K.R. MANGALAM UNIVERSITY
THE COMPLETE WORLD OF EDUCATION

SCHOOL OF ENGINEERING AND TECHNOLOGY

Programme Handbook

(Programme Study and Evaluation Scheme)

B. Tech (Computer Science & Engineering)

with Specialization in UI/UX

Programme Code: 36

FOUR YEAR UNDERGRADUATE PROGRAMME

(with effect from 2024-25 session)

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

Contents

Preface

Welcome to the School of Engineering and Technology at K. R. Mangalam University. It is with great enthusiasm that we introduce you to an institution dedicated to nurturing future leaders in engineering and technology.

Established in 2013, our School has rapidly evolved into a premier center for innovation, quality education, and skill development. With a focus on imparting advanced knowledge and fostering creativity, we are committed to providing a transformative educational experience. Our state-of-the-art infrastructure, cutting-edge laboratories, and a distinguished team of faculty members collectively create an environment where academic and professional excellence thrives.

Our diverse programs encompass undergraduate degrees (B.Tech, BCA, B.Sc), postgraduate studies (M.Tech, MCA), and doctoral research across all engineering disciplines. Notably, we offer specialized B.Tech programs in areas such as Artificial Intelligence & Machine Learning, Data Science, Cyber Security, Full Stack Development, and UI/UX Development. These programs are designed to equip students with both technical proficiency and a deep understanding of emerging technologies.

At the heart of our mission is a commitment to a curriculum that integrates the best practices from leading global institutions while also incorporating insights from the Open-Source Society University. This curriculum emphasizes problem-solving, interdisciplinary learning, and innovative teaching methodologies, all aligned with the National Education Policy (NEP) 2020.

Our emphasis on industry integration is reflected in our collaborations with renowned organizations such as IBM, Samatrix, Xebia, E.C Council, and ImaginXP. These partnerships ensure that our students gain practical experience and insights that are directly applicable to industry demands. Elective options across diverse domains, including AI, Cloud Computing, Cyber Security, and Full Stack Development, offer students the flexibility to tailor their educational experience to their career aspirations.

We are also dedicated to fostering a culture of innovation and entrepreneurship through our Entrepreneurship and Incubation Center and initiatives like 'MindBenders,' 'Hack-KRMU,' and

participation in the 'Smart India Hackathon.' These programs are designed to inspire and prepare students to become forward-thinking leaders in the technology sector.

Our modern computing facilities and comprehensive infrastructure support advanced research, simulations, and hands-on projects, ensuring that our students are well-prepared for the challenges of the professional world. K. R. Mangalam University is recognized for its commitment to providing quality education, and our alumni have made notable contributions across various sectors, from multinational corporations to public sector enterprises.

We are excited to accompany you on this journey and look forward to supporting your academic and professional growth. Welcome to a community where excellence and innovation are at the core of everything we do.

School of Engineering & Technology
K.R Mangalam University

Categories of Courses

Major: The major would provide the opportunity for a student to pursue in-depth study of a particular subject or discipline.
Minor: Students will have the option to choose courses from disciplinary/interdisciplinary minors and skill-based courses. Students who take a sufficient number of courses in a discipline or an interdisciplinary area of study other than the chosen major will qualify for a minor in that discipline or in the chosen interdisciplinary area of study.

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 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): The Skill Enhancement Courses (SEC) equip students with practical expertise and hands-on experience in key areas of UX/UI, including technological integration, AI-driven design, and interaction design principles. These courses enhance employability by focusing on real-world applications, portfolio development, and the skills needed to succeed in modern design and technology-driven industries.

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

Research Project / Dissertation: Students choosing a 4-Year Bachelor's degree (Honours with Research) are required to take up research projects under the guidance of a faculty member. The students are expected to complete the Research Project in the eighth semester. The research outcomes of their project work may be published in peer-reviewed journals or may be presented in conferences /seminars or may be patented.

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

The School of Engineering and Technology at K. R. Mangalam University started in 2013 to create a niche of imparting quality education, innovation, entrepreneurship, skill development and creativity. It has excellent infrastructure, state of the art Labs, and a team of qualified and research-oriented faculty members. The school is offering undergraduate programs (B.Tech, BCA, B.Sc), postgraduate programs (M.Tech, MCA) and Ph.D (all disciplines of Engineering). We are offering B.Tech programs in recent areas of specializations like AI & ML, Data Science, Cyber Security, Full stack development, UI/UX development etc.

Our strength lies in our highly qualified, research oriented, and committed teaching faculty. We believe in empowering minds through expert guidance, ensuring that our students receive a world-class education that prepares them for the challenges of the ever-evolving technological landscape.

The School of Engineering & Technology is committed to providing a cutting-edge curriculum by integrating the best practices from top global universities and leveraging the rich knowledge resources of the Open-Source Society University. The curriculum focuses on problem-solving, design, development, interdisciplinary learning, skill development, research opportunities and application of various emerging technologies with a focus on innovative teaching learning methodologies.

We take pride in offering an industry-integrated curriculum that goes beyond traditional education. Collaborations and training led by industry experts, along with partnerships with renowned organizations such as IBM, Samatrix, Xebia, E.C Council, ImaginXP etc ensure that our students gain practical insights and skills that align with real-world industry demands.

With elective options across various domains, including AI, Cloud Computing, Cyber Security, and Full Stack Development, we empower students to customize their learning experience. Our goal is to provide the flexibility needed for each student to shape their academic and professional future.

We prioritize career growth by offering comprehensive training, placements, international internships, and preparation for further studies. Our commitment to nurturing globally competitive professionals is reflected in the diverse pathways we pave for our students.

SOET aims at transforming the students into competitive engineers with adequate analytical skills, making them more acceptable to potential employers in the country. At our school, we emphasize learning through doing. Whether it's project-based learning, field projects, research projects, internships, or engaging in competitive coding, our students actively shape their futures by applying theoretical knowledge to practical

scenarios. We provide opportunities for industrial projects, R&D projects, and start-up projects in the final year, ensuring that our students engage in real-world innovation.

We are dedicated to fostering a culture of innovation and entrepreneurship, recognizing these as essential pillars for the success of our students in the rapidly evolving world of technology. We inspire innovation and entrepreneurship through our dynamic Entrepreneurship and Incubation Center, engaging contests like 'MindBenders' ,'Hack-KRMU,' participation in ‘Smart India Hackathon’, International Conference ‘MRIE’ empowering students to become forward-thinking leaders in the ever-evolving realm of technology.

We pride ourselves on providing state-of-the-art computing facilities and infrastructure. Our modern labs and computing resources are equipped to support the diverse needs of our students, enabling them to engage in advanced research, simulations, and hands-on projects.

K.R. Mangalam University has marked its presence in Delhi NCR as a value-based university, successfully imparting quality education in all domains. Our alumni are working across all sectors of technology, from MNCs to PSUs.

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

The field of Computer Science & Engineering is at the forefront of technological advancements, shaping the world we live in today. It encompasses a diverse range of disciplines, including computer systems, algorithms, software development, networking, artificial intelligence, and more. As technology continues to revolutionize every aspect of our lives, the demand for skilled computer scientists and engineers is ever-increasing. In response to the growing importance of user-centered design in technology, our B. Tech Computer Science & Engineering program now offers a specialization in UX/UI (User Experience/User Interface).

Our B. Tech Computer Science & Engineering with a specialization in UX/UI is designed to provide students with a comprehensive understanding of both the foundational principles of computer science and the essential skills required to excel in the field of user experience and interface design. Over the course of four years, students will delve into core computer science subjects such as programming languages, data structures, operating systems, database management, computer architecture, and software engineering, along with specialized courses in UX/UI.

The UX/UI specialization equips students with the knowledge and practical skills needed to create user-friendly, aesthetically pleasing, and effective interfaces for various digital platforms. Students will explore topics such as: Human-Computer Interaction (HCI), User Research and Usability Testing, Information Architecture, Interaction Design, Design Thinking.

At our institution, we emphasize a hands-on approach to learning, combining theoretical knowledge with practical application. Students will have the opportunity to work on real-world projects, engage in laboratory experiments, and participate in internships to gain valuable industry experience. Our UX/UI specialization includes practical projects where students design and develop user interfaces, conduct usability tests, and iterate on designs based on user feedback.

Furthermore, our curriculum is designed to keep pace with the rapidly evolving nature of the computer science and UX/UI fields. We strive to incorporate the latest trends and emerging technologies, ensuring that our graduates are equipped with the knowledge and adaptability necessary to thrive in a competitive industry. Courses are regularly updated to reflect advancements in UX/UI design tools, methodologies, and best practices.

As technology continues to reshape our world, computer scientists and engineers with expertise in UX/UI have a pivotal role to play in driving innovation and creating solutions to complex challenges. Graduates of our B. Tech Computer Science & Engineering with a specialization in UX/UI will be well-prepared for careers such as: UX/UI Designer, Interaction Designer, Usability Analyst, Front-end Developer, Product Designer and User Researcher

We are committed to providing a supportive and inclusive learning environment, where students can explore their passions, develop their skills, and unlock their full potential. Through dedicated faculty, state-of-the-art infrastructure, and a vibrant community, we strive to create an enriching educational experience

that prepares students for successful careers in the field of Computer Science & Engineering with a specialization in UX/UI.

We invite aspiring students to embark on this exciting journey with us, as together, we explore the limitless possibilities of computer science and engineering and user-centered design. Join us in making a positive impact on the world through innovative and user-friendly technology solutions.

Program Highlights

- Professionally qualified, competent, and committed teaching faculty.
- Industry enabled curriculum and training from industry experts.
- Consistent interaction with renowned academicians and experts.
- Emphasis on project-based learning, techno-pedagogy, field projects, research projects, internships, continuous and comprehensive evaluation.
- Access to certification courses, ability & skill development programs, value-added courses besides core curriculum.
- Effective career counselling, guidance and mentoring program to excel in professional and personal spheres of life.
- Special programs for advanced and slow learners with focus on inclusion and student diversity.
- Focus on career progression through training, placements and preparation for higher studies.
- Centre of excellence in AI, Machine Learning & Data Science, Robotics & Automation.

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 UX & UI design.

PSO2: Applying UX & UI principles to create user-friendly and effective interfaces.

PSO3: Analysing user needs, behaviours, and feedback to inform design decisions.

PSO4: Evaluating and iterating on design solutions to enhance user experience.

PSO5: Designing and developing innovative UX & UI solutions to address complex user interaction challenges.

