Manipulation**: XOR operations, Bitwise AND, OR, NOT, Counting set bits, Power of two, Bit masking **Divide and Conquer:** Matrix exponentiation, Strassen's algorithm for matrix multiplication, Closest pair of points **Two Pointers:** Fast and slow pointer, Merging sorted arrays, Triplets, pairs with given sum **Sliding Window :** Maximum in a sliding window, Smallest subarray with sum greater than a given value **Union-Find (Disjoint Set Union - DSU) :** Union by rank, Path compression **String Matching:** KMP algorithm, Rabin-Karp, Z-algorithm		
Unit Number: 4	**Title: Miscellaneous**	**No. of hours: 6**
Content: **Hashing: Hash tables, Hash maps and sets, Collision handling, Anagram checks** **Simulation and Design Problems:** LRU cache design, Parking lot simulation, Elevator design, Rate limiter **Concurrency:** Multithreading problems, Deadlock detection **Graphical Algorithms:** Flood fill, Convex hull, Image rendering algorithms		

Lab Experiments

S. No.	Problem Statement	Mapped CO
1	Design a simple e-commerce system with modularity, scalability, and maintainability.	CO1
2	Identify functional and non-functional requirements for a social media platform.	CO1
3	Implement vertical scaling on a single server and measure performance.	CO2
4	Set up horizontal scaling across multiple servers using a cloud platform.	CO2
5	Implement a round-robin load balancer for distributing requests across multiple servers.	CO2
6	Compare client-side and server-side caching strategies for a web application.	CO2
7	Implement Least Recently Used (LRU) cache eviction policy.	CO2

S. No.	Problem Statement	Mapped CO
8	Create a B-tree index for a database to optimize search queries.	CO3
9	Implement ACID-compliant transactions in an SQL database.	CO3
10	Implement database sharding for scalability in a NoSQL database.	CO3
11	Normalize a database schema to 3NF based on given business requirements.	CO3
12	Use bitwise operations to check if a number is a power of two.	CO4
13	Implement matrix multiplication using Strassen’s algorithm.	CO4
14	Find the closest pair of points in a set using divide and conquer.	CO4
15	Merge two sorted arrays using the two-pointer technique.	CO4
16	Find the maximum element in a sliding window of size k.	CO4
17	Solve the union-find problem using path compression and union by rank.	CO4
18	Implement the KMP string matching algorithm to find a pattern in a text.	CO4
19	Implement Rabin-Karp algorithm for string matching in a large document.	CO4
20	Design a hash table with collision handling using chaining.	CO4
21	Implement an anagram checker using hash maps.	CO4
22	Simulate an LRU cache system design.	CO2, CO4
23	Design a rate limiter using a sliding window algorithm.	CO4

S. No.	Problem Statement	Mapped CO
24	Implement deadlock detection using multithreading.	CO4
25	Solve the flood fill problem using depth-first search (DFS).	CO4
26	Implement the convex hull algorithm for a set of 2D points.	CO4
27	Design a simple elevator simulation system with multithreading.	CO4
28	Implement a parking lot simulation with object-oriented design principles.	CO2, CO4
29	Design a system to detect and recover from transaction rollbacks in a database.	CO3
30	Optimize an e-commerce website with horizontal scaling and caching strategies.	CO2

Learning Experiences

Classroom Learning Experience

1. Engagement through Lecture PPTs: Utilize well-structured presentations to convey key concepts of system design, database indexing, and algorithmic techniques.
2. Problem-Based Theory Assignments: Assign real-world challenges like load balancing and caching strategies to enhance problem-solving skills.
3. Project-Based Lab Work: Facilitate hands-on lab assignments where students design and implement system blueprints and databases collaboratively.
4. Comprehensive Question Bank: Provide a diverse question bank covering the syllabus to allow systematic practice for exams and coding interviews.
5. Model Question Papers and Assessments: Conduct continuous assessments through quizzes and coding challenges to test understanding of complex concepts.
6. Support & Feedback System: Offer timely feedback on assignments and projects, with access to instructors for additional support and clarification.

Outside Classroom Learning Experience

1. Use of ICT Tools & Interactive Boards: Host course materials on Moodle LMS for anytime access, utilizing interactive boards for live demonstrations.
2. Video Lectures for Critical Topics: Provide pre-recorded lectures on advanced topics like ACID transactions and multithreading for flexible learning and review.

Text Books:

- Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3rd ed.). MIT Press. ISBN: 978-0262033848.
- McDowell, G. L. (2015). *Cracking the Coding Interview: 189 Programming Questions and Solutions* (6th ed.). CareerCup. ISBN: 978-0984782857.

Online References

- LeetCode (www.leetcode.com)
- HackerRank (www.h System Design Primer (https://github.com/donnemartin/system-design-primer)ackerrank.com)

Semester: VII

(DEPARTMENT ELECTIVE-II)

SECURE CODING AND VULNERABILITIES

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **Secure Coding & Vulnerabilities**	**Course Code**	**L-T-P**	**Credits**
	ENSP301	4-	4

0
-
0

Type of Course: DSE-2

Pre-requisite(s), if any:

Course Perspective. This course provides an in-depth exploration of secure coding practices and the identification and mitigation of common vulnerabilities in software development. Students will gain a solid foundation in security concepts, secure application design, and the implementation of security best practices throughout the software development lifecycle. By understanding the principles of secure coding and the types of vulnerabilities that can compromise applications, students will be equipped to develop robust, secure software. The course covers essential topics such as input validation, authentication, cryptography, buffer overflows, SQL injection, and application security testing. The course is divided into four modules:

a) Introduction to Coding and Security
b) Secure Application Design and Architecture
c) Secure Coding Practices and Vulnerabilities
d) Application Security Testing and Deployment

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** different types of application security threats and their potential impact.
CO 2	**Applying** secure design principles and architectures to develop robust and secure applications.
CO 3	**Implementing** secure coding practices for input validation, authentication, cryptography, session management, and error handling.
CO 4	**Conducting** static and dynamic application security testing to identify vulnerabilities and implement secure deployment and maintenance practices.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Numb er: 1	**Title: Introduction to coding and Security**	**No. of hours: 10**

Content Summary:

Introduction-security concepts-CIA Triad, Viruses, Trojans, and Worms, threat, vulnerability, risk, attack. Coding Standards: Dirty Code and Dirty Compiler, Dynamic Memory Management functions, Common memory management Errors (Initialization Errors, Forget to Check Return Values, accessing already freed memory, Freeing the same memory multiple times, Forget to free the allocated memory), Integer Security –Introduction to integer types: Integer

Data Types, data type conversions, Integer vulnerabilities and mitigation strategies

Unit Number: 2	**Title: Secure Application Design and Architecture**	**No. of hours: 10**

Content Summary:

Security requirements gathering and analysis, Secure software development life cycle (SSDLC), Security issues while writing SRS, Design phase security, Development Phase, Test Phase, Maintenance Phase, Writing Secure Code – Best Practices SD3 (Secure by design, default and deployment), Security principles and Secure Product Development Timeline.

Unit Number: 3	**Title: Secure Coding Practices and Vulnerabilities**	**No. of hours: 10**

Content Summary:

Input validation Techniques-whitelist validation, regular expressions, authentication and authorization, Cryptography, buffer overflows, Session management and protection against session-related attacks, Secure error handling and logging practices, SQL Injection Techniques and Remedies, Race conditions

Unit Number: 4	**Title: Application Security Testing and Deployment**	**No. of hours: 10**

Content Summary:

Security code overview, Secure software installation. The Role of the Security Tester, Building the Security Test Plan. Testing HTTP-Based Applications, Testing File-Based Applications, Testing Clients with Rogue Servers, Static and Dynamic Application Security Testing (SAST & DAST), Secure Deployment and Maintenance, Patch management and software updates, Vulnerability scanning and penetration testing.

Learning Experiences:

Classroom Learning Experience

1. **Hands-on Vulnerability Testing**: Practice identifying and mitigating common software vulnerabilities through hands-on exercises.
2. **Code Review Sessions**: Conduct peer code reviews to spot potential security flaws and enhance secure coding practices.
3. **Case Studies**: Analyze real-world security breaches to understand the exploitation of vulnerabilities.
4. **Interactive Labs**: Implement secure coding techniques such as input validation and buffer overflow protection.
5. **Security Audits**: Perform security audits on sample applications to assess vulnerabilities like SQL injection and session hijacking.
6. **Role Play**: Simulate attacker and defender roles in vulnerability exploitation and mitigation scenarios.

Outside Classroom Learning Experience

1. **Project-Based Learning**: Develop secure applications by applying best practices in secure design, coding, and testing.
2. **Tools Exploration**: Learn to use static and dynamic application security testing tools (SAST & DAST) for real-world applications.
3. **Collaborative Learning**: Work in groups to design security testing plans and assess security risks in various application environments.
4. **Real-World Simulations**: Conduct vulnerability scanning and penetration testing in simulated deployment environments.