Career Avenues

Graduates of the B.Tech Computer Science & Engineering program with a specialization in UI/UX have diverse career avenues available to them. Here are some potential career paths:

1. **UI/UX Designer**: UI/UX designers focus on creating user-friendly interfaces and enhancing user experiences for websites, applications, and digital products. They conduct user research, develop wireframes and prototypes, and collaborate with development teams to implement design solutions.

2. **Interaction Designer**: Interaction designers specialize in creating interactive experiences for users. They design the way users interact with a product, ensuring smooth and intuitive user flows. This role involves a deep understanding of user behavior and interaction principles.

3. **Visual Designer**: Visual designers focus on the aesthetics of a digital product. They create visually appealing layouts, select color schemes, typography, and design elements that align with the brand's identity. Their goal is to create an engaging and visually cohesive user experience.

4. **User Researcher**: User researchers gather insights about users' needs, behaviors, and motivations through various research methods such as interviews, surveys, and usability testing. They analyze data to inform design decisions and improve user satisfaction.

5. **Information Architect**: Information architects organize and structure digital content in a way that is easy for users to navigate and find information. They create site maps, user flows, and information hierarchies to ensure a logical and efficient user experience.

6. **Usability Analyst**: Usability analysts evaluate the usability of digital products through user testing and feedback. They identify usability issues, provide recommendations for improvement, and work with design and development teams to enhance user experience.

7. **Product Designer**: Product designers are involved in the entire design process, from concept to final product. They work on understanding user needs, defining product features, creating prototypes, and collaborating with cross-functional teams to bring the product to market.

8. **Front-End Developer**: Front-end developers implement the visual and interactive elements of a digital product. They work closely with UI/UX designers to translate design mockups into functional and responsive code using HTML, CSS, JavaScript, and other front-end technologies.
9. **Design Consultant**: Design consultants provide expert advice and guidance on UI/UX design projects. They work with clients to understand their needs, offer design solutions, and help improve the overall user experience of their digital products.

10. **Creative Director**: Creative directors oversee the creative vision and direction of a project or organization. They lead design teams, set design standards, and ensure that all design work aligns with the brand's identity and goals.

11. **Accessibility Specialist**: Accessibility specialists focus on ensuring that digital products are accessible to all users, including those with disabilities. They follow accessibility guidelines, conduct audits, and implement changes to improve accessibility.

12. **Design Educator**: Graduates with a passion for teaching can pursue careers as design educators, sharing their knowledge and skills with students in academic institutions or through online platforms and workshops.

13. **Freelance Designer**: Freelance designers work independently, offering their UI/UX design services to various clients and projects. This career path provides flexibility and the opportunity to work on diverse projects.

14. **Innovation Strategist**: Innovation strategists work on identifying and implementing new design trends and technologies to create cutting-edge user experiences. They stay updated with industry developments and explore innovative solutions for digital products.

These career avenues offer a wide range of opportunities for graduates to apply their skills in UI/UX design and make significant contributions to the digital world.

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.

12.1 Eligibility Criteria for Award of Degree

Students must successfully complete the minimum required credits of 175 to be eligible for award of degree without any backlog.

Student's Structured Learning

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 classroom
- Outside classroom

d. Educational Planning and Execution

The B.Tech CSE specialization with UX/UI program at K.R. Mangalam University is designed to foster a holistic educational experience, integrating both theoretical knowledge and practical skills. 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 specialization with UX/UI 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 Introduction to UX design 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.

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.

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.

UX/UI Specialization

In the later stages of the program, students focus on specialized UX/UI courses such as Advanced Interaction Design, Usability Testing & Evaluation, Visual Design Systems, and Designing for Emerging Technologies. These courses are paired with hands-on labs and real-world projects that enhance practical skills in user-centered design. Students gain in-depth knowledge of UX/UI fundamentals, user research methodologies, and design systems, equipping them to tackle industry challenges. The program also integrates advanced certification opportunities like Adobe Certified Professional in UX Design and Human-Computer Interaction (HCI) certifications, ensuring students meet global standards in UX/UI expertise.

Skill Development

The program places a strong emphasis on developing practical skills to ensure students are fully prepared for the demands of the UX/UI and tech industry. Courses on emerging technologies like Artificial Intelligence, Machine Learning, and Cloud Computing, along with specialized UX/UI subjects, equip students with cutting-edge knowledge. Value-Added Courses (VAC) such as Design Thinking & Innovation, AWS Cloud Fundamentals, Software Testing, and Cybersecurity ensure a balance between academic learning and real-world industry needs.

Collaborative projects, internships, and industry-based certification courses (offered in partnership with leading organizations like ImaginXP) further enhance students' professional skills. These

experiences help students build a strong portfolio and gain the expertise needed to excel in a dynamic and competitive 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 Virtual Reality, Wireframing, UX Design for Futuristic Technologies - HMI, Generative AI, and Design Thinking 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.

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.

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.

Career Services and Training: Career Services at K.R. Mangalam University are designed to bridge the gap between academic and professional life. The Career Development and Placement Cell (CDPC) works tirelessly to ensure students are well-prepared for the job market. Through personalized career counseling, resume-building workshops, and industry-specific training, students receive the support they need to secure internships and placements. The university also collaborates with industry partners to conduct mock interviews, aptitude training, and soft skills workshops, ensuring students are equipped with the competencies required to excel in their careers. Additionally, career fairs and campus recruitment drives provide direct employment opportunities for students across various domains

f. Assessment and Evaluation:

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.

Importance of Structured Learning Experiences

- Holistic and Structured Academic Journey: The curriculum focuses on employability, entrepreneurship, and personal development through a balanced education that integrates major/minor selection, internships, projects, coding and hands-on industry experience.

- Comprehensive Learning and Support: Students benefit from experiential learning through labs, workshops, and guest lectures, while academic support, mentor-mentee programs, and counselling services cater to the needs of both slow and advanced learners.
- Continuous Evaluation and Growth: A robust assessment framework with regular feedback, emphasis on academic integrity, and structured evaluation methods ensures ongoing student development and success.

13.3 Educational Planning and Execution

Effective educational planning and execution is a cornerstone of delivering a high-quality academic experience. At the School of Engineering & Technology, we focus on a structured and dynamic approach to ensure that students gain the knowledge, skills, and competencies required to excel in the rapidly evolving technological landscape. Our planning and execution encompass the following key aspects:

- **Curriculum Design:** The curriculum is meticulously crafted, aligning with global standards and the National Education Policy (NEP) 2020. It integrates theoretical learning with practical application, focusing on interdisciplinary knowledge, skill development, and the latest trends in technology, such as AI and ML, Cybersecurity, Full Stack and Data Science etc
- **Learning Objectives:** Each course is designed with clear learning outcomes to ensure students understand the relevance of their education to real-world applications. The focus is on building foundational knowledge while advancing to specialized skills that enhance employability and entrepreneurship.
- **Academic Scheduling:** A well-structured academic calendar is developed, ensuring a balanced distribution of coursework, projects, internships, and examinations. Students are guided in course registration, ensuring a smooth academic journey through carefully timed assessments, practical experiences, and project work.
- **Project-Based Learning:** We emphasize experiential and project-based learning to foster critical thinking, problem-solving, and innovation. Through real-world projects, Internships, contests and collaborative opportunities, students gain practical insights into engineering challenges.
- **Continuous Evaluation:** An ongoing evaluation system is employed with periodic assessments, ensuring continuous learning and improvement. The system includes practical exams, theoretical assessments, internships, and project evaluations, providing a holistic view of student progress. Through strategic educational planning and precise execution, we ensure

that our students graduate with the skills and knowledge required to meet the demands of industry and society.

Scheme Of Studies

Program Name	B.Tech (CSE specialization with UI/UX)
Total Credits	175
Total Semesters	8

Program Name	I	II	III	IV	V	VI	VII	VIII	Total Credits
B. Tech CSE-UX/UI	24	23	28	26	25	23	14	12	175

SEMESTER I

SN	Category	Course Code	Course Title	L	T	P	C	
1	Major1	ENMA101	Engineering Calculus	3	1	-	4	
2	IDC-1	ENSP105	Introduction to UX Design	4	-	-	4	Imagin XP
3	Major-2	ENPH101/ENCH101	Engineering Physics / Engineering Chemistry	3	1	-	4	
4	Major-3	ENCS101	Fundamentals of Computer programming using Python	4	-	-	4	
5	Major-4	ENPH151/ENCH151	Engineering Physics lab / Engineering Chemistry lab	-	-	2	1	
6	VAC-I	VAC-151	Environmental Studies & Disaster Management	2	-	-	2	
7	SEC-1	SEC033	Engineering Drawing & Workshop Lab	-	-	4	2	
8	Major-5	ENCS151	Fundamentals of Computer Programming Lab	-	-	2	1	
9	MOOC-1	SEC067	Essentials of Computer Science (MOOC-I)	-	-	-	2	

TOTAL	16	2	8	24	

Note:

- Course "Essentials of Computer Science" will be offered in an online self-paced mode. Students will be required to complete the suggested online module and produce the certification. Marks shall be allocated based on internal evaluation of 100 marks.
- Value added Course on “Environmental Studies & Disaster Management” will be provided to all the students.

SEMESTER II

SN	Category	Course Code	Course Title	L	T	P	C	
1	Major-6	ENMA102	Linear Algebra and Ordinary Differential Equations	3	1	-	4	
2	IDC-2	ENSP160	Empathy and Understanding Problems Lab	-	-	4	2	Imagin XP
3	Major-7	ENCH101/ENPH101	Engineering Chemistry / Engineering Physics	3	1	-	4	
4	Major-8	ENCS102	Object Oriented Programming using C++	4	-	-	4	
5	Major-9	ENCH151/ENPH151	Engineering Chemistry Lab/Engineering Physics lab	-	-	2	1	
6	Major-10	ENCS152	Object Oriented Programming using C++ Lab	-	-	2	1	
7	Open Elective-1		Open Elective-1	3	-	-	3	
8	Proj-1	ENSI152	Minor Project-I	-	-	-	2	
9	SEC-2		Applied Generative AI: Practical Tools and Techniques	-	-	4	2	
TOTAL				13	2	12	23	

Note:

- The marks for the Minor Project (ENS152) will be allocated solely through internal evaluation, with a total of 100 marks. There will be no end-term exams for this project.
- Students are required to choose one of the various open electives offered by the university. These electives will culminate in an end-term examination worth 100 marks.

SEMESTER III

SN	Category	Course Code	Course Title	L	T	P	C	
1	Major-11	ENCS201	Java Programming	3	1	-	4	
2	Major-12	ENCS205	Data Structures	4	-	-	4	
3	IDC-3	ENSP203	User Research	4	-	-	4	Imagin XP
4	SEC-4	SEC043	Technology in Experience Design	2			2	Imagin XP
5	Major-13	ENCS253	Data Structures Lab	-	-	2	1	
6	IDC-4	ENSP255	User Research Lab	-	-	2	1	Imagin XP
7	Open Elective-2		Open Elective -II	3	-	-	3	Open Elective
8	VAC-2		Students can choose any one elective from the pool of the VAC Courses offered by School	2	-	-	2	
9	AEC-1	AEC006	Verbal Ability	3	-	-	3	
10	INT-1	ENSI251	Summer Internship-I	-	-	-	2	
11	AUDIT-1		Competitive Coding - I	3	-	-	0	
12	Major-14	ENCS251	Java Programming Lab	-	-	2	1	
13	**CS-1**	**CS002**	**Community Service**	**1**	**-**	**-**	**1**	

TOTAL	25	1	6	28	

Note:

- For the "Summer Internship," students are required to complete a 6-week internship during the summer break and submit a completion certificate. The evaluation, which is based on learning outcomes achieved during the internship, will be conducted in the 3rd semester and will be graded on a scale of 100 marks.
- Audit Course: it is zero credit and must pass course. End term exam of 100 marks will be conducted

- Students have the option to enroll in one of the Value-Added Courses (hands-on courses) offered at the school level. This elective course will conclude with an end-term examination worth 100 marks.

**VAC-2					
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

SN	Category	Course Code	Course Title	L	T	P	C	
1	Major-15	ENCS202	Analysis and Design of Algorithms	3	1	-	4	
2	Major-16	ENCS204	Database Management Systems	3	1	-	4	
3	IDC-5	ENSP210	Introduction to UI Design	4	-	-	4	Imagin XP
4	Major-17	ENCS256	Analysis and Design of Algorithms Lab	-	-	2	1	
5	Major-18	ENCS254	Database Management Systems Lab	-	-	2	1	
6	AEC-2	AEC007	Communication & Personality Development	3	-	-	3	
7	Open Elective-3		Open Elective -III	3	-	-	3	
8	SEC-5	SEC044	Information Architecture	2			2	Imagin XP
9	IDC-6	ENSP260	UI Design Lab	-	-	2	1	
10	Proj-2	ENSI252	Minor Project-II	-	-	-	2	
11	AUDIT-2		Competitive Coding- II	3	-	-	0	
12	**CS-2**	**CS001**	**Club/Society**	**1**	-	-	**1**	
TOTAL				22	2	6	26	

Note:

- For the "Minor Project," students will undergo internal evaluation, which will be graded on a scale of 100 marks.
- Audit Course: it is zero credit and must pass course. End term exam of 100 marks will be conducted

SEMESTER V

SNo	Category	Course Code	Course Title	L	T	P	C	
1	Major-19	ENCS301	Theory of Computation	3	1	-	4	
2	Major-20	ENCS303	Operating Systems	3	1	-	4	
3	SEC-6	SEC045	Introduction to Interaction Design	2		-	2	Imagin XP
4	IDC-7	ENSP315	Design Thinking	4	-	-	4	Imagin XP
5	Major-21	ENCS351	Operating System Lab	-	-	2	1	
6	IDC-8	ENSP365	Design Thinking Lab	-	-	2	1	Imagin XP
7	AEC-3	AEC008	Arithmetic and Reasoning Skills-I	3	-	-	3	
8	INT-2	ENSI351	Summer Internship-II	-	-	-	2	
9	IDC-9	ENSP317	Usability Testing	4	-	-	4	Imagin XP
10	AUDIT-3		Competitive Coding - III	2	-	-	0	
TOTAL				21	2	4	25	

Note:

- For the "Summer Internship," students must complete a 6-week internship during the summer and submit a completion certificate. The evaluation will take place during the 3rd semester and will be graded on a scale of 100 marks.
- Audit Course: it is zero credit and must pass course. End term exam of 100 marks will be conducted

SEMESTER VI

SNo	Category	Course Code	Course Title	L	T	P	C	
1	Major-22	ENCS302	Computer Organization & Architecture	3	1	-	4	
2	DSE-1		Discipline Specific Elective -I	4	-	-	4	
3	Major-23	ENCS304	Computer Networks	3	1	-	4	
4	Major-24		Web Development with HTML/CSS/Java Script/Python	4	-	-	4	
5	DSE-2		Discipline Specific Elective -I Lab	-	-	2	1	
6	Major-25	ENCS352	Computer Networks Lab	-	-	2	1	
7	Major-26	ENCS358	Web Development Lab	-	-	2	1	
8	IDC-10	ENSP368	Visual Design Tools Lab	-	-	4	2	
9	Proj-3	ENSI352	Minor Project-III	-	-	-	2	
10	AUDIT-4		Competitive Coding- IV	2	-	-	0	
TOTAL				16	2	10	23	

Note:

- For the "Minor Project," students will undergo internal evaluation, which will be graded on a scale of 100 marks.
- Audit Course: it is zero credit and must pass course. End term exam of 100 marks will be conducted
- Students can choose one of the following department electives:

Discipline Specific Elective I								
(i)	DSE	ENSP318	Wireframing & Prototyping	4	-	-	4	Imagin XP
	DSE	ENSP370	Wireframing & Prototyping Lab	-	-	2	1	Imagin XP
(ii)	DSE	ENSP320	Metaverse & its Applications	4	-	-	4	
	DSE	ENSP372	Metaverse Lab	-	-	2	1	
(iii)	DSE	ENSP322	Introduction to Virtual Reality and Augmented Reality	4	-	-	4	
	DSE	ENSP374	Virtual Reality and Augmented Reality Lab	-	-	2	1	

SEMESTER VII

SN	Category	Course Code	Course Title	L	T	P	C	
1	DSE-3		Discipline Specific Elective-II	4	-	-	4	Imagin XP
2	DSE-4		Discipline Specific Elective-III	4	-	-	4	Imagin XP
3	SEC-7	SEC046	Portfolio Development & Review	2	-	-	2	Imagin XP
4	INT-3	ENSI451	Summer Internship-III	-	-	-	2	
5	MOOC-2		Applied Programming and Problem-Solving Skills for Campus Interviews	-	-	-	2	
TOTAL				**10**	**0**	**0**	**14**	

Note:

- For the "Summer Internship," students must complete a 6-week internship during the summer and submit a completion certificate. The evaluation will take place during the 7th semester and will be graded on a scale of 100 marks.
- Students will be required to undergo a special placement preparation boot camp program offered by the CDC/School in hybrid mode. No end term exams to be conducted, only internal evaluations will be done out of 100 marks. The modules will focus on preparing students for technical interviews, coding questions, aptitude and soft skills, and mock interviews.
- Students can choose among the following department electives:

Discipline Specific Elective - II								
1	DSE	ENSP425	UX Design for Futuristic Technologies - HMI	4	-	-	4	Imagin XP
2	DSE	ENSP427	Introduction to Computer Vision	4	-	-	4	
3	DSE	ENSP429	Graphics Design	4	-	-	4	

Discipline Specific Elective - III	

1	DSE	ENSP431	Design Thinking for Product Management	4	-	-	4	Imagin XP
2	DSE	ENSP433	Adobe Photoshop	4	-	-	4	
3	DSE	ENSP435	Introduction to Figma	4	-	-	4	

Semester VIII

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

Note:

- 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.

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).

Evaluation Scheme (Laboratory):

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

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

Syllabus

Semester: 1

ENGINEERING CALCULUS

Program Name	B.Tech (CSE) with specialization in UI/UX		
Course Name: Engineering Calculus	**Course Code**	**L-T-P**	**Credits**
	ENMA101	3-1-0	4
Type of Course:	Major		
Total Contact Hours	40		
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).
- Center of mass and center 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 Experience

Classroom Learning Experience

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

Outside Classroom Learning Experience

- **Theory Assignments**: Apply calculus techniques in take-home assignments.
- **Lab Projects**: Work on hands-on calculus applications using software.
- **Question Bank**: Practice with model papers and self-assessment.
- **Online Forums**: Discuss and collaborate on calculus problems online.
- **Self-Study for Case Studies**: Research and apply calculus to real-world scenarios.

- **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

INTRODUCTION TO UX DESIGN

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Introduction to UX Design	**ENSP105**	4-0-0	4
Type of Course:	Industry Driven Course		
Pre-requisite(s), if any:			

Course Perspective. To understand the concept of UX design and how it has evolved. Able to understand UX design process and methodology. Able to understand how UX industry work. To know the job, roles and responsibilities in UX industry. To understand the importance of UX in digitalization and different types of industries. The course is divided into 4 modules:

a) Evolution of UX Design

b) Process & Methodologies

c) Tools & Technology in UX Design

d) Multiple domains & trends in UX Design

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

COs	Statements
CO1	**Understanding** the concept of UX design and how it has evolved
CO2	**Defining** and recognize the importance of UX in digitalization and different types of industries.
CO3	**Examining** UX design process and methodology.
CO4	**Evaluating** how UX industry works.

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: Evolution of UX Design**	**No. of hours: 6**
Content Summary: Understand the evolution of UX design as an industry practice and learning about UX industryexperts, Design around us, Job roles and responsibilities in the UX industry		
Unit Number: 2	**Title: Process & Methodologies**	**No. of hours: 6**
Content Summary: Understanding UX design processes and methodologies – user centered design, 5S model,		
Unit Number: 3	**Title: Tools & Technology in UX Design**	**No. of hours: 9**
Content Summary: Tools, prototype, Industry standards, Technology, NFC, Chatbot, Siri		
Unit Number: 4	**Multiple domains & trends in UX Design**	**No. of hours: 9**
Content Summary: UX industry trends in various sectors		

Learning Experience

Classroom Learning Experience

- Interactive Lectures: Utilize presentations and visual aids to introduce key UX design principles, focusing on concepts like user research, wireframing, and prototyping. Demonstrate the use of design tools such as Figma or Adobe XD.
- Theory Assignments: Assign problem-based tasks where students analyze existing user interfaces or create basic user personas and journey maps, applying UX design theories to improve user experiences in real-world applications.
- Lab Projects: Conduct hands-on lab sessions where students use UX design tools like Figma or Sketch to design prototypes for user interfaces. Encourage students to work through iterations based on user feedback or specific design briefs.