Text Books and References

1. Secure Coding: Principles and Practices, Mark G. Graff, Kenneth R. Van Wyk, O'Reilly Media
2. Writing Secure Code, Michael Howard and David LeBlanc,Microsoft Press, 2nd Edition, 2004
3. Buffer Overflow Attacks: Detect, Exploit, Prevent by Jason Deckard ,Syngress,1st Edition, 2005
4. Threat Modeling, Frank Swiderski and Window Snyder,Microsoft Professional, 1st Edition ,2004
5. Secure Coding: Principles and Practices by Mark G. Graff, Kenneth R. van Wyk, Publisher(s): O'Reilly Media, Inc., 2003

6. The Software Vulnerability Guide (Programming Series) by H. Thompson (Author), Scott G. Chase, 2005

Additional Readings:

Online Learning References for "Secure Coding and Vulnerabilities"

1. **OWASP - Secure Coding Practices - Quick Reference Guide**
    - This guide provides a quick reference to secure coding practices based on OWASP's recommendations for secure software development.
    - Link: OWASP - Secure Coding Practices
2. **NPTEL - Secure Coding**
    - Offered by IITs through NPTEL, this course covers secure coding practices and principles for writing secure software.
    - Link: NPTEL - Secure Coding
3. **Mozilla Developer Network (MDN) - Web Security**
    - Comprehensive documentation on web security principles, secure coding practices, and common vulnerabilities in web applications.
    - Link: MDN - Web Security
4. **Google Code University - Web Security**
    - Learn about web security from Google, including secure coding practices and how to protect web applications from common threats.
    - Link: Google Code University - Web Security

SECURE CODING AND VULNERABILITIES LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Secure Coding & Vulnerabilities Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP351	0-0-2	1
Type of Course:	DSE-2		
Pre-requisite(s), if any:			

Lab Experiments

Defined Course Outcomes

COs	
CO1	**Implementing** fundamental security concepts such as the CIA Triad (Confidentiality, Integrity, and Availability) and demonstrate secure coding practices to prevent common vulnerabilities.
CO2	**Analyzing** and fix memory management errors and integer vulnerabilities, applying mitigation strategies to enhance software security.

CO3	**Developing** secure software by following the Secure Software Development Life Cycle (SSDLC), incorporating security principles and best practices throughout the development process.
CO4	**Designing** and test secure applications, performing vulnerability scanning, penetration testing, and implementing security measures to protect against attacks such as SQL injection and buffer overflow.

Ex. No	**Experiment Title**	**Mapped CO/COs**
P1	**Project Title: Secure Memory Management System** **Problem Statement:** Develop a secure memory management system for a critical application such as a healthcare management system. This system should handle dynamic memory allocation and deallocation securely, preventing common memory management vulnerabilities.	CO1
P2	**Project Title: Secure E-commerce Platform Design** **Problem Statement:** Design and implement a secure e-commerce platform that ensures data security throughout the software development life cycle (SDLC). The platform should handle sensitive user information securely and provide a robust security architecture.	CO2
P3	**Project Title: Secure Banking Application** **Problem Statement:** Develop a secure online banking application that ensures the protection of user data and prevents common vulnerabilities such as SQL injection, buffer overflow, and session hijacking.	CO3

P4	**Project Title: Comprehensive Security Testing and Deployment for a Social Media Platform** **Problem Statement:** Develop a social media platform with a focus on security testing and secure deployment. The platform should protect user data and provide a secure environment for social interactions.	CO4

CYBER CRIME INVESTIGATION & DIGITAL FORENSICS

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **Cyber Crime Investigation & Digital Forensics**	**Course Code**	**L-T-P**	**Credits**
	ENSP303	4-0-0	4
Type of Course:	DSE-2		
Pre-requisite(s), if any:			

Course Perspective. The course offers an in-depth exploration of the methodologies and techniques employed in identifying, investigating, and prosecuting cybercrimes. As digital technologies permeate every aspect of modern life, understanding how to safeguard and investigate electronic evidence becomes crucial for ensuring security and justice. This course covers the foundational concepts of digital forensics, types of cybercrimes, investigation procedures, and the utilization of forensic tools. It prepares students to handle and analyze digital evidence proficiently, contributing to the effective enforcement of cyber laws. The course is divided into four comprehensive units:

a) Introduction
b) Types of Cyber Crimes
c) Investigation of Cyber Crimes
d) Forensic Tools and Processing of Electronic Evidence

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the nature and classification of conventional and cyber-crimes.
CO 2	**Analyzing** various types of cyber-crimes and their modes of operation.
CO 3	**Evaluating** the impact of cyber-crimes on individuals, organizations, and society.
CO 4	**Developing** an understanding of digital forensics and the investigative procedures used in cyber-crime cases.
CO 5	**Applying** forensic tools and techniques to retrieve and analyze digital evidence.

Course Outline:

Unit Number: 1	**Title: Title: Introduction**	**No. of hours: 10**

Content:

Introduction to Digital Forensics, Definition and types of cybercrimes, electronic evidence and handling, electronic media, collection, searching and storage of electronic media, introduction to internet crimes, hacking and cracking, credit card and ATM frauds, web technology, cryptography, emerging digital crimes and modules.

Unit Number: 2	**Title: Types of Cyber Crimes**	**No. of hours: 10**

Content:

Crimes targeting Computers: Unauthorized Access Packet Sniffing Malicious Codes including Trojans, Viruses, Logic Bombs, etc. Online based Cyber Crimes: Phishing and its variants Web Spoofing and E-mail Spoofing Cyber Stalking Web defacement financial crimes, ATM and Card Crimes etc. Spamming Commercial espionage and Commercial Extortion online Software and Hardware Piracy Money Laundering Fraud& Cheating Other Cyber Crimes.

Unit Number: 3	**Title: Investigation of Cyber Crimes**	**No. of hours: 10**

Content:

Investigation of malicious applications Agencies for investigation in India, their powers and their constitution as per Indian Laws Procedures followed by First Responders; Evidence Collection and Seizure Procedures of Digital mediums Securing the Scene, Documenting the Scene, Evidence Collection and Transportation Data Acquisition Data Analysis Reporting

Unit Number: 4	**Title: Forensic Tools and Processing of Electronic Evidence**	**No. of hours: 10**

Content:

Introduction to Forensic Tools, Usage of Slack space, tools for Disk

Imaging, Data Recovery, Vulnerability Assessment Tools, Encase and FTK tools, Anti Forensics and probable counters, retrieving information, process of computer forensics and digital investigations, processing of digital evidence, digital images, damaged SIM and data recovery, multimedia evidence, retrieving deleted data: desktops, laptops and mobiles, retrieving data from slack space, renamed file, ghosting, compressed files.

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures and Video Sessions**: Engage with interactive presentations and videos on cyber crime and digital forensics.
2. **Problem-Based Theory Assignments**: Analyze real-world cyber crime scenarios to encourage complex problem-solving.
3. **Project-Based Lab Assignments**: Use forensic tools in hands-on labs to investigate simulated cyber crimes.
4. **Collaborative Group Work**: Work in groups on case studies to promote teamwork and peer learning.
5. **Continuous Assessment and Feedback**: Monitor progress through assessments with regular instructor feedback.

Outside Classroom Learning Experience

1. **Use of ICT Tools and Moodle LMS**: Access course materials via Moodle and use interactive boards for discussions.
2. **Engagement with a Question Bank and Model Papers**: Utilize a question bank and model papers for exam preparation.
3. **Application of Theoretical Knowledge to Practical Scenarios**: Apply theory to practical cases in the full cycle of cyber crime investigations.

Text Books & References

1. Moore, Robert, (2011). Cybercrime, investigating high-technology computer crime(2nd Ed.). Elsevie

2. C. Altheide& H. Carvey Digital Forensics with Open Source Tools, Syngress, 2011.
3. Majid Yar, "Cybercrime and Society", SAGE Publications Ltd, Hardcover, 2nd Edition, 2013.
4. Robert M Slade, "Software Forensics: Collecting Evidence from the Scene of a Digital Crime", Tata McGraw Hill, Paperback, 1st Edition, 2004.

Additional Readings:

Online Learning References:

I) **Cybrary - Digital Forensics**
   a. A free online course that covers various aspects of digital forensics, including tools, techniques, and procedures for investigating cybercrimes.
   b. Link: Cybrary - Digital Forensics

II) **Pluralsight - Digital Forensics Fundamentals**
   a. This course offers a thorough understanding of digital forensics, covering the fundamentals, tools, and techniques used in the field.
   b. Link: Pluralsight - Digital Forensics Fundamentals

III) **SANS Institute - Digital Forensics and Incident Response Blog**
   a. A blog providing insights, case studies, and updates on the latest in digital forensics and incident response.
   b. Link: SANS Institute - Digital Forensics Blog

IV) **OWASP - Open Web Application Security Project**
   a. Provides resources on web security, including best practices for secure coding and tools for vulnerability assessment, which are essential for investigating cybercrimes.
   b. Link: OWASP - Open Web Application Security Project

CYBER CRIME INVESTIGATION & DIGITAL FORENSICS LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **Cyber Crime Investigation & Digital Forensics Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP353	0-0-2	1
Type of Course:	DSE-2		
Pre-requisite(s), if any:			

Proposed Lab Experiments

Defined Course Outcomes

COs	
CO1	**Understanding** the fundamental concepts and principles of digital forensics and cybercrimes.
CO2	**Applying** the knowledge of digital forensics techniques and procedures to collect, analyse, and preserve electronic evidence in various types of cybercrimes.
CO3	**Evaluating** and utilize forensic tools and technologies for data acquisition, analysis, and recovery in the investigation of cybercrimes.
CO4	**Analyzing** and interpret digital evidence obtained from different sources, such as electronic media, internet crimes, malicious applications, and various forms of cybercrimes.

Experiment Title	**Mapped CO/CO**

	s
Project Title: Comprehensive Study on Cybercrime and Digital Forensics **Problem Statement:** Conduct a comprehensive study on various types of cybercrimes and the role of digital forensics in investigating these crimes. The project will involve collecting electronic evidence, understanding cybercrime techniques, and applying digital forensics methodologies.	CO1
Project Title: Simulation and Prevention of Cyber Crimes **Problem Statement:** Develop a comprehensive simulation and prevention strategy for various types of cybercrimes. The project will involve creating scenarios for unauthorized access, phishing, and malware attacks, and implementing preventive measures.	CO2
Project Title: Investigation and Reporting of Cyber Crime Incidents **Problem Statement:** Investigate a simulated cybercrime incident, collect and analyze digital evidence, and report the findings. The project will cover the entire investigation process from securing the scene to data analysis and reporting.	CO3
Project Title: Advanced Digital Forensics and Evidence Processing **Problem Statement:** Develop a system for advanced digital forensics and processing of electronic evidence. The project will involve using forensic tools for data recovery, vulnerability assessment, and processing digital evidence from various devices.	CO4

AI IN CYBER SECURITY

Program Name:	**B. Tech (Computer Science and Engineering)**		
Course Name: AI in Cyber Security	**Course Code**	**L-T-P**	**Credits**
	ENSP305	4-0-0	4
Type of Course:	DSE-2		
Pre-requisite(s), if any:			

Course Perspective. The course delves into the integration of Artificial Intelligence (AI) techniques within the realm of cyber security, highlighting the transformative potential of AI in detecting, preventing, and responding to cyber threats. As cyber threats evolve in complexity and scale, AI offers advanced methodologies to enhance security measures and mitigate risks effectively. This course provides a comprehensive understanding of the applications of AI in cyber security, from fundamental machine learning and deep learning techniques to their practical implementations in threat detection and prevention.

Students will explore the history, evolution, and current trends of AI in cyber security, gaining insights into the ethical considerations and challenges associated with the adoption of AI technologies in this critical field. Through detailed case studies and practical examples, the course bridges theoretical concepts with real-world applications, equipping students with the skills necessary to leverage AI for robust cyber defense strategies.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the concepts and applications of AI in the field of cyber security.
CO 2	**Expressing** the ethical and legal considerations associated with the use of AI in cyber security.
CO 3	**Determining** emerging trends and technologies in AI for cyber security, and their potential impact on the field.
CO 4	**Identifying** strategies for integrating AI-driven solutions into existing cyber security frameworks, policies, and practices.
CO 5	**Articulating** critical thinking and problem-solving skills to address real-world cyber security challenges using AI techniques.
CO 6	**Designing** machine learning techniques for threat detection and prevention in cyber security, including supervised and unsupervised algorithms.

Course Outline:

Unit Number: 1	Title: Introduction to AI and Cyber Security	No. of hours: 10

Content:

Overview of Artificial Intelligence and its applications in Cyber Security

Evolution and impact of AI on Cyber Security

Understanding Cyber Security threats and the role of AI

Basic principles of Machine Learning (ML) and Deep Learning (DL) in Cyber Security

Ethical considerations and challenges of AI in Cyber Security

Unit Number: 2	Title: Machine Learning Techniques for Cyber Security	No. of hours: 10

Content:

- Introduction to Machine Learning techniques relevant to Cyber Security
- Overview of supervised and unsupervised ML models
- Feature engineering and data preparation for ML models
- Practical applications and case studies of ML in Cyber Security.

Unit Number: 3	Title: Deep Learning Techniques for Cyber	No. of hours: 10

Content:

Introduction to Deep Learning and its significance in Cyber Security

Applications of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs)

Overview of Generative Adversarial Networks (GANs) and their use in Cyber Security

Case studies illustrating the use of DL techniques for Cyber Security problems

Unit Numb	Title: AI for Cyber Security: Threat Detection and Prevention	No. of hours: 10

er: 4

Content:

AI applications in threat detection and prevention

Overview of traditional vs. AI-driven threat detection methods

Fundamentals of supervised and unsupervised ML algorithms for threat detection

Advanced deep learning techniques for threat detection (CNNs, RNNs)

Feature selection, emerging trends, and challenges in AI for Cyber Security

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures and Video Sessions**: Engage with presentations and videos on AI in cyber security.
2. **Problem-Based Theory Assignments**: Analyze real-world threats and AI solutions to enhance critical thinking.
3. **Project-Based Lab Assignments**: Implement AI algorithms in labs to detect and mitigate cyber threats.
4. **Collaborative Group Work**: Work in teams on case studies at the intersection of AI and cyber security.
5. **Continuous Assessment and Feedback**: Monitor progress through assessments and receive regular feedback.

Outside Classroom Learning Experience

1. **Use of ICT Tools and Moodle LMS**: Access course materials via Moodle LMS for flexible learning.
2. **Engagement with a Question Bank and Model Papers**: Utilize a question bank and model papers for exam prep.
3. **Application of Theoretical Knowledge to Practical Scenarios**: Apply AI concepts to real-world cyber security cases.

Text Book References

1. Artificial Intelligence for Cybersecurity" by Bhaskar Sinha (Auerbach Publications)
2. Machine Learning and Security: Protecting Systems with Data and Algorithms" by Clarence Chio and David Freeman (O'Reilly Media)

Additional Readings:

Online Learning Resources:

I) **Cybrary - Introduction to Artificial Intelligence for Cyber Security**
   a. This course offers insights into how AI can be applied to cyber security, including threat detection and response.
   b. Link: Cybrary - Introduction to Artificial Intelligence for Cyber Security

II) **Pluralsight - Machine Learning and AI for Cybersecurity**
   a. This course provides an in-depth look at how machine learning and AI can be used to enhance cyber security measures.
   b. Link: Pluralsight - Machine Learning and AI for Cybersecurity

III) **FutureLearn - Artificial Intelligence for Cyber Security by Coventry University**
   a. This course explores the application of AI in cyber security, covering topics like threat detection, response, and mitigation.
   b. Link: FutureLearn - Artificial Intelligence for Cyber Security

IV) **MIT OpenCourseWare - Artificial Intelligence**
   a. Lecture notes, assignments, and exams from MIT's course on Artificial Intelligence, providing a deep dive into AI concepts applicable to cyber security.
   b. Link: MIT OpenCourseWare - Artificial Intelligence

V) **IBM - Introduction to Cyber Security Tools & Cyber Attacks**
   a. A course that covers various cyber security tools and techniques, including the use of AI and machine learning for threat detection and prevention.
   b. Link: IBM - Introduction to Cyber Security Tools & Cyber Attacks

AI IN CYBER SECURITY LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: AI in Cyber Security Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP355	0-0-2	1
Type of Course:	DSE-2		

Pre-requisite(s), if any: basic understanding of web development technologies such as HTML, CSS, and JavaScript. Additionally, students should have some familiarity with networking concepts, operating systems, and databases.

Defined Course Outcomes

COs	Statement
CO1	**Analyzing** the history, evolution, and ethical considerations of AI in cyber security, documenting key milestones and advancements, and discussing the implications of AI applications.
CO2	**Implementing** and evaluate machine learning models for classifying and detecting cyber threats, using various datasets and techniques such as supervised and unsupervised learning, deep learning, and anomaly detection.
CO3	**Developing** and apply feature engineering, data preparation, and model training techniques to enhance the performance and accuracy of cyber security models.
CO4	**Conducting** comprehensive case studies and surveys on the application of AI in cyber security, identifying emerging trends, challenges, and documenting methodologies and findings.

Lab Experiments

Ex.	**Experiment Title**	**Map**

No		ped CO/COs
P 1	**Project Title: Comprehensive Analysis of AI in Cyber Security** **Problem Statement:** Conduct a comprehensive analysis of the role of AI in cyber security. The project will involve studying the history, evolution, and current trends in AI applications for cyber security, and understanding the basic principles of machine learning and deep learning in this context.	CO1
P 2	**Project Title: Machine Learning Models for Cyber Threat Detection** **Problem Statement:** Develop and evaluate different machine learning models for detecting cyber threats. The project will involve implementing supervised and unsupervised learning techniques, performing feature engineering, and analyzing case studies.	CO2
P 3	**Project Title: Deep Learning Models for Advanced Cyber Threat Detection** **Problem Statement:** Develop and evaluate deep learning models for advanced cyber threat detection. The project will involve implementing CNNs, RNNs, and GANs, and analyzing their applications in cyber security.	CO3

P 4	**Project Title: AI-Based Comprehensive Threat Detection System** **Problem Statement:** Develop a comprehensive AI-based system for threat detection and prevention in cyber security. The project will involve implementing machine learning and deep learning models and addressing the challenges of traditional threat detection methods.	C O 4

SOCIAL MEDIA SECURITY

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Social Media Security	**ENSP307**	4-0-0	4
Type of Course:	DSE-2		
Pre-requisite(s), if any:			

Course Perspective. This course introduces students to the critical concepts of social media security, addressing the growing need to understand and manage security and privacy issues in the digital age. Social media platforms have become integral to personal, professional, and commercial interactions, creating a complex landscape of potential security threats and privacy concerns. This course aims to equip students with the knowledge and skills required to navigate and mitigate these risks effectively. Students will explore the technical, legal, and social dimensions of social media security, developing strategies to safeguard personal information, ensure user trust, and comply with legal standards.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Demonstrating** an understanding of the different types of social media platforms, their features, and their impact on communication, marketing, and society.
CO 2	**Acquiring** knowledge and skills in social media monitoring techniques, including data collection, analysis, and the use of relevant tools and technologies.
CO 3	**Developing** the ability to analyze and evaluate viral content on social media, understand the factors contributing to its spread, and recognize its implications for marketing and online engagement.
CO 4	**Identifying** the challenges, opportunities, and pitfalls associated with social media marketing, and formulate strategies for effective audience targeting, engagement, and brand promotion.
CO 5	**Developing** strategies to safeguard personal information, foster user trust, and mitigate associated risks.