Outside Classroom Learning Experience

- Question Bank & Model Papers: Provide a question bank and model papers for continuous assessment and exam preparation.
- Group Work: Facilitate collaborative projects and problem-solving to enhance peer learning and teamwork.
- Continuous Feedback: Offer regular assessments and feedback, with support available through office hours and online forums.
- Case Studies: Use real-world case studies to connect theoretical concepts with practical applications.

Text Books

1. Designing for Digital Age: How to create human-centered products and services -Kim Goodwin

Reference Books

1. Sketching the User experiences - Bill Buxton
2. The design of everyday things - Don Norman
3. The elements of user experience - Jesse James Garrett

Additional Readings:

1. Explore Historical Context: Encourage learners to explore the historical evolution of UX design through interactive timelines and interviews with industry pioneers, enhancing their understanding of its development and impact on various industries.

2. Interactive Process Learning: Utilize simulations and interactive case studies to help learners apply the UX design processes and methodologies like user-centered design and the 5S model in real-world scenarios.

3. Tool Proficiency: Offer practical exercises that allow learners to gain hands-on experience with current UX tools and technologies such as prototyping software, NFC, and voice interaction systems like Siri.

4. Trend Analysis: Provide resources for analyzing current trends in UX design across different sectors, including healthcare, finance, and technology, to help learners predict and adapt to emerging demands.

ENGINEERING PHYSICS

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Engineering Physics	**ENPH101**	3-1-0	4
Type of Course:	Major		
Pre-requisite(s), if any: Integration/Differentiation			

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 nanomaterials in engineering contexts.

CO 4	**Designing** and proposing innovative applications of lasers, fiber optics, and smart materials for specific engineering challenges.
CO 5	**Analyzing** and solving 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

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

Outside Classroom Learning Experience

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

Text Book

1. Fundamentals of Physics" by David Halliday, Robert Resnick, and Jearl Walker

Reference Books

1. University Physics with Modern Physics" by Hugh D. Young and Roger A. Freedman
2. Engineering Physics" by Gaur and Gupta
3. Concepts of Modern Physics" by Arthur Beiser
4. 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

R 3. YouTube - Stanford University Physics Lectures

- This channel features recorded lectures from Stanford University's physics department, covering advanced topics such as quantum mechanics and general relativity.
- Link: Stanford Physics YouTube

ENGINEERING PHYSICS LAB

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Engineering Physics Lab	**ENPH151**	0-0-2	1
Type of Course:	Major		
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 3	**Analyzing** the accuracy and reliability of experimental measurements and results obtained from the conducted experiments.

CO 4	**Evaluating** 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

Ex. No	Experiment Title	Mapped CO/COs
1	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
2	To determine the moment of inertia of a flywheel about its own axis of motion.	CO1, CO2, CO3, CO4
3	To determine the value of acceleration due to gravity using Kater`s pendulum.	CO1, CO2, CO3, CO4
4	To determine the wavelength of sodium light using Newton`s ring apparatus.	CO1, CO2, CO3
5	To determine the wavelength of prominent lines of mercury by plane diffraction grating.	CO1, CO2, CO3
6	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
7	To determine the specific rotation of cane sugar solution with the help of half shade polarimeter.	CO1, CO2, CO3, CO4
8	To determine the wavelength of He-Ne LASER using transmission diffraction grating.	CO1, CO2, CO3

FUNDAMENTALS OF COMPUTER PROGRAMMING

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Fundamentals of Computer Programming	**ENCS101**	4-0-0	4
Type of Course:	Major		
Pre-requisite(s), if any: None			

Course Perspective. This course introduces students to the foundational aspects of computer programming, with a focus on understanding computer fundamentals, programming in Python, and applying these skills to data pre-processing, classification, and visualization. Designed to equip students with both theoretical knowledge and practical programming skills, the course bridges the gap between computer science principles and their application in real-world scenarios.

The course is structured into four comprehensive modules:

a) Getting started with Python

b) Control Flow and Data Structures in Python

c) Object-Oriented Programming and File Handling in Python

d) Control Flow and Data Structures in Python

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

COs	Statements
CO 1	**Analyzing** the structure and functionality of computer systems including hardware, software, and operating system components to understand their roles in computing environments.
CO 2	**Applying** Python programming skills to solve problems involving data types, control structures, functions, and error handling effectively in varied programming scenarios.
CO 3	**Designing** object-oriented programs using Python to demonstrate understanding of classes, objects, inheritance, and polymorphism in developing reusable and modular code.

CO 4	**Evaluating** data pre-processing techniques and implement appropriate visualization strategies using Python libraries to enhance the interpretation of complex datasets.

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: Getting started with Python**	**No. of hours: 10**
Content: **Introduction to Python Programming:** History of Python, Python Features, Local Environment Setup, setting up Python environment: Installing Python, IDEs (e.g., VSCode, Anaconda, PyCharm); **Python Programming Basics:** Python Syntax, Keywords, Understanding Variables, numbers, and data types. **Operators:** Arithmetic, Assignment, Comparison, Logical, Identity, Membership, Bitwise. **Python String:** Manipulating strings, Modify Strings, String Concatenation, Format – Strings, Escape Characters, Inbuilt method of Strings.		
Unit Number: 2	**Title: Control Flow and Data Structures in Python**	**No. of hours: 10**
Content: **Conditional statements**: if, elif, else; Loops: for loop, while loop, nested loops; Control flow statements: break, continue. **Functions**: Defining functions, parameters, function calls, return statement; Scope and lifetime of variables. Recursive and Lambda Functions. **Basics of Python Data Structure**: Mutable: List, Dictionary, Set, Immutable Types: Numbers, String, tuple. **Lists**: Operations, methods, slicing; Tuples and sets: Properties, operations; Dictionaries: Creating, accessing, modifying.		
Unit Number: 3	**Title: Object-Oriented Programming and File Handling in Python**	**No. of hours: 10**
Content: **OOPs Concept:** Introduction to object-oriented programming (OOP), Abstraction, encapsulation, Polymorphism; **Classes and objects:** Defining classes, creating objects; Constructors in Python – Parameterized and Non-parameterized.		

Inheritance, and polymorphism: Types, Method overriding and overloading; Special methods (dunder methods): __init__, __str__, __repr__ . **File handling:** Opening, reading, writing, and closing files; **Exception handling**: try, except, finally blocks.		
Unit Number: 4	**Title: Data handling and web development in python**	**No. of hours: 10**
Content: **Data visualization with matplotlib:** line plot, multiple subplots in one figure, histograms, bar charts, pie charts, scatter plots **Handling data with pandas:** series, dataframes, read and write csv file, operations using dataframe **Numpy arrays:** numpy - datatype, array operations, statistical functions,. **Introduction to Web Development with Python:** Overview of web development Basics of client-server architecture.		

Learning Experiences

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

Outside Classroom Learning Experience

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

Text Books

1. John V Guttag. “Introduction to Computation and Programming Using Python”, Prentice Hall of India
2. R. Nageswara Rao, “Core Python Programming”, Dreamtech
3. Wesley J. Chun. “Core Python Programming, Second Edition”, Prentice Hall
4. Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, “Data Structures and Algorithms in Python”, Wiley

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
3. **GitHub Learning Lab**
    - An interactive learning platform where you can learn to code and collaborate on projects directly within GitHub, which is an essential tool for programmers.
    - Link: GitHub Learning Lab
4. **LeetCode**
    - Ideal for practicing Python coding skills through interactive coding challenges and problems, particularly useful for preparing for technical job interviews.
    - Link: LeetCode

ENGINEERING DRAWING & WORKSHOP LAB

Program Name	**B.Tech(CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Engineering Drawing and Workshop Lab	**SEC033**	0-0-4	2
Type of Course:	SEC		
Pre-requisite(s), if any:			

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	**Exposuring** to some of the advanced and latest manufacturing techniques being employed in the industry.

EXPERIMENT NO.	EXPERIMENT TITLE	MAPPED CO/PO
	ENGINEERING GRAPHICS & DRAWING	
1	Manual drafting of basic geometrical shapes using set squares, and compass.	CO1
2	Understand and draw different projections, points, quadrants and lines.	CO1
3	Draw orthographic projections of simple objects like cubes, cylinders, and prisms.	CO1
4	Draw isometric view of simple components/assemblies.	CO1
5	Understanding of CAD Systems and basic commands of AutoCAD.	CO1
6	Draw orthographic projections by using AutoCAD tools.	CO1
7	Draw projections using an open-source tool like Libre CAD, FreeCAD.	CO2
8	Create a 3D model of simple objects (like a nut and bolt) by using AutoCAD or similar software.	CO2
9	Draw and assemble a small mechanical device (like a piston or gear assembly) using CAD software	CO2
10	Design and draw a 3D model of simple machine, incorporating all the skills learned (Mini Project)	CO2
	WORKSHOP TECHNOLOGY	
1	Understand the basic fitting operations.	CO3
2	Create simple components and joints in Carpentry Shop.	CO3
3	Study and practical exposure to different metal joining techniques.	CO3

4	Manufacturing of metallic parts by using different machine tools.	CO3
5	Understand basic sheet metal operations and make a sheet metal job.	CO4
6	Understanding of electrical connections with practical demonstration.	CO4
7	Learn the fundamentals of CNC machines and create metallic components using NC Code.	CO4
8	Exposure to the advanced additive manufacturing techniques, 3D Printing to create simple parts.	CO4
9	Understand the importance of safety in the workshop.	CO5
10	Assembly of manufactured parts in terms of small machine (Mini Project).	CO5

Learning Experiences

- Hands-on practice with manual drafting, fabrication, and CAD tools enhances real-world engineering skills.
- Technology integration using AutoCAD, CNC machines, and 3D printing prepares students for modern industry standards.
- Collaborative projects foster teamwork and problem-solving in practical engineering applications.
- Continuous assessment and personalized feedback ensure consistent progress and improvement.
- Access to digital resources via Moodle LMS and video lectures supports flexible and self-paced learning.
- Emphasis on safety teaches proper use of tools and equipment, ensuring professional practices.
- Exposure to latest techniques keeps students updated with advanced manufacturing methods and technologies.

Web References

- https://www.cartercenter.org/resources/pdfs/health/ephti/library/lecture_notes/env_health_science_students/engineeringdrawing.pdf
- https://cdn.upgrad.com/uploads/production/418bce1a-1177-4b36-a5a5-7f77199cf72f/Summary+Doc+Session+1.pdf

- https://www.bcsd.org/site/handlers/filedownload.ashx?moduleinstanceid=1059&dataid=5082&FileName=TechnicalDrawingBasics.pdf
- https://testbook.com/objective-questions/mcq-on-engineering-drawing--5eea6a1439140f30f369f2cd

FUNDAMENTALS OF COMPUTER PROGRAMMING LAB

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Fundamentals of Computer Programming lab	**ENCS151**	0-0-2	1
Type of Course:	Major		

Defined Course Outcomes

CO1	**Demonstrating** proficiency in using Python's development environment and effectively utilize basic and advanced programming constructs to solve problems.
CO 2	**Developing** Python programs that incorporate fundamental data structures, control flows, and functions to process and manipulate data
CO 3	**Constructing** object-oriented Python applications demonstrating an understanding of classes, objects, inheritance, and polymorphism to solve complex problems
CO 4	**Implementing** the effectiveness of algorithms involving searching, sorting, and recursion in Python, and visualize data using Python's libraries to communicate results clearly and effectively

Lab Experiments

S.N	Lab Task	Mapped CO/COs
1	Write a Python function that takes two numbers as input and returns a string indicating whether the first number is greater than, less than, or equal to the second number using comparison operators.	CO1
2	Create a Python program that functions as a calculator. It should support addition, subtraction, multiplication, division, and modulus operations. Maintain a history of calculations performed and provide an option to display the history. Implement error handling for division by zero.	CO1
3	Develop a Python function that takes a list of tuples as input, where each tuple contains two numbers. For each tuple, compare the numbers using all comparison operators and store the results in a dictionary. Return a list of dictionaries for all comparisons.	CO1
4	Write a Python program to simulate the operations of basic logic gates (AND, OR, NOT, NAND, NOR, XOR) using functions. The program should take two boolean inputs and the type of gate, then return the result of the logical operation.	CO1
5	Create a Python function to check if any permutation of an input string is a palindrome. For example, the string "civic" is a palindrome, and "ivicc" is a permutation of "civic". The function should return True if any permutation is a palindrome, otherwise False.	CO1
6	Develop a Python function that takes a string and a substring as input and returns the number of times the substring appears in the string. Implement this without using built-in string functions and optimize it for large inputs.	CO1
7	Write a Python function that performs bitwise operations (AND, OR, XOR) on two arrays of integers. The function should take two arrays and an operator as input, perform the bitwise operation element-wise, and return the resulting array.	CO1
8	Create a class CustomString that mimics some of the built-in string methods in Python. Implement methods like find(), replace(), split(), and join(). The class should be able to handle these operations efficiently and correctly.	CO1
9	Develop a classic Snake game using Python. The game should allow users to control a snake that moves around the screen, eating food to grow longer while avoiding collisions with the walls or its own tail.	CO1
10	Create a Python function that finds all prime numbers up to a given number n. Use nested loops to check for primality and optimize the function to handle large inputs efficiently. Additionally, allow the user to break the loop prematurely if they input a specific command.	CO2
11	Write a Python program that takes a list of tuples representing student names and grades, and sorts the list by grades using a lambda function. Ensure that if two students have the same grade, they are sorted by their names.	CO2
12	Create a Python function that takes a list of integers and returns a new list containing only the unique elements of the original list. Use a set to remove duplicates and maintain the order of elements as they appear in the original list.	CO2
13	Write a Python function that takes a string of text and returns a dictionary with the frequency of each word in the text. The function should handle punctuation and capitalization correctly and provide a case-insensitive count.	CO2
14	Create a Python function that takes a tuple as an argument and returns a new tuple with the elements reversed. Additionally, demonstrate the use of tuples as function parameters and unpacking tuple elements within the function.	CO2

15	Create a Python function that checks if a given string containing parentheses is balanced. The function should return True if the parentheses are balanced and False otherwise. Use a stack to solve this problem and handle multiple types of parentheses: (), {}, and [].	CO2
16	Create a Python function that takes a list of strings and groups them into anagrams. The function should return a list of lists, where each sublist contains words that are anagrams of each other. Use dictionaries to store and group the anagrams efficiently	CO2
17	Create a class BankAccount that represents a bank account with attributes account_number, account_holder, and balance. Implement methods for depositing, withdrawing, and checking the balance. Override the __str__ and __repr__ methods to provide meaningful string representations of the account. Include error handling to prevent overdrafts.	CO3
18	Create a class Book with attributes title, author, and isbn. Implement a class Library that manages a collection of books. Implement methods to add, remove, and search for books by title or author. Use file handling to save and load the library collection from a file.	CO3
19	Write a Python program that copies the contents of one file to another. Implement functions to open, read, write, and close files. Use exception handling to manage file errors, such as file not found or permission denied.	CO3
20	Create a base class Employee with attributes name and employee_id. Derive classes Manager and Developer from Employee, each with additional attributes and methods specific to their roles. Override methods where necessary and use polymorphism to write a function that prints the details of all employees.	CO3
21	Develop a class Item with attributes name, price, and quantity. Create a class ShoppingCart that manages a collection of items. Implement methods to add, remove, and display items in the cart. Use file handling to save the cart to a file and load it back.	CO3
22	Write a Python program that encrypts and decrypts the contents of a file using a simple Caesar cipher. Implement functions to read the file, encrypt/decrypt its contents, and write the results to another file. Use exception handling to manage file-related errors and ensure that the program handles different types of input files correctly.	CO3
23	**Project Title: Data Visualization Dashboard using Flask** Develop a data visualization dashboard using Flask that allows users to upload datasets (e.g., CSV files), process the data, and generate various visualizations. The dashboard should be interactive, providing options for users to select different types of charts and filter data.	CO4
24	**COVID-19 Data Analysis and Visualization** Create a Python application to analyze and visualize COVID-19 data using pandas, numpy, and matplotlib. The application should read data from a CSV file containing COVID-19 statistic	CO4
25	**Project Title: Blog Web Application using Flask** Create a blog web application using Flask that allows users to create, read, update, and delete blog posts	CO4
26	**Project : Health and Fitness Tracker** Develop a Health and Fitness Tracker that reads user data from a CSV file, including steps taken, calories burned, and heart rate. Use Pandas to clean and preprocess the data, Numpy for statistical analysis to calculate daily averages and trends, and Matplotlib to create line plots for	CO4

	steps and calories over time, histograms for heart rate distribution, and scatter plots for calories vs. steps. Develop a Flask web application to display these visualizations and allow users to upload their CSV files for personalized tracking.	

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
- **GitHub**
    - Not just a code repository, GitHub offers collaborative features and a wealth of open-source projects where students can engage in real-world software development and contribute to ongoing projects.
    - Link: GitHub

Essentials of Computer Science

Program Name	B.Tech (CSE) with specialization in UI/UX		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Essentials of Computer Science	**SEC067**	0-0-0	2
Type of Course:	MOOC		
Pre-requisite(s): Nil			

Course Perspective. This course introduces students to the Fundamentals of Computer Science, a foundational area that underpins many modern technological principles and practices. The Fundamentals of Computer Science explores the essential concepts and techniques used in computing and programming, emphasizing both theoretical understanding and practical application. The course covers a range of topics from basic computer architecture and programming languages to advanced concepts like data structures, algorithms, software engineering, and web development. This comprehensive approach equips students with the skills necessary to analyze, model, and solve complex computing problems. The course is divided into 4 units:

a) Introduction to Computer Science and Programming Basics

b) Data Structures and Algorithms

c) Software Development and Engineering

d) Advanced Topics and Applications

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

COs	Statements
CO 1	**Understanding** the foundational principles and history of computer science.
CO 2	**Developing** basic understanding of fundamental programming skills using languages such as Python, C++, and JavaScript.

CO 3	**Analyzing** software development and engineering principles to design and implement solutions, including web development, database management, and basic cybersecurity protocols.
CO 4	**Evaluating** essential data structures and algorithms to determine their efficiency and suitability for various 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 Computer Science and Programming Basics**	**No. of hours: NA**

Content:

- **Introduction to Computer Science**
  - Overview of computer science
  - History and impact of computing
  - Basic computer architecture
  - Number system & conversion
- **Basics of Programming**
  - Introduction to programming languages (Python, C, JavaScript)
  - Basic syntax and semantics
  - Writing and running simple programs
- **Problem-Solving Techniques**
  - Algorithms and pseudocode
  - Debugging and error handling
  - Basic problem-solving strategies
- **Basics of Windows and Linux Commands**
  - Introduction to operating systems
  - Basic Windows commands (e.g., dir, copy, del)
  - Basic Linux commands (e.g., ls, cp, rm)
  - File system navigation and management
  - Understanding working environments and command-line interfaces
- **Introduction to Networks**
  - Basics of computer networks
  - Network topologies and protocols
  - Introduction to the Internet and how it works

Sources:

- CS50's Introduction to Computer Science
- Computer Science 101 by Stanford University
- Introduction to Computer Science (Udemy)
- IT Fundamentals - Everything you need to know about IT (Udemy)
- Master Computer Fundamentals Course - Beginner to Intermediate (Udemy)

Unit Number: 2	Title: Data Structures and Algorithms	No. of hours: NA

Content:

- **Basic Data Structures**
  - Arrays and lists
  - Stacks and queues
  - Linked lists
- **Algorithms**
  - Sorting algorithms (bubble sort, merge sort, quicksort)
  - Searching algorithms (linear search, binary search)
  - Algorithm analysis (time and space complexity)
- **Recursion**
  - Introduction to recursion
  - Recursive problem solving
  - Examples of recursive algorithms (e.g., factorial, Fibonacci sequence)

Sources:

- Fundamentals of Computing by Rice University
- CS50's Introduction to Computer Science
- Introduction to Computer Science (Udemy)
- Computer Science 101 by Stanford University

Unit Number: 3	Title: Software Development and Engineering	No. of hours: NA

Content:

- **Software Development Lifecycle**
  - Requirements analysis
  - Design and architecture
  - Implementation and testing
- **Programming Paradigms**
  - Procedural programming
  - Object-oriented programming
  - Functional programming
- **Software Tools and Environment**
  - Integrated Development Environments (IDEs)
  - Version control systems (Git)
  - Debugging and profiling tools
- **Open-Source Tools**
  - Introduction to open-source tools and platforms
  - Using GitHub for version control and collaboration
  - Data science and machine learning with Kaggle
  - Other useful open-source tools (e.g., Jupyter Notebooks, Visual Studio Code)
- **Agile Methodologies**
  - Introduction to Agile principles
  - Scrum framework
  - Kanban and other Agile methodologies

Sources:

- CS50's Introduction to Computer Science
- Fundamentals of Computing by Rice University
- Computer Science 101 by Stanford University
- Computer Science 101 - Computers & Programming for Beginners (Udemy)
- IT Fundamentals - Everything you need to know about IT (Udemy)

Unit Number: 4	Title: Advanced Topics and Applications	No. of hours: NA

Content:

- **Web Development**
  - HTML, CSS, and JavaScript
  - Client-server architecture
  - Introduction to web frameworks
- **Databases**
  - SQL and relational databases
  - NoSQL databases
  - Basic database design and querying
- **Cybersecurity Basics**
  - Principles of cybersecurity
  - Common threats and vulnerabilities
  - Basic encryption and security protocols
- **Latest Technologies and Careers in Computer Science**
  - Overview of latest technologies (e.g., AI, blockchain, IoT, cloud computing)
  - Emerging domains in computer science
  - Various careers in computer science
  - Skills and qualifications needed for different career paths
- **Introduction to Machine Learning**
  - Basic concepts of machine learning
  - Types of machine learning (supervised, unsupervised, reinforcement learning)
  - Introduction to neural networks

Sources:

- CS50's Introduction to Computer Science
- Computer Science 101 by Stanford University
- Fundamentals of Computing by Rice University
- Introduction to Computer Science (Udemy)

Learning experiences for the course:

- Interactive Tutorials and Quizzes: Pre-recorded video lectures with interactive quizzes at regular intervals to reinforce key programming and computer science concepts.
- Self-Guided Coding Exercises: Students will engage in hands-on coding tasks in Python, C, and JavaScript through online platforms like Replit or CodeSandbox, with instant feedback and automated testing.