Course Outline:

Unit **Title: Social Media Overview** **No. of hours: 10**

Number: 1		

Content Summary:

Overview of Social Media

Types and Platforms

Social Media Monitoring and Analysis

Data Collection and Analysis Methods: BoW Model, TF-IDF

Network Analysis Basics: Node Centrality, Degree Distribution, Clustering Coefficient

Introduction to Synthetic Networks: Random Graphs, Preferential Attachment

Unit Number: 2	**Title: S**ocial Media Management and Marketing	**No. of hours: 10**

Content Summary:

Strategies for Recruitment and Employment Screening

Customer Engagement and Content Management

Evaluating Social Media Campaigns: Effective vs. Ineffective

Ethical Considerations and Privacy in Crowdsourcing

Managing and Promoting Social Media Presence

Unit Number: 3	**Title: Privacy Issues in Social Media**	**No. of hours: 10**

Content Summary:

Privacy Settings and Personal Identifiable Information (PII) Leakage

Types of Privacy Attacks: Identity Disclosure, Inference Attacks, De-anonymization

Privacy Metrics: k-anonymity, l-diversity, Differential Privacy

Balancing Personalization and Privacy

Impact of Privacy on User Trust

Unit	**Title: Social Media Security:**	**No. of hours: 10**

Number: 4	**Laws, Best Practices, and Case Studies**

Content Summary:

Legal Aspects: Posting and Content Regulations

Best Practices: Content Moderation, User Authentication

Security Awareness and Education

Case Studies: Facebook, Twitter, Instagram, YouTube, LinkedIn, and others

Learning Experiences

Classroom Learning Experience

1. **Interactive Lectures and Video Sessions**: Learn about social media security threats through engaging presentations.
2. **Problem-Based Theory Assignments**: Analyze real incidents to enhance critical thinking on social media security.
3. **Project-Based Lab Assignments**: Implement security measures in labs to protect social media accounts.
4. **Collaborative Group Work**: Explore security risks through team case studies on social media platforms.
5. **Continuous Assessment and Feedback**: Receive ongoing assessments and instructor feedback on progress.

Outside Classroom Learning Experience

1. **Use of ICT Tools and Moodle LMS**: Access course materials anytime via Moodle LMS.
2. **Engagement with a Question Bank and Model Papers**: Prepare for exams using a question bank and model papers.
3. **Application of Theoretical Knowledge to Practical Scenarios**: Apply concepts to real-world social media security cases.

References

1. Mastering Social Media Mining, Bonzanini Marco, Packt Publishing Limited

2. Mining the Social Web, Mikhail Klassen and Matthew A. Russell, O'Reilly Media, Inc
3. Social media mining: an introduction, Zafarani, Reza, Mohammad Ali Abbasi, and Huan Liu, Cambridge University Press
4. Social Media Security: Leveraging Social Networking While Mitigating Risk, Michael Cross, Syngress
5. Social Media and the Law: A Guidebook for Communication Students and Professionals, Daxton R. Stewart, Taylor & Francis Ltd
6. Security in the Digital Age: Social Media Security Threats and Vulnerabilities by Henry A. Oliver, Create Space Independent Publishing Platform.

Additional Readings:

Online Learning Resources for Social Media Security

R 1. Coursera - Social Media Marketing Specialization

- **Provider:** Northwestern University
- **Description:** This specialization covers the major social media platforms, marketing strategies, and data analysis tools.
- **Link:** Coursera Social Media Marketing Specialization

R 2. edX - Cybersecurity Fundamentals

- **Provider:** Rochester Institute of Technology
- **Description:** This course offers foundational knowledge in cybersecurity, including threats, vulnerabilities, and defense strategies.
- **Link:** edX Cybersecurity Fundamentals

R 3. Udemy - The Complete Cyber Security Course: Network Security!

- **Instructor:** Nathan House
- **Description:** This comprehensive course covers network security, including how to secure your network, protect your devices, and more.
- **Link:** Udemy Cyber Security Course

SOCIAL MEDIA SECURITY LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Social Media Security Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP357	0-0-2	1
Type of Course:	DSE-2		
Pre-requisite(s), if any:			

Course Outcomes (CO)

COs	Statements
CO1	**Analyzing** different social media platforms, their features, and the ethical and privacy considerations in crowdsourcing and data handling.
CO2	**Implementing** data collection, text analysis, and network analysis techniques on social media datasets, demonstrating proficiency in using APIs and various models.
CO3	**Developing** strategies and plans for using social media in various contexts such as employment screening, customer engagement, and small business promotion.
CO4	**Evaluating** privacy settings, metrics, and security incidents on social media platforms, applying best practices for user authentication, access control, and content moderation.

Lab Experiments

Experiment Title	Mapped CO/COs
Project Title: Comprehensive Analysis of Social Media Platforms **Problem Statement:** Conduct a comprehensive analysis of different social media platforms, their features, and the data they generate. The project will involve collecting and analyzing data from social media, performing content analysis, and understanding network properties.	CO1
Project Title: Effective Social Media Management and Marketing Strategy **Problem Statement:** Develop an effective social media management and marketing strategy for a small business. The project will involve analyzing customer engagement, creating marketing strategies, and addressing ethical considerations in social media use.	CO2

Project Title: Privacy Protection in Social Media **Problem Statement:** Develop strategies and tools to protect user privacy on social media platforms. The project will involve analyzing privacy settings, simulating privacy attacks, and evaluating privacy metrics.	CO3
Project Title: Enhancing Security and Compliance on Social Media Platforms **Problem Statement:** Develop a comprehensive approach to enhance security and ensure compliance with laws on social media platforms. The project will involve researching laws, developing best practices, and analyzing case studies of security incidents.	CO4

(DEPARTMENT ELECTIVE-III)
MOBILE APPLICATION DEVELOPMENT USING IOS

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Mobile Application Development using iOS	**Course Code**	**L-T-P**	**Credits**
	ENSP409	4-0-0	4
Type of Course:	DSE-3		
Pre-requisite(s), if any: Basics of Android			

Course Perspective. The objective of the course is to provide skills to develop applications for OS X and iOS. It includes an introduction to the development framework Xcode. Objective-C is used as a programming language to develop applications. Objective-C is the superset of the C programming language and provides object-oriented capabilities and a dynamic runtime. Objective-C inherits the syntax, primitive types, and flow control statements of C and adds syntax for defining classes and methods. The course is divided into 4 modules:

1. Introduction to IDE and SDK of iOS App Development
2. Swift Programming
3. Encapsulating Data
4. Developing iOS Applications

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the fundamental concepts of variables, constants, and basic data types in SWIFT.
CO 2	**Analyzing** the use of control flow statements such as for, if, and switch in various programming scenarios.
CO 3	**Applying** object-oriented concepts in SWIFT, including the use of classes, structures, and protocols.
CO 4	**Creating** functions, closures, and extensions to enhance code modularity and reuse.
CO5	**Evaluating** error handling techniques and type checking mechanisms to develop robust SWIFT applications

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Introduction to SWIFT Language**	**No. of hours: 10**

Content Summary:

Variables & Constants, Introduction to functions (methods), Arrays, Dictionaries, Data, Date and other basic data types, Enums, structures, closuresFor, If, switch statement, Object oriented concepts with SWIFT, Type check, AnyObject, Any Protocols, Extensions, Error handling, Working with classes

Unit Number: 2	**Title: Working with Xcode**	**No. of hours: 8**

Content Summary:

Introduction to XCODE, COCOA touch framework, iOS application architecture, Application lifecycle

Unit Number: 3	**Title: Introduction to view controllers and Views**	**No. of hours: 12**

Content Summary: View Controllers, view, view lifecycle, Basic Controls – Label, Buttons, Text field, image View, Table view with default cells and customized cells, Collection view with default cells and customized cells, Picker view, Date picker, scroll view, navigation and Tab bar controller, Understanding Interface builder, XIB files, Creating outlets and Actions, Handling touch and gesture events, Segment and Page control, switch view, UIAlertView

Unit Number: 4	Title: Integrating with Database	No. of hours: 10

Content Summary:

Introduction to data storage methods in iOS, Using Core Data, SQLite database, User Defaults, Property List

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures**: Explore iOS app development concepts through engaging presentations.
2. **Problem-Based Assignments**: Analyze real-world scenarios to improve mobile development skills.
3. **Project Labs**: Build and test iOS applications in hands-on lab sessions.
4. **Collaborative Work**: Work in teams on case studies related to mobile app development.
5. **Continuous Feedback**: Receive regular assessments and instructor feedback to track progress.

Outside Classroom Learning Experience

1. **Moodle Access**: Access course materials anytime via Moodle for convenient learning.
2. **Question Bank**: Use a question bank and model papers for exam preparation.
3. **Real-World Applications**: Apply iOS development concepts to practical mobile app scenarios.

References

1. iOS 14 Programming for Beginners: Kickstart your iOS app development journey with the Swift programming language and Xcode 12, 6th Edition, Ahmad Sahar and Craig Clayton

2. Mastering iOS 14 Programming: Build professional-grade iOS applications with Swift 5 and Xcode 12

Additional Readings:

Online Learning Resources for Mobile Application Development Using iOS

- **Apple Developer Documentation**
    - **Description:** Comprehensive documentation and tutorials for iOS app development using Swift and Xcode.
    - **Link:** Apple Developer Documentation
- **Ray Wenderlich: iOS and Swift Tutorials**
    - **Description:** A collection of high-quality tutorials and courses on iOS app development, covering Swift, Xcode, and various iOS frameworks.
    - **Link:** Ray Wenderlich iOS Tutorials
- **GitHub: iOS Development Resources**
    - **Description:** A curated list of open-source projects, libraries, and resources for learning and improving iOS development skills.
    - **Link:** GitHub - iOS Development Resources

MOBILE APPLICATION DEVELOPMENT USING IOS LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Mobile Application Development using iOS Lab	**ENSP459**	0-0-2	1
Type of Course:	DSE-3		

Pre-requisite(s), if any: Basics of Android

Lab Experiments

Defined Course Outcomes

COs	
CO1	**Understanding** and apply fundamental concepts of iOS development using Xcode and the Cocoa Touch framework to build robust and user-friendly applications.
CO2	**Developing** interactive and dynamic user interfaces in iOS applications using view controllers, views, and gesture recognizers.
C	**Creating** and manage user interfaces and view controllers in

O	iOS applications using Xcode, demonstrating proficiency in Interface Builder and UIKit components.
3	
CO 4	**Developing** interactive and dynamic user interfaces in iOS applications using view controllers, views, and gesture recognizers.