- Discussion Forums and Peer Reviews: Students will collaborate and share knowledge via discussion boards and peer reviews, fostering a supportive learning environment.

- Project-Based Learning: Learners will complete individual projects, such as developing a basic web application or implementing algorithms, with clear guidelines and milestones.
- Instructor Support and Feedback: Instructors will provide periodic feedback on assignments and be available for Q&A through virtual office hours or email support.

Students can choose any one of the following online modules for certification.

1. **CS50's Introduction to Computer Science by Harvard University**
    - Platform: Harvard Online Learning
    - Link: CS50's Introduction to Computer Science
2. **Computer Science 101 by Stanford University**
    - Platform: Stanford Online
    - Link: Computer Science 101
3. **Fundamentals of Computing by Rice University**
    - Platform: Coursera
    - Link: Fundamentals of Computing
4. **Introduction to Computer Science**
    - Platform: Udemy
    - Link: https://www.udemy.com/course/introduction-to-computer-science/
5. **Computer Science 101 - Computers & Programming for Beginners**
    - Platform: Udemy
    - Link: https://www.udemy.com/course/computer-science-101-computers-programming-for-beginners/
6. **IT Fundamentals - Everything you need to know about IT**
    - Platform: Udemy
    - Link: https://www.udemy.com/course/it-fundamentals-everything-you-need-to-know-about-it/

7. **Master Computer Fundamentals Course-Beginner to Intermediate**

- Platform: Udemy
- Link: https://www.udemy.com/course/master-computer-fundamentals-skills-beginner-to-intermediate/

Please Note:

1. **Enrollment**: Students must enroll in any one of the above specified courses on their respective platforms.
2. **Certification**: students will be required to submit the course completion certificate as an outcome.
3. **Self-paced**: Students will be required to complete any of the certifications on their own. No physical classes shall be conducted
4. **Assignments**: Complete all assignments, quizzes, and problem sets as required by each course.

SEMESTER: 2

LINEAR ALGEBRA AND ORDINARY DIFFERENTIAL EQUATIONS

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: **Linear Algebra and Ordinary Differential Equations**	**Course Code**	**L-T-P**	**Credits**
	ENMA102	3-1-0	4
Type of Course:	Major		
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	**Remembering** basic concepts in linear algebra and ODEs (e.g., matrices, determinants, eigenvalues, eigenvectors, differential equations).
CO 2	**Understanding** and explain the properties and relationships between different concepts in linear algebra and ODEs.
CO 3	**Applying** matrix operations, solve systems of linear equations, and analyze eigenvalues and eigenvectors in engineering contexts.
CO 4	**Analyzing** and identify the appropriate methods to solve first-order and second-order ODEs based on the problem type.
CO 5	**Assessing** the accuracy and stability of numerical solutions to linear systems and ODEs.

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: 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

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

Outside Classroom Learning Experience

- **Theory Assignments**: Assign take-home projects applying concepts to practical problems.
- **Lab Projects**: Facilitate hands-on projects involving software tools for linear algebra and ODEs.
- **Question Bank**: Provide practice problems and model papers for self-assessment.
- **Online Forums**: Create platforms for students to discuss and collaborate on problems.
- **Self-Study for Case Studies**: Encourage independent research on applications of linear algebra and ODEs.
- **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.

EMPATHY AND UNDERSTANDING PROBLEMS LAB

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Empathy and Understanding Problems Lab	**ENSP160**	0-0-4	2
Type of Course:	IDC		

Defined Course Outcomes

CO1	Developing the ability to empathize with users and understand their needs and problems
CO2	Applying various UX research methods to gather user insights
CO3	Analyzing and synthesizing research findings to define user problems
CO4	Creating user personas and journey maps to illustrate user experiences
CO5	Evaluating solutions and prototypes based on user-centered design principles

List of Experiments

Ex. No	**Experiment Title**	**Mapped CO/COs**
1	Introduction to Empathy and User-Centered Design	CO 1
2	Conducting User Interviews	CO 2
3	Creating Empathy Maps	CO 1, CO 2
4	Performing Contextual Inquiries	CO 2
5	Analyzing User Research Data	CO 3
6	Identifying User Pain Points	CO 3
7	Defining User Problems	CO 3

8	Creating User Personas	CO 4
9	Developing User Journey Maps	CO 4
10	Ideation Techniques: Brainstorming and Mind Mapping	CO 5
11	Sketching Initial Design Concepts	CO 5
12	Prototyping with Paper and Pencil	CO 5
13	Introduction to Digital Prototyping Tools	CO 5
14	Creating Low-Fidelity Prototypes	CO 5
15	Conducting Usability Testing for Low-Fidelity Prototypes	CO 5
16	Iterating Based on Feedback	CO 5
17	Developing High-Fidelity Prototypes	CO 5
18	Conducting Usability Testing for High-Fidelity Prototypes	CO 5
19	Finalizing Design Based on Iterative Feedback	CO 5
20	Presenting Design Solutions	CO 5
21	Case Study Analysis: Successful UX Projects	CO 1, CO 3
22	Case Study Analysis: Failed UX Projects	CO 1, CO 3
23	Group Discussion: Ethical Considerations in UX Research	CO 1
24	Role-Playing Exercises to Understand Diverse Perspectives	CO 1
25	Field Observations: Understanding Real-World User Context	CO 2
26	Survey Design and Implementation	CO 2
27	Data Analysis: Using Surveys for UX Research	CO 2, CO 3
Project 1	User Research and Persona Development	CO 1, CO 2, CO 4
Project 2	Designing a User-Centered Solution	CO 3, CO 5
Project 3	Prototyping and Usability Testing	CO 5

Project 4	Comprehensive UX Project: From Research to Prototype	CO 1, CO 2, CO 3, CO 4, CO 5

ENGINEERING CHEMISTRY

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
ENGINEERING CHEMISTRY	**ENCH101**	4-0-0	4
Type of Course:	Major		
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** fundamental concepts in water analysis, fuel, battery Construction, working and applications, and polymer structures.
CO 2	**Explaining** processes for water hardness determination, fuel energy values, and Ni-Cd, and Lithium-ion battery.
CO 3	**Applying** methods to determine water hardness, fuel energy (calorific value) and analysis of coal, and EMF of the cell and cell representation
CO 4	**Analyzing** issues like boiler scale formation, fuel knocking in engines, and differences among various battery types and polymers.

CO 5	**Evaluating** the efficiency, environmental impact, and industrial suitability of water treatment methods, fuel types, battery technologies, and polymers, such as biodegradable polymers and fuel cells.

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

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

- **Continuous Feedback**: Implement in-class quizzes and feedback sessions to assess understanding.

Outside Classroom Learning Experience

- **Theory Assignments**: Assign take-home projects applying chemistry concepts to engineering challenges.
- **Lab Projects**: Facilitate hands-on experiments that explore chemical principles in practical applications.
- **Question Bank**: Provide practice problems and model papers for self-assessment.
- **Online Forums**: Create platforms for students to discuss and collaborate on chemistry problems.
- **Self-Study for Case Studies**: Encourage independent research on recent advancements in engineering chemistry.
- **Collaborative Projects**: Organize group projects focused on developing sustainable chemical processes or materials.
-

Text Books

1. Principles of Physical Chemistry by B. R. Puri, L. R. Sharma and M. S. Pathania, S. Nagin Chand and Co.
2. Physical Chemistry by Soni and Dharmatha, S. Chand & Sons.
3. 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	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
ENGINEERING CHEMISTRY LAB	**ENCH151**	0-0-2	1
Type of Course:	Major		

Defined Course Outcomes

CO1	**Understanding** principles behind laboratory techniques, equipment operation, and different types of titrations.
CO2	**Applying** precise methods to measure ion concentrations and perform experiments with accuracy.
CO3	**Interpreting** experimental data, recognize patterns, and ensure correct procedures are followed.
CO4	**Evaluating** the accuracy and reliability of results and assess the effectiveness of the methods used.
CO5	**Performing** experiments safely and effectively, following standard laboratory procedures.

List of 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-formaldehyde resin	CO4, 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 Fe3+ in solution	CO2, CO3, CO5
14	Determination of molar absorptivity of a compound (KMnO4 or any water-soluble food colorant).	CO2, CO3, CO5
15	Preparation of a nickel complex [Ni(NH3)6]Cl2 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 (CSE) with specialization in UI/UX**		
Course Name: **Object Oriented Programming using C++**	**Course Code**	**L-T-P**	**Credits**
	ENCS102	4-0-0	4
Type of Course:	Major		
Pre-requisite(s), if any: Basics of C programming			

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

- Lecture PPTs: Structured slides covering key concepts with embedded problem-solving examples.
- Problem-Based Theory Assignments: Assignments integrating theory and practical scenarios for concept application.

- Lab Assignments with Mini Projects: Hands-on coding tasks and mini projects to implement OOP principles.

Outside Classroom Learning Experience

- Question Bank: Comprehensive set of practice questions and coding problems for self-assessment.
- Sample Question Papers: Practice papers to simulate final exams and test understanding.
- Discussion Forums: Online forums for peer discussion and instructor support on assignments and projects.