Experiment	COs
Set up the iOS development environment by installing Xcode. Create a simple "Hello, World!" iOS application to familiarize with the Xcode IDE and Swift programming basics.	CO1
Develop a basic iOS application that demonstrates the use of Swift syntax, variables, data types, and control flow. Create a simple calculator app to perform basic arithmetic operations.	CO1
Use Xcode and Interface Builder to design a user interface for an iOS app. Create a simple user interface with labels, buttons, and text fields, and handle user interactions.	CO1
Implement a simple iOS app to demonstrate the app lifecycle and navigation between view controllers. Create a multi-screen app that navigates between different views using navigation controllers.	CO1
Design a responsive user interface using Auto Layout and	C

the constraint system. Create an iOS app with a login screen that adjusts to different screen sizes and orientations.	O2
Implement navigation between different views using storyboards and segues. Create a multi-screen app with a main menu and detailed views for each menu item.	CO2
Implement gesture recognition and touch event handling in an iOS app. Create an app that responds to tap, swipe, and pinch gestures to perform different actions.	CO2
Implement data persistence using Core Data. Create an iOS app that allows users to add, edit, and delete notes, and save them to a local database.	CO3
Use User Defaults and the file system to store and retrieve user preferences and data. Create an app that saves user settings and displays them when the app is reopened.	CO3
Implement offline data storage and synchronization. Create an iOS app that allows users to add data while offline and syncs with a remote server when the device is back online.	CO3
Implement advanced UI components and animations in an iOS app. Create a visually appealing app with custom views, animations, and transitions between screens.	CO4
Access and use iOS sensors and hardware features. Create an app that uses the camera to take photos, and the GPS to display the user's current location on a map.	CO4
Debug and test an iOS app using Xcode's debugging tools. Implement unit tests and UI tests to ensure the app functions correctly under different scenarios.	CO4

DEVOPS & AUTOMATION

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: **DevOps & Automation**	**Course Code**	**L-T-P**	**Credits**
	ENSP411	4-0-0	4
Type of Course:	DSE-3		
Pre-requisite(s), if any: Nil			

Course Perspective. Throughout the subject, students will engage in hands-on exercises and projects to gain practical experience with various DevOps tools and practices. By the end of the course, students will be well-equipped to embrace the DevOps culture and apply automation techniques to enhance software development, delivery, and operations processes.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** the principles and benefits of DevOps, and its role in enhancing collaboration and efficiency between development and operations teams.

CO 2	**Acquiring** hands-on experience with popular DevOps tools such as Git, Jenkins, Docker, Kubernetes, and Ansible for implementing continuous integration, continuous delivery, and automated deployment processes.
CO 3	**Demonstrating** proficiency in containerization and orchestration techniques using Docker and Kubernetes for efficient and scalable application deployment and management.
CO 4	**Implementing** configuration management and Infrastructure as Code (IaC) using Ansible and Terraform to automate the provisioning and management of infrastructure resources.
CO 5	**Developing** skills in monitoring, logging, and security practices in the context of DevOps, ensuring application performance, resilience, and adherence to security best practices.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: Introduction to DevOps**	**No. of hours: 12**
Content Summary: **DevOps Principles and Culture**: Understand the core principles of DevOps and its cultural impact. Collaboration, automation, continuous integration, continuous delivery, and continuous deployment. **DevOps Toolchain**: Overview of tools and technologies used in DevOps practices. Introduction to popular DevOps tools like Git, Jenkins, Docker, Kubernetes, and Ansible. **Version Control with Git**: Branching, merging, and collaborative development		

using Git. **Continuous Integration (CI):** Setting up CI pipelines with Jenkins for automated building and testing. **Continuous Delivery and Deployment:** Implementing CD pipelines for deploying.		
Unit Number: 2	**Title: Version Control and CI/CD**	**No. of hours: 8**
Content Summary: **Version Control with Git**: Version control concepts, Git workflows, and collaboration strategies. **Continuous Integration with Jenkins**: Setting up Jenkins pipelines, automated testing, and deployment. **Maven Integration**: Integrate Maven for dependency management and building projects.		
Unit Number: 3	**Title: Containerization and Orchestration**	**No. of hours: 8**
Content Summary: **Introduction to Docker:** Docker concepts, container management, and Docker file creation. **Container Orchestration with Kubernetes:** Kubernetes architecture, deployment, scaling, and networking. **Docker Compose:** Managing multi-container applications with Docker Compose.		
Unit Number: 4	**Title: Configuration Management and Monitoring**	**No. of hours: 12**
Content Summary: **Configuration Management with Ansible**: Ansible playbooks, roles, and		

infrastructure automation.

Infrastructure as Code (IaC): Terraform for provisioning and managing infrastructure.

Monitoring and Logging: Monitoring tools, log management, and application performance monitoring in DevOps.

Security in DevOps: Implementing security best practices in CI/CD pipelines and containerized environments.

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures**: Explore DevOps concepts and automation techniques through engaging presentations.
2. **Problem-Based Assignments**: Analyze real-world scenarios to enhance DevOps and automation skills.
3. **Project Labs**: Implement automation tools and practices in hands-on lab sessions.
4. **Collaborative Work**: Work in teams on case studies related to DevOps implementation.
5. **Continuous Feedback**: Receive regular assessments and instructor feedback to monitor progress.

Outside Classroom Learning Experience

1. **Moodle Access**: Access course materials anytime via Moodle for flexible learning.
2. **Question Bank**: Utilize a question bank and model papers for effective exam preparation.
3. **Real-World Applications**: Apply DevOps and automation concepts to practical scenarios.

References

- Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation, Authors: Jez Humble and David Farley, Publisher: Pearson Education, Inc., Year: 2011
- The Kubernetes Book, Author: Nigel Poulton, Publisher: Independently published, Year: 2018
- Building Microservices: Designing Fine-Grained Systems, Author: Sam Newman, Publisher: O'Reilly Media, Inc., Year: 2015
- Microservices Patterns: With examples in Java, Author: Eberhard Wolff, Publisher: Manning Publications, Year: 2018
- Terraform: Up & Running: Writing Infrastructure as Code, Author: Yevgeniy Brikman, Publisher: O'Reilly Media, Inc., Year: 2017

Additional Readings:

Online Learning Resources for DevOps & Automation

I) **Kubernetes Academy by VMware**

a. **Description:** Free courses provided by VMware on Kubernetes, covering everything from basic concepts to advanced orchestration techniques.

b. **Link:** Kubernetes Academy by VMware

II) **HashiCorp Learn: Terraform**

a. **Description:** HashiCorp's official resource for learning Terraform, providing tutorials and hands-on labs for infrastructure as code.

b. **Link:** HashiCorp Learn - Terraform

III) **Docker: Docker for Developers**

a. **Description:** Docker's official training resources for developers, covering containerization, Docker Compose, and more.

b. **Link:** Docker - Docker for Developers

DEVOPS & AUTOMATION LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: DevOps & Automation Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP461	0-0-2	1
Type of Course:	DSE-3		
Pre-requisite(s), if any:			

Lab Experiments

Defined Course Outcomes

COs	**Course Outcomes**
CO1	**Implementing** collaborative development and continuous integration using Git and Jenkins, demonstrating proficiency in version control, automated testing, and deployment processes.
CO2	**Developing** and deploy microservices applications using Docker for containerization and Kubernetes for orchestration, managing multi-container applications efficiently.
CO	**Managing** automated infrastructure provisioning and configuration using Ansible and Terraform, demonstrating expertise in

3	infrastructure as code and configuration management.
CO4	**Implementing** continuous monitoring, logging, and security best practices in a DevOps environment, ensuring application performance, system health, and data integrity.

Experiment	**Mapped CO(s)**
Set up a Git repository and practice branching, merging, and collaborative development. Create a small project and manage code versions using Git.	CO1
Install and configure Jenkins for continuous integration. Create a simple CI pipeline that automatically builds and tests a project whenever code changes are committed to the repository.	CO1
Implement a continuous delivery pipeline using Jenkins. Deploy a sample application to a staging environment automatically after successful builds and tests.	CO1
Implement different Git workflows (e.g., GitFlow, Feature Branch Workflow) for a collaborative project. Manage branches, merges, and resolve conflicts.	CO2
Set up a Jenkins pipeline for continuous integration.	C

Configure automated testing and deployment for a sample project. Integrate with a version control system like Git.	O2
nstall Docker and create Dockerfiles for a sample application. Build, run, and manage containers using Docker commands.	C O3
Use Docker Compose to manage multi-container applications. Create a Docker Compose file to run a web application with a database and other services.	C O3
Use Terraform to provision and manage cloud infrastructure. Create Terraform scripts to deploy a web application on a cloud provider (e.g., AWS, Azure).	C O4
Set up monitoring and logging for a sample application. Use tools like Prometheus, Grafana, and ELK Stack (Elasticsearch, Logstash, Kibana) to monitor and analyze application performance and logs.	C O4

.NET FRAMEWORK

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: .NET Framework	**Course Code**	**L-T-P**	**Credits**
	ENSP413	4-0-0	4
Type of Course:	DSE-3		
Pre-requisite(s), if any:			

Course Perspective. The ".NET Framework" syllabus covers introduction and components of .NET, programming languages, Visual Studio, OOP, exception handling, memory management, Windows Forms/WPF, ASP.NET, web services,

.NET Core, Entity Framework, and WCF. Emphasis on practical application and development skills for building robust and secure applications.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** .NET Framework's architecture, CLR, and CTS for cross-language integration and platform independence.
CO 2	**Applying** OOP concepts in .NET for designing robust software solutions.
CO 3	**Utilizing** Visual Studio debugging for diagnosing and fixing errors in .NET applications.
CO 4	**Demonstrating** proficiency in memory management and garbage collection in .NET.
CO 5	**Designing** web applications using ASP.NET, incorporating best practices.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number:1	**Title: .NETFramework**	**No. of hours: 8**
Content Summary: NET Framework - Architecture, Common Language Runtime, Common Type System, Namespaces, Assemblies, Memory Management, Process		

Management, Class Libraries		
Unit Number:2	**Title:.NET Framework Fundamentals**	**No. of hours: 8**
Content Summary: Object-Oriented Programming (OOP) in .NET, Classes, objects, and inheritance, Exception Handling and Debugging, Debugging techniques and tools in Visual Studio, Logging and error reporting in .NET applications, Memory Management and Garbage Collection, Automatic memory management in .NET, Garbage collection, Finalizers and the Dispose pattern		
Unit Number:3	**Title: Building Applications with .NET Framework**	**No. of hours: 12**
Content Summary: .NET - Declaration, Expression, Control Structures, Function, String, Array, Encapsulation, Class, Property, Indexer, Delegate, Inheritance, Interface, Polymorphism, Exception Handling, Modules, Graphics, File handling and Data Access. .NET – Form- Event–Form Controls – Containers – Menus - Data controls - Printing – Reporting – Dialogs – Components - Single and Multiple Document Interfaces.		
Unit Number:4	**Title: ASP.NETFramework**	**No. of hours: 12**
Content Summary: ASP.NET – Web Pages, Web Forms, Web Site Design, Data Controls, Validation Controls, HTML, Navigation Controls, Login Controls, Reports - Master Pages – Web Service Architecture - Basic Web Services – Web Reference – Standards		

Learning Experiences:

1. **Interactive Lectures**: Explore .NET Framework concepts through engaging presentations.

2. **Problem-Based Assignments**: Analyze real-world scenarios to enhance .NET development skills.
3. **Project Labs**: Build and test applications using the .NET Framework in hands-on labs.
4. **Collaborative Work**: Work in teams on case studies related to .NET application development.
5. **Continuous Feedback**: Receive regular assessments and instructor feedback to track progress.

Outside Classroom Learning Experience

1. **Moodle Access**: Access course materials anytime via Moodle for convenient learning.
2. **Question Bank**: Utilize a question bank and model papers for effective exam preparation.
3. **Real-World Applications**: Apply .NET concepts to practical application scenarios.

Textbooks

1. Pro C# 8 with .NET Core: Foundational Principles and Practices in Programming by Andrew Troelsen and Philip Japikse, Apress, 9th Edition, 2020
2. Pro ASP.NET Core 3 by Adam Freeman, Apress
3. ASP.NET Core in Action by Andrew Lock

Additional Readings:

Online Learning Resources:

I) Online Tutorials and Documentation: Direct students to the official Microsoft documentation for .NET Framework, which provides comprehensive guides and resources. Microsoft .NET Documentation

II) Hands-on Coding Exercises: Assign coding exercises from platforms like LeetCode or HackerRank that focus on implementing concepts of .NET Framework. LeetCode HackerRank

.NET FRAMEWORK LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
.NET Framework Lab	**ENSP463**	0-0-2	1
Type of Course:	DSE-3		
Pre-requisite(s), if any: Nil			

Defined Course Outcomes

COs	Statements
CO1	**Understanding** and apply object-oriented design principles, exception handling, memory management, and debugging techniques to develop robust .NET applications.
CO2	**Developing** graphical user interfaces and handle events in .NET applications to create interactive and user-friendly software solutions.
CO3	**Implementing** web development techniques in ASP.NET, including web forms, user authentication, master pages, and web services to build secure and dynamic web applications.
CO4	**Analyzing** and utilize data handling, reporting, and visualization techniques to create comprehensive and functional software systems for various domains.

Lab Experiments

Experiment	Mapped

	CO/COs
Explore the architecture of the .NET Framework. Create a simple console application to understand the basic structure and components of a .NET project.	CO1
Demonstrate the functionality of the Common Language Runtime (CLR). Create a .NET application that uses various data types and namespaces to show how the CLR manages execution.	CO1
Implement a .NET application that showcases the Common Type System (CTS). Define and use various data types, and demonstrate type conversion and interoperability.	CO1
Create and manage assemblies in a .NET application. Demonstrate how to build, reference, and use assemblies in a multi-project solution.	CO1
Implement a simple object-oriented application in .NET. Define classes, create objects, and demonstrate inheritance and polymorphism.	CO2
Implement a .NET application that logs errors and handles exceptions gracefully. Use a logging framework (e.g., NLog, log4net) to record application events and errors.	CO2
Build a .NET application demonstrating advanced OOP concepts such as encapsulation, properties, indexers, delegates, interfaces, and polymorphism.	CO3
Create a .NET application that handles graphics and file I/O. Implement functionality to draw shapes, handle	CO

images, and perform file read/write operations.	3
Implement a .NET application with a rich user interface. Use forms, event handling, form controls, containers, menus, data controls, printing, and reporting functionalities to create a feature-rich application.	CO3
Create a basic ASP.NET web application. Design web pages and web forms to understand the structure and components of an ASP.NET project.	CO4
Develop an ASP.NET application with user authentication. Use login controls to implement user authentication and authorization, and create a simple reporting feature to display user data.	CO4
Create and consume a basic web service in ASP.NET. Implement a web service that provides data to a client application, and demonstrate how to use web references to integrate the web service with an ASP.NET project.	CO4

NEW AGE PROGRAMMING LANGUAGES

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
New-Age programming languages	**ENSP415**	4-0-0	4
Type of Course:	DSE-3		
Pre-requisite(s), if any: Nil			

Course Perspective. New-Age programming languages (GO, F#, Clojure, Kotlin) provides an introduction to the concepts and applications of modern programming languages. It explore the features and benefits of GO, F#, Clojure, and Kotlin, and develop practical skills in programming using these languages. The course will cover language syntax, data types, control structures, functional programming concepts, concurrency, and integration with other technologies.

The Course Outcomes (COs). On completion of the course the participants will be:

COs	Statements
CO 1	**Understanding** principles and paradigms of modern programming languages.
CO 2	**Developing** proficiency in syntax, data structures, and control flow of each

	language.
CO 3	**Exploring** unique features and strengths of each language.
CO 4	**Applying** development tools to improve code quality and productivity.
CO 5	**Designing** and implement projects integrating multiple programming languages.

CO = Course outcomes. A student is expected to have learnt concepts and demonstrated/developed abilities or skills related to strategic management at the end of the course.

Course Outline:

Unit Number: 1	**Title: GO programming Language**	**No. of hours: 10**

Content Summary:

Overview and Comparison: Overview of GO, F#, Clojure, and Kotlin, Comparison with traditional programming languages, Installation and setup of development environment,

GO Programming Basics: Introduction to GO syntax and data types, Control structures in GO, Functions and packages, Arrays, slices, and maps, Structs and custom data types,Pointers and memory management

Unit Number: 2	**Title: F# Programming Language**	**No. of hours: 10**

Content Summary:

Introduction to F# syntax and functional programming concepts, Data Types, Variables, Operators, Decision Making, Loops, Functions, Strings, Options, Immutable data types and pattern matching, Higher-order functions and currying, Asynchronous and parallel programming in F#, Object-Oriented Programming with F#, Database access with F#, Querying and manipulating data using F#, Integration with relational and NoSQL databases

Unit Number: 3	**Title: Introduction to Clojure Programming**	**No. of hours: 10**

Content Summary:

Introduction to Clojure: Overview of Clojure and its features, Setting up the development environment,

Basic Syntax and Functional Programming, Basic syntax and data structures, Functional programming concepts, Immutable data and pure functions, Higher-order functions and recursion, Collections and sequence operations, Restructuring and pattern matching

Error Handling and Testing: Exception handling and error management in Clojure, Testing strategies and frameworks in Clojure,

Data Manipulation and Transformation: Data manipulation with Clojure's sequence functions, Data transformation with transducers, Data-driven development with data literals and data readers

Unit Number: 4	**Title: Introduction to Kotlin Programming**	**No. of hours: 10**

Content Summary:

Overview of Kotlin and its advantages, Setting up the development environment, Basic syntax and data types in Kotlin, Conditional statements and loops, Function declarations and parameters, Lambda expressions and higher-order functions,

Object-Oriented Programming in Kotlin: Classes, objects, and inheritance, Properties and access modifiers, Interfaces and abstract classes, Understanding nullable and non-nullable types, Safe calls and the Elvis operator, Type inference and smart casting,

Collections and Functional Programming: Working with lists, sets, and maps in Kotlin, Collection operations and transformations, Introduction to functional programming concepts in Kotlin, Creating extension functions in Kotlin, Using DSLs for domain-specific problems, Builder pattern and DSL implementation.

Learning Experiences:

Classroom Learning Experience

1. **Interactive Lectures**: Explore new programming languages through engaging presentations.
2. **Problem-Based Assignments**: Analyze real-world scenarios to enhance programming skills.
3. **Project Labs**: Build and test applications using various new programming languages in hands-on labs.
4. **Collaborative Work**: Collaborate on case studies related to modern programming practices.
5. **Continuous Feedback**: Receive regular assessments and instructor feedback to monitor progress.