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 (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Object Oriented Programming using C++ Lab	**ENCS152**	0-0-2	1
Type of Course:	Major		
Pre-requisite(s), if any: Basics of C programming			

Lab Experiments

Defined Course Outcomes

COs	**Lab tasks**
CO 1	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.
CO1	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.
CO1	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.
CO1	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.
CO1	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.

CO1	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.
CO1	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.
CO1	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.
CO2	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.
CO2	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.
CO2	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.
CO2	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.
CO2	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.
CO2	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.
CO3	Develop a Vehicle Management System that demonstrates different types of 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.
CO3	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.
CO3	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.
CO3	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.
CO3	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.
CO3	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.
CO4	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.
CO4	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.
CO4	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.
CO4	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.
CO4	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.
CO4	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.

Minor Project-I

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
COURSE NAME: **Minor Project-I**	**COURSE CODE**	**L-T-P**	**CREDITS**
	ENSI152	0-0-0	2
TYPE OF COURSE:	Project		
PRE-REQUISITE(S), IF ANY: NA			

Course Perspective:

The objective of Minor Project-I 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.

Applied Generative AI: Practical Tools and Techniques

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Applied Generative AI: Practical Tools and Techniques		0-0-4	2
Type of Course:	SEC		

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 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	**CO2**

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 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.	**CO3**
Complex Applications and Ethical Frameworks **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.	**CO4**
Project 1: Intelligent Email Assistant **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. **Objectives:** 1. Explore and understand the basic functionalities of Hugging Face and LangChain. 2. Implement basic prompt engineering to draft and respond to emails. 3. Automate the organization of emails into categories such as work, personal, and promotions. 4. Integrate GenAI tools to ensure seamless operation and user-friendly interaction.	CO1,CO2, 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. **Objectives:** 1. Understand the basic applications of GenAI in content creation for social media. 2. Implement GenAI models to generate posts, captions, and hashtags. 3. Ensure the content generated is relevant, engaging, and tailored to specific platforms. 4. Automate the content creation process to save time and resources.	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. **Objectives:** 1. Understand the ethical implications and challenges associated with generative AI in healthcare. 2. Develop guidelines for the responsible use of generative AI in medical applications. 3. Address issues such as bias, privacy, transparency, and accountability in AI usage. 4. Create a framework that can be adopted by healthcare organizations to ensure ethical AI practices.	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. **Objectives:** 1. Implement GenAI models for analyzing financial data. 2. Use GenAI to generate detailed financial reports based on historical data. 3. Ensure the reports provide accurate and reliable insights for decision-making. 4. Automate the report generation process to save time and improve efficiency.	CO1,CO2, CO3,CO4

Learning Experiences

Classroom Learning Experience

- 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.

Outside Classroom Learning Experience

- 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

SEMESTER: III

JAVA PROGRAMMING

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Java Programming	**ENCS201**	3-1-0	4
Type of Course:	Major		
Pre-requisite(s), if any: Basic of 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)

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 Number: 3	**Title: Exception Handling, Multithreading and Wrapper Class**	**No. of hours: 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

- Interactive Lecture Sessions: Students will engage with lecture PPTs and interactive teaching boards, enhancing their understanding of Java concepts through visual aids and active participation during class discussions.
- Problem-Based Learning: Regular theory assignments will encourage students to apply their knowledge to real-world programming problems, reinforcing key topics and promoting critical thinking.
- Project-Based Lab Work: Students will complete hands-on, project-based lab assignments, applying Java concepts to develop functional programs. This will simulate real-world coding experiences, fostering practical skills.
- Comprehensive Study Support: A question bank covering the entire syllabus and model question papers will be provided, allowing students to self-assess their progress and prepare for exams effectively.
- Continuous Assessment & Feedback: Ongoing evaluations through quizzes, assignments, and lab projects will keep students engaged, while timely feedback ensures improvement in understanding and problem-solving abilities.

Outside Classroom Learning Experience

- Technology-Enhanced Learning: Moodle LMS will serve as a platform for students to access all course materials, including assignments, question banks, and support resources, ensuring continuous learning beyond the classroom.
- Collaborative Learning & Peer Support: Group projects and peer review activities will allow students to collaborate, share knowledge, and support each other's learning, fostering teamwork and communication skills.

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	**B.Tech (CSE) with specialization in UI/UX**		
COURSE NAME: **JAVA PROGRAMMING LAB**	**COURSE CODE**	**L-T-P**	**CREDITS**
	ENCS251	0-0-2	1
TYPE OF COURSE:	Major		
PRE-REQUISITE(S), IF ANY: basic working knowledge of C++ programming will be an added advantage			

DEFINED COURSE OUTCOMES

COS	
CO 1	**Demonstrating** the use of primitive data types, type casting, and basic input/output operations in Java.
CO 2	**Implementing** control structures such as conditional statements and loops to perform arithmetic operations and generate sequences.

CO 3	**Creating** and manipulating arrays and demonstrating basic inheritance, polymorphism, and class hierarchies in Java applications.
CO 4	**Developing** and testing advanced Java applications using multithreading, file handling, collections, and exception handling to solve real-world problems.

LIST OF EXPERIMENTS

	Lab Task	MAPPED CO/COS
1	A Java Application that manages a small library system	**CO1**
2	A Java Application to manage a simple bank account system	**CO1**
3	A Java class hierarchy for a simple educational institution system	**CO2**
4	A system for managing different types of vehicles in a rental service	**CO2**
4	A Java Application for a basic shape drawing application	**CO3**
5	A Java multithreaded application that simulates a banking system	**CO3**
6	A file management system that supports operations such as reading,writing, copying, and navigating files and directories	**CO4**
7	A contact management system that utilizes different data structures like Lists, Sets, Queues, and Maps	**CO4**

DATA STRUCTURES

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: **Data Structure**	**Course Code**	**L-T-P**	**Credits**
	ENCS205	4-0-0	4
Type of Course:	Major		
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 applying basic and advanced data structures.
CO 2	**Analyzing** and comparing various sorting and searching algorithms.
CO 3	**Designing** and utilizing algorithms for advanced data manipulation.
CO 4	**Implementing** and evaluating algorithms using hashing and advanced algorithmic techniques.
CO 5	**Developing** applications that integrate data structures with file I/O operations and handle data dynamically.

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 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: Trees and Graphs**	**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: Advanced Sorting, Searching, and Algorithm Techniques**	**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**
  - **Interactive Lectures:** Introduce key concepts in data structures using PPTs and coding demonstrations.
  - **Conceptual Understanding**: Cover topics like arrays, linked lists, stacks, queues, trees, and graphs.
  - **Problem-Solving Sessions**: Conduct in-class exercises focused on implementing and using various data structures.
  - **Theory Assignments**: Assign theoretical problems that reinforce data structure concepts, discussed in class.
  - **Group Work**: Collaborate on projects that require designing and optimizing data structures.
  - **Case Studies**: Analyze real-world applications of data structures in software development.
  - **Continuous Feedback**: Implement quizzes and peer reviews to assess understanding and coding practices.
  - **Outside Classroom Learning Experience**
  - **Theory Assignments**: Assign take-home projects that apply data structure concepts to practical problems.
  - **Lab Projects**: Facilitate hands-on programming tasks using data structures in real-world scenarios.
  - **Question Bank**: Provide practice problems and resources for self-assessment on data structures.
  - **Online Forums**: Create platforms for discussing data structure challenges and solutions.
  - **Self-Study for Case Studies**: Encourage independent research on efficient data structure implementations.
  - **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

USER RESEARCH

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
User Research	**ENSP203**	4-0-0	4
Type of Course:	IDC		
Pre-requisite(s), if any: NA			

Course Perspective. Students will be able to understand the importance of User research, Understanding the different user research methodologies. Able to grasp hands-on experience of tools for user research. Understanding cognitive psychology and user behavior. Performing user research with users on a chosen problem. The course is divided into 4 modules:

a) Introduction to User Research

b) User Research Methodologies

c) Hands on Practice of Methodologies

d) Tools of Empathy and Analysis

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

COs	Statements
CO1	**Understanding** the fundamental principles and importance of user research to effectively interpret user interactions and behaviors.
CO2	**Defining** and applying various user research methodologies, including the creation of personas, user segments, and effective use of online survey tools.
CO3	**Analyzing** and synthesizing data from different user research methods such as interviews, surveys, and focus groups to draw actionable insights.
CO4	**Employing** empathy and analytical tools such as empathy maps, persona, and user journey maps to enhance the understanding of user needs and improve research outcomes.

CO5	**Designing** and execute a comprehensive user research project, integrating qualitative and quantitative research techniques to address real-world 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 User Research**	**No. of hours: 10**
Content Summary: **Introduction:** Definition, Importance, Applications, planning Designs and improving user experience, identifying potential issues early in the design process to avoid costly mistakes Understanding User interactions, User needs and understanding the context of the user's tasks and goals.		
Unit Number: 2	**Title: User Research Methodologies**	**No. of hours: 10**
Content Summary: User Segment, defining persona for research & recruiting users, desk research, primary research Preparing a Questionnaire for user research, focus group discussion, personal interviews, do and don'ts of interviewing Online surveys - tools, do and don'ts, Analysis Interview Tips & Techniques		
Unit Number: 3	**Title: Field study: Hands on Practice of Methodologies**	**No. of hours: 10**
Content Summary: Preparing and Conducting Stakeholder workshop Preparing questionnaire for Interviews, and Online surveys		
Unit Number: 4	**Title: Research Analysis**	**No. of hours: 10**

Content Summary:

Analyzing qualitative and quantitative results, Transcribing interviews, Thematic analysis, Cluster analysis

Tools of empathy like Persona, Empathy Map, understanding user scenarios, Storyboarding and when to use it

User Journey Map, Steps to create a journey map, AS-IS vs TO-BE journey maps, Documenting Qualitative Research

Learning Experiences

Classroom Learning Experience

- Interactive Lectures: Use PowerPoint slides, videos, and real-life examples to introduce key concepts such as user research methods, the importance of planning, and how to identify potential issues early in the design process.
- Theory Assignments: Challenge students with assignments that involve preparing research plans, creating user personas, and developing questionnaires for interviews and surveys. Assign them real-world projects where they must apply user research methodologies and analyze user needs.
- Hands-on Field Study Projects: Organize workshops where students practice conducting stakeholder interviews, focus group discussions, and online surveys. Have them prepare questionnaires, recruit participants, and collect user data in real-world or simulated environments.
- Lab Sessions: Implement lab sessions where students analyze qualitative and quantitative data collected during field studies. Introduce tools like empathy maps, user journey maps, and storyboarding. Students will use these tools to document and interpret research findings.
- Tools and Techniques: Introduce various user research tools (e.g., personas, empathy maps) and analysis techniques (e.g., thematic and cluster analysis) to enhance the students' understanding of how to interpret research data.

Outside Classroom Learning Experience

- Group Work: Encourage collaborative projects where students conduct user research, analyze data, and present findings as a team. This promotes peer learning and exposes them to different perspectives in research approaches.

- Case Studies: Use real-world case studies to illustrate how user research has impacted design processes, improving user experiences and preventing costly mistakes in product development.
- Continuous Feedback: Offer regular feedback through assessments, provide office hours for one-on-one support, and create online forums where students can ask questions and share experiences about the research process.

Textbooks

1. Practitioner's guide to empathy and user research by Eshayat Taskin, Shashank Shwet, Sonam Agarwal, Vidhika Rohatgi
2. Interviewing Users: How to Uncover Compelling Insights by Steve Portigal
3. Research Design: Quantitative, Qualitative, Mixed Methods, Arts-Based, and Community-Based Participatory Research Approaches by Patricia Leavy
4. User Experience Mapping: Enhance UX with User Story Map, Journey Map and Diagrams by Peter W. Szabo

Additional Readings:

1. **Explore the Essentials:** Dive into the basics of user research to understand user interactions and why they're critical for designing user-centric products and services.
2. **Master Methodologies:** Gain proficiency in various user research methodologies, including creating personas, conducting focus groups, and utilizing online survey tools effectively.
3. **Practical Engagement:** Apply your knowledge by preparing and conducting stakeholder workshops, crafting questionnaires, and executing interviews to gather meaningful user insights.
4. **Empathy and Analysis:** Learn to use tools like empathy maps and user journey maps to deepen your understanding of user needs and effectively document qualitative and quantitative research findings.
5. **Project Implementation:** Put theory into practice by undertaking a project focused on user research, integrating all the skills and knowledge acquired to solve real-world problems.

Software and Development Tools

1. Figma
2. Sketch
3. Miro
4. Google Forms

Datasets

1. User interview transcripts
2. Survey response data
3. Usability test recordings
4. Contextual inquiry notes

Online Platforms and Communities

1. UX Stack Exchange
2. Interaction Design Foundation
3. User Experience Professionals Association (UXPA)
4. Figma Community

Virtual Labs and Additional Resources

1. Nielsen Norman Group UX Research resources
2. Coursera: "UX Research at Scale: Surveys, Analytics, Online Testing" by the University of Michigan
3. edX: "User Experience (UX) Research and Design" by the University of Michigan
4. IDEO U: "Insights for Innovation" course

TECHNOLOGY IN EXPERIENCE DESIGN

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: Technology in experience design	**Course Code**	**L-T-P**	**Credits**
	SEC043	2-0-0	2
Type of Course:	SEC		
Pre-requisite(s), if any: NA			

Course Perspective. Students will Get to know futuristic technologies and their implementation in design. Able to comprehend technology constraints on design. To Understand technology for digital experience and product ecosystems. Research project in design using latest technology. The course is divided into 4 modules:

a) Technology for Experience Design

b) Technological feasibility & viability

c) Futuristic Technologies

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

COs	Statements
CO1	**Defining** futuristic technologies and their implementation in design.
CO2	**Recognizing** and comprehending technology constraints on design.
CO3	**Evaluating** and understanding technology for digital experience and product ecosystems.
CO4	**Analyzing** and applying insights in design project.
CO5	**Designing** and using the latest technology in any existing or new product ecosystem.

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: Technology for digital experience**	**No. of hours: 10**
Content Summary: **Introduction:** Understanding technology for digital experience and product ecosystems – form factors, operating systems, Wi-Fi **Hardware components:** Bluetooth, sensors and other hardware components, Discussing how technology shapes user experiences and product ecosystems **Operating systems:** Overview of iOS, Android, Windows, macOS and their unique features, OS constraints and capabilities influence design decisions.		
Unit Number: 2	**Title: Technological feasibility & viability**	**No. of hours: 10**
Content Summary: Understanding technological feasibility and viability. Technology constraints on design. Techniques for analyzing the viability of a technology for a specific use case. Identifying and addressing technological constraints and discussing common trade-offs between feasibility, viability, and user experience.		
Unit Number: 3	**Title: Futuristic Technologies**	**No. of hours: 10**
Content Summary: Learning about futuristic technologies and their implementation in design, Wearable medical devices - Key considerations when designing for wearable medical devices Introduction to Augmented reality and virtual reality, design Differences between AR and VR and their respective use cases Understanding principles and successful implementations of AR and VR in various industries, Artificial Intelligence and its use cases		
Unit Number: 4	**Futuristic Technologies continued**	**No. of hours: 10**
Content Summary: Introduction to Internet of Things and UX - Understanding the IoT ecosystem, its components and its key considerations.		

Conversational Design - Understanding conversational design, its applications and key principles for designing effective conversational interfaces. How to design for chatbots and voice interfaces

Principles for designing voice user interfaces (VUIs) and Techniques for testing and refining conversational and voice interfaces.

Defining product ecosystems and constraints of key technologies

Learning Experiences

Classroom Learning Experience

- Interactive Lectures: Use engaging presentations, videos, and live demonstrations to introduce students to the role of technology in shaping user experiences.
- Technology Demonstrations: In-class demonstrations on how technologies like Bluetooth, sensors, and other hardware components work. Use physical devices or simulators to help students understand the ecosystem and how it affects the user experience.
- Hands-on Projects: Assign projects where they must analyze trade-offs between technology constraints, feasibility, and user experience. This gives them real-world insight into how technical limitations impact design decisions.
- Lab Sessions: Conduct lab sessions where students work with AR/VR tools or simulators to create basic designs and interfaces.

Outside Classroom Learning Experience

- Case Studies: Use case studies on wearable medical devices, augmented reality (AR), virtual reality (VR), and artificial intelligence (AI). Discuss successful implementations and key considerations when designing for these emerging technologies.
- Futuristic Technology Workshops: Organize workshops that focus on wearable technology, AR/VR, and IoT. Students can engage in hands-on practice designing interfaces for wearables, creating voice interfaces for virtual assistants, and building user flows for IoT devices.
- Group Work: Facilitate group projects where students collaborate to design products that incorporate futuristic technologies like AR, VR, AI, or IoT. This encourages teamwork and allows them to explore the integration of advanced technology into user experience design.
- Support and Feedback: Seek additional help from the course instructor as needed. The instructor will be available for support, and continuous feedback will be provided to guide students' progress.

Textbooks

1. Emotions, technology and design - Sharon Y. Tettegah
2. Augmented Reality: Principles and Practice - Dieter Schmalstieg
3. Augmented Reality: An emerging technologies guide - Gregory Kipper and Joseph Rampolla

Additional Readings:

1. Explore the relationship between device form factors and user experience to understand how technology shapes interactions in digital ecosystems.
2. Investigate the impact of technological constraints on design to enhance feasibility and viability in product development.
3. Analyze various operating systems and hardware components like WiFi and Bluetooth to determine their roles in enhancing user connectivity and functionality.
4. Study the integration of futuristic technologies such as wearable medical devices in modern design to grasp their real-world applications and benefits.
5. Delve into the principles of Internet of Things (IoT), augmented reality (AR), and virtual reality (VR) to understand their transformative effects on both everyday and specialized technologies.

DATA STRUCTURES LAB

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Data Structure lab	**ENCS253**	0-0-2	1
Type of Course:	Major		
Pre-requisite(s), if any: Basics of Computer Programming			

Proposed Lab Experiments

Defined Course Outcomes

COs	
CO 1	**Analyzing** and evaluating the time and space complexity of algorithms for various scenarios, demonstrating an understanding of asymptotic notations.
CO 2	**Implementing** and manipulating single-dimensional and multi-dimensional arrays, including operations like insertion, deletion, and traversal.
CO 3	**Developing** and performing operations on linked lists (single, doubly, and circularly linked), stacks, and queues using both array and linked list representations.
CO 4	**Designing** and analyzing the efficiency of different sorting and searching algorithms, as well as implementing and comparing advanced data structures like binary search trees, AVL trees, and graph algorithms.

S.N	**Experiment Title**	**Mapped CO/COs**
1	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
2	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
3	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

4	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
5	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
6	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
7	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
8	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
9	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
10	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
11	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 the function to handle polynomials of different degrees.	CO2
12	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
13	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
14	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
15	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
16	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
17	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
18	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
19	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.	CO2

20	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
21	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
22	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
23	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
24	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
25	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
26	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
27	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
28	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
29	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
30	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

USER RESEARCH LAB

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: **User Research Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP255	0-0-2	1
Type of Course:	IDC		
Pre-requisite(s), if any: Basics of Computer Programming			

Proposed Lab Experiments

Defined Course Outcomes

COs	
CO 1	**Understanding** and **applying** various user research methodologies
CO 2	**Conducting** user interviews and surveys to gather insights
CO 3	**Analyzing** and synthesizing research data to identify user needs and pain points
CO 4	**Creating** and utilizing user personas and journey maps to inform design decisions

Ex. No	**Experiment Title**	**Mapped CO/COs**
1	Introduction to User Research Methods	CO1
2	Conducting User Interviews	CO2
3	Designing and Distributing Surveys	CO2
4	Analyzing Survey Data	CO3
5	Creating Empathy Maps	CO1, CO3

6	Performing Contextual Inquiries	CO2
7	Analyzing Interview Data	CO3
8	Identifying User Needs and Pain Points	CO3
9	Creating User Personas	CO4
10	Developing User Journey Maps	CO4
11	Affinity Diagramming for Data Synthesis	CO3
12	Conducting Usability Tests	CO2
13	Analyzing Usability Test Results	CO3
14	Role-Playing to Understand Diverse User Perspectives	CO1
15	Heuristic Evaluation of User Interfaces	CO1, CO3
16	Field Observations: Understanding Real-World User Context	CO2
17	Diary Studies for Longitudinal Research	CO2
18	Card Sorting for Information Architecture	CO1, CO3
19	Creating a Research Plan	CO1
20	Presenting Research Findings	CO3, CO4

VERBAL ABILITY

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Life Skills for Professionals - I	**AEC006**	3-0-0	3
Type of Course:	AEC		
Pre-requisite(s), if any:			

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 Skills**	**No. of hours: 6**
Content: Logical Arrangement of Sentences, Comprehending passages, Contextual questions, Anagrams, Analogies		

References

R1. Norman Lewis – Word Power Made Easy
R2. Wren & Martin – High School English Grammar & Composition
R3. R