Outside Classroom Learning Experience

1. **Moodle Access**: Access course materials anytime via Moodle for flexible learning.
2. **Question Bank**: Utilize a question bank and model papers for effective exam preparation.
3. **Real-World Applications**: Apply concepts from new programming languages to practical scenarios.

Text Books:

1. The Go Programming Language, Alan A. A. Donovan and Brian W. Kernighan, Addison-Wesley Professional.
2. An Introduction to Programming in Go, Caleb Doxsey, CreateSpace Independent Publishing.

References

3. Real-World Functional Programming: With Examples in F# and C#, Tomas Petricek and Jon Skeet, Manning.
4. Programming F# 3.0: A Comprehensive Guide for Writing Simple Code to Solve Complex Problems, Chris Smith, O'Reilly Media.
5. Getting Clojure: Build Your Functional Skills One Idea at a Time, Russ Olsen, O'Reilly.
6. The Joy of Clojure, Michael Fogus and Chris Houser, Manning Publication.
7. Atomic Kotlin, Bruce Eckel and Svetlana Isakova, Mindview LLC.
8. Kotlin in Action, Dmitry Jemerov and Svetlana Isakova, Manning Publication.

Additional Readings:

Online Learning Resources for New-Age Programming Languages

a) **Go (Golang)**
    1. **Coursera: Programming with Google Go**
        1. **Description:** An introductory course to Go programming, covering language syntax, data structures, and more.
        2. **Link:** Coursera - Programming with Google Go
    2. **Go by Example**
        1. **Description:** A hands-on introduction to Go using annotated example programs.
        2. **Link:** Go by Example

b) **F#**
    1. **Microsoft Learn: Introduction to F#**
        1. **Description:** A series of modules introducing the F# language, its syntax, and functional programming concepts.
        2. **Link:** Microsoft Learn - Introduction to F#

c) **Clojure**
    1. **ClojureBridge**

    1. **Description:** Free Clojure workshops for beginners, including resources and exercises.
    2. **Link:** [ClojureBridge]
2. **Learn Clojure: Clojure for the Brave and True**
    1. **Description:** A beginner-friendly book that teaches Clojure through real-world projects and examples.
    2. **Link:** [Clojure for the Brave and True]

d) **Kotlin**

1. **Kotlin Lang: Kotlin Documentation**
    1. **Description:** Official Kotlin documentation and tutorials by JetBrains.
    2. **Link:** Kotlin Documentation
2. **Udacity: Kotlin for Android Developers**
    1. **Description:** A course by Udacity focusing on Kotlin for Android development.
    2. **Link:** [Udacity - Kotlin for Android Developers]

NEW AGE PROGRAMMING LANGUAGES LAB

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: New Age Programming languages Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP465	0-0-2	1
Type of Course:	DSE-3		
Pre-requisite(s), if any: Nil			

Course Outcomes (CO)

COs	Statements
CO1	**Understanding** the fundamental principles and paradigms of modern programming languages.
CO2	**Developing** proficiency in using the syntax, data structures, and control flow constructs of each language.
CO3	**Exploring** the unique features and strengths of each language, such as Go's focus on concurrency, F#'s functional programming capabilities, Clojure's emphasis on immutability and simplicity, and Kotlin's interoperability with existing Java code.

CO4	**Applying** the languages' respective development tools and best practices.
CO5	**Implementing** projects that utilize the strengths of each language to tackle complex problems or tasks.

Lab Experiments

Experiment Title	C C
Develop a RESTful API for a simple blog application in Go. The API should allow users to create, read, update, and delete blog posts. Use Go's built-in net/http package and struct types for handling blog post data	C C 1
Create a command-line tool in Go that fetches and displays current weather information for a specified city. Use a public weather API and Go's JSON parsing capabilities to implement this tool.	C C 1
Set up the F# development environment. Create a simple F# program to demonstrate basic syntax, data types, and variables.	C C 1
Develop a functional calculator application in F#. The calculator should support basic arithmetic operations, as well as more advanced functions like trigonometry and logarithms. Use pattern matching and immutable data structures to handle calculations.	C C 2
Create a small web application in F# using Suave (a lightweight web server library). The application should allow users to register, log in, and create simple posts. Implement basic session management and data storage.	C C 2
Build a financial portfolio tracker in F#. The application should	C

allow users to input and track their investments, calculate current value, and generate reports. Use F#'s asynchronous programming capabilities to fetch real-time stock prices from a financial API.	C 2
Develop a to-do list application in Clojure. The application should allow users to add, remove, and mark tasks as complete. Use Clojure's sequence operations and immutable data structures to manage tasks.	C C 3
Create a simple web scraper in Clojure. The scraper should fetch data from a specified website, parse the HTML content, and extract specific information. Use Clojure's libraries for HTTP requests and HTML parsing.	C C 3
Develop a Kotlin-based Android application for tracking fitness activities. The app should allow users to log their workouts, view statistics, and set goals. Use Kotlin's object-oriented features and Android SDK for development.	C C 4
Create a Kotlin DSL (Domain-Specific Language) for generating HTML pages. The DSL should allow users to define HTML structures using Kotlin syntax and generate the corresponding HTML code.	C C 4

Summer Internship-III

Program Name:	**Bachelor of Technology (CSE) with specialization in AI & ML**		
Course Name: Summer Internship-III	**Course Code**	**L-T-P**	**Credits**
	ENSI451	---	2
Type of Course:	INT-3		
Pre-requisite(s), if any: NA			

Course Outcomes (CO)

CO1	Applying theoretical knowledge from core subjects to real-world problems in an industry or academic setting.
CO2	Demonstrating the acquisition of new technical skills relevant to the field of computer science and engineering during the internship.
CO3	Developing a comprehensive case study, project, or research paper that reflects the practical application of internship experiences.
CO4	Presenting internship outcomes effectively, showcasing professional growth, technical competencies, and communication skills.

Duration:

The internship will last for six weeks. It will take place after the completion of the 6^{th} semester and before the commencement of the 7^{th} semester.

Internship Options:

Students can choose from the following options:

- **Industry Internship (Offline) or Internship in Renowned Academic Institutions (Offline):**
    - Students must produce a joining letter at the start and a relieving letter upon completion.

Report Submission and Evaluation:

1. **Report Preparation:**
    - Students must prepare a detailed report documenting their internship experience and submit it to the department. A copy of the report will be kept for departmental records.
2. **Case Study/Project/Research Paper:**
    - Each student must complete one of the following as part of their internship outcome:
        1. A case study
        2. A project
        3. A research paper suitable for publication
3. **Presentation:**
    - Students are required to present their learning outcomes and results from their summer internship as part of the evaluation process.

Evaluation Criteria for Summer Internship (Out of 100 Marks):

valuation Criteria	**Maximum Marks**
Relevance to Learning Outcomes	**30 Marks**
- Case Study/Project/Research Paper Relevance	15 Marks
- Application of Theoretical Knowledge	15 Marks
Skill Acquisition	**40 Marks**

valuation Criteria	**Maximum Marks**
- New Technical Skills Acquired	20 Marks
- Professional and Soft Skills Development	20 Marks
Report Quality	**15 Marks**
- Structure and Organization	8 Marks
- Clarity and Comprehensiveness	7 Marks
Presentation	**15 Marks**
- Content Delivery	8 Marks
- Visual Aids and Communication Skills	7 Marks

| Total | 100 Marks |

Detailed View:

1. **Relevance to Learning Outcomes (30 Marks)**
   - **Case Study/Project/Research Paper Relevance (15 Marks):**
     1. Directly relates to core subjects: 15 marks
     2. Partially relates to core subjects: 10 marks
     3. Minimally relates to core subjects: 5 marks
     4. Not relevant: 0 marks
   - **Application of Theoretical Knowledge (15 Marks):**
     1. Extensive application of theoretical knowledge: 15 marks
     2. Moderate application of theoretical knowledge: 10 marks
     3. Minimal application of theoretical knowledge: 5 marks
     4. No application of theoretical knowledge: 0 marks
2. **Skill Acquisition (40 Marks)**
   - **New Technical Skills Acquired (20 Marks):**
     1. Highly relevant and advanced technical skills: 20 marks
     2. Moderately relevant technical skills: 15 marks
     3. Basic technical skills: 10 marks
     4. No new skills acquired: 0 marks

- **Professional and Soft Skills Development (20 Marks):**
    1. Significant improvement in professional and soft skills: 20 marks
    2. Moderate improvement in professional and soft skills: 15 marks
    3. Basic improvement in professional and soft skills: 10 marks
    4. No improvement: 0 marks

3. **Report Quality (15 Marks)**
    - **Structure and Organization (8 Marks):**
        1. Well-structured and organized report: 8 marks
        2. Moderately structured report: 6 marks
        3. Poorly structured report: 3 marks
        4. No structure: 0 marks
    - **Clarity and Comprehensiveness (7 Marks):**
        1. Clear and comprehensive report: 7 marks
        2. Moderately clear and comprehensive report: 5 marks
        3. Vague and incomplete report: 2 marks
        4. Incomprehensible report: 0 marks
4. **Presentation (15 Marks)**
    - **Content Delivery (8 Marks):**
        1. Clear, engaging, and thorough delivery: 8 marks
        2. Clear but less engaging delivery: 6 marks
        3. Somewhat clear and engaging delivery: 3 marks
        4. Unclear and disengaging delivery: 0 marks
    - **Visual Aids and Communication Skills (7 Marks):**
        1. Effective use of visual aids and excellent communication skills: 7 marks
        2. Moderate use of visual aids and good communication skills: 5 marks
        3. Basic use of visual aids and fair communication skills: 2 marks
        4. No use of visual aids and poor communication skills: 0 marks

Total: 100 Marks

Semester: VIII

Industrial Project/R&D Project/Start-up Project

<table>
<tr><td>Program</td><td colspan="3">Bachelor of Technology (CSE) with specialization in AI & ML</td></tr>
<tr><td rowspan="2">Course Name:
Industrial Project/R&D Project/Start-up Project</td><td>Course Code</td><td>L-T-P</td><td>Credits</td></tr>
<tr><td>ENSI452</td><td></td><td>12</td></tr>
<tr><td>Type of Course:</td><td colspan="3">PROJ-4</td></tr>
<tr><td colspan="4">Pre-requisite(s), if any:</td></tr>
<tr><td colspan="4">Preface:
The B.Tech Final Semester Full-Time Project Work is a culmination of the academic journey for engineering students at the School of Engineering & Technology, K.R. Mangalam University. This detailed Standard Operating Procedure (SOP) is designed to guide students through their project, ensuring a comprehensive, practical, and outcome-driven approach.
The SOP provides a framework for students to choose from three types of projects—Industrial Projects, Research & Development (R&D) Projects, and Start-up Projects. It emphasizes experiential learning, real-world problem-solving, and interdisciplinary collaboration, reflecting NEP 2020’s focus on holistic development, innovation, and entrepreneurship. Students will work under the mentorship of both internal faculty and external experts, ensuring they are equipped with the skills and knowledge required to excel in industry, research, or entrepreneurship.
This document outlines each stage of the project work, from proposal submission to final evaluation, and offers clear guidelines for successful completion. By adhering to this SOP, students will not only demonstrate their technical proficiency but also contribute meaningfully to industry, academia, and society.</td></tr>
</table>

Standard Operating Procedure (SOP) for B.Tech Final Semester Full-Time Project Work

1. Introduction

The **B.Tech Final Semester Full-Time Project Work** is an essential academic requirement aimed at providing students with the opportunity to apply theoretical knowledge to practical challenges. The project is designed to foster critical thinking, problem-solving, innovation, and research-oriented learning, with a focus on real-world industrial, research, and entrepreneurial domains. Students may choose from:

- **Industrial Project**: Solving real industrial problems in collaboration with an industry partner.
- **Research & Development (R&D) Project**: Contributing to academic and applied research, with external guidance from academic/research institutions.
- **Start-up Project**: Developing and launching innovative start-up ideas with entrepreneurial mentors.

The SOP ensures that the project aligns with **NEP 2020 guidelines**, emphasizing interdisciplinary, practical, and outcome-based learning.

---

2. Objectives

The primary objectives of the full-time project are:

- **Application of Theoretical Knowledge**: Enabling students to apply their academic learning to practical problems.
- **Holistic Development**: Promoting interdisciplinary learning, critical thinking, creativity, and problem-solving.

- **Research and Innovation**: Encouraging innovative solutions, leading to publications, patents, or prototypes.
- **Industry Collaboration**: Fostering partnerships with industries for real-world problem-solving.
- **Entrepreneurship Development**: Developing entrepreneurial skills and creating viable start-ups.
- **Global Competency**: Ensuring students develop the skills required to excel in global environments through research, innovation, and collaboration.

---

3. Types of Projects

a) Industrial Project

Students working on **Industrial Projects** will:

- Collaborate with an industry partner.
- Identify specific, real-world challenges faced by the company.
- Propose and implement a solution that provides value to the industry.
- Develop a final product or prototype that can be implemented in the industrial setting.

Project Proposal:

- Problem Statement and Objectives: Identify the industrial problem and outline the objectives.
- Proposed Solution: Present a detailed methodology for solving the problem.
- Deliverables: Define tangible deliverables, including prototypes, software, or hardware.
- Expected Impact: Outline the expected impact on the industry.

Evaluation Criteria:

- Practical implementation and solution viability (40%)
- Project innovation (20%)

- Industrial applicability and impact (20%)
- Final presentation and report quality (20%)

b) Research & Development (R&D) Project

The **R&D Project** focuses on creating innovative research outcomes through collaborations with academic or research institutions. This can result in publications, research reports, or new discoveries.

Project Proposal:

- Literature Review: Detailed research on existing work related to the chosen topic.
- Hypothesis/Research Questions: Define the specific research problem or question.
- Methodology: Include data collection, experimental design, and analysis techniques.
- Research Timeline: Step-by-step phases of research with milestones.

External Mentor: Collaboration with an **external academic expert** is mandatory for research projects. The external mentor must be a research professional with expertise in the specific field of study.

Internal Mentor: Each student will also be assigned an **internal faculty member** who will supervise the project. The internal mentor will ensure that the research meets academic standards and deadlines.

Evaluation Criteria:

- Quality of Research and Novelty (30%)
- Research Methodology (25%)
- Contributions to the field (20%)
- Final Report, Presentation, and Publication (25%)

c) Start-up Project

The **Start-up Project** involves developing a business model or creating a start-up venture. Students work on a product/service idea that addresses a significant market need or societal problem.

Project Proposal:

- Start-up Idea: Explain the business or product idea.
- Market Research: Detailed research on the market, target customers, competitors, and potential revenue streams.
- Business Plan: Define the steps needed to take the idea to market, including funding, development phases, marketing, and operational plans.
- Product Prototype: If applicable, develop a working prototype.

Mentorship:

- **External Mentor**: An industry/start-up expert will guide the student in refining the idea, business model, and market strategy.
- **Internal Faculty Mentor**: An internal mentor will provide academic guidance and ensure the start-up idea is feasible and innovative.

Evaluation Criteria:

- Start-up viability and market potential (30%)
- Product or service innovation (30%)
- Prototype/Business Model Development (20%)
- Final Pitch/Presentation and Start-up Plan (20%)

---

4. Roles and Responsibilities

a) Student's Responsibilities:

- Select a suitable project topic based on interests (industrial, R&D, or start-up).
- Draft and submit a detailed proposal with objectives, methodology, timelines, and deliverables.
- Coordinate with both external and internal mentors regularly for feedback and guidance.
- Maintain a weekly progress report for both mentors.

- Submit a final comprehensive report and present the project.

b) Internal Supervisor:

- Guide the student throughout the project.
- Provide academic input and ensure that the project aligns with the program outcomes.
- Conduct progress reviews and ensure timelines are adhered to.
- Evaluate the project at the mid-term and final stages.

c) External Mentor:

- Offer specialized industrial, research, or entrepreneurial guidance.
- Provide real-world problem insights for industrial and start-up projects.
- Ensure the project is relevant to the chosen industry, research domain, or start-up ecosystem.
- Participate in the final evaluation of the project.

---

5. Project Phases

Phase 1: Proposal Submission and Approval

- Students will submit a project proposal during the first two weeks of the final semester.
- The proposal must include the problem statement, objectives, literature review (for R&D projects), methodology, and expected outcomes.
- The proposal is subject to review and approval by the internal supervisor and external mentor.

Phase 2: Planning and Resource Allocation

- Once approved, the student will develop a project plan that includes:
    - **Project Milestones**: Break down the project into smaller tasks with defined milestones.

- **Resource Requirements**: Identify any software, hardware, lab resources, or tools required for the project.
- **Team Roles**: For group projects, define the roles of each team member.
- **Risk Assessment**: Highlight potential risks and the corresponding mitigation strategies.

Phase 3: Mid-term Review

- A mid-term review will be conducted halfway through the project to assess progress.
- Students will present their work to a committee consisting of the internal supervisor, external mentor, and department head.
- The review will assess the progress against the timeline and suggest course corrections if needed.

Phase 4: Final Execution and Evaluation

- **Industrial Projects**: Students must submit a prototype or industrial report, demonstrating the solution's applicability to the industry.
- **R&D Projects**: Students must submit a final research report or publish findings in academic journals.
- **Start-up Projects**: Students must present a business plan, along with a working prototype, market analysis, and revenue model.

Phase 5: Final Report Submission and Presentation

- **Final Report**: The project report should contain a title page, abstract, introduction, problem statement, objectives, methodology, results, discussion, conclusions, future scope, references, and appendices.
- **Presentation**: Students will deliver a final presentation to a panel of evaluators, showcasing their work, findings, or product.
- **Evaluation**: Based on the final report and presentation, students will be awarded marks in accordance with the evaluation rubrics.

6. Collaboration and Mentorship

For **Research Projects**, the mentorship will involve both:

- **External Mentor**: An academic expert outside the institution, preferably from a reputed university or research institute.
- **Internal Mentor**: A faculty member from the student's department to provide academic and administrative guidance.

For **Industrial Projects**:

- External mentorship will come from industry professionals, preferably from the partnering company.

For **Start-up Projects**:

- External mentorship will involve experienced entrepreneurs, start-up founders, or investors.

Mentors will:

- Provide critical inputs on the technical, business, or research aspects of the project.
- Offer feedback and advice during each phase of the project.

---

7. NEP 2020 Guidelines

The project structure is designed to ensure interdisciplinary learning and foster entrepreneurial and research innovation, in line with the **NEP 2020** guidelines:

- **Interdisciplinary Approach**: Students are encouraged to explore projects that bridge different fields of study.
- **Flexibility**: Students have the flexibility to choose between industrial, research, or start-up projects.
- **Experiential Learning**: Real-world problem-solving and hands-on project work are at the core of this initiative.
- **Collaboration**: The integration of external mentors ensures industry and academic collaboration.

8. Documentation and Submission Requirements

Students are required to:

- Submit their proposal, mid-term report, final report, and any supporting documents via the **Learning Management System (LMS)**.
- Maintain detailed project logs and weekly reports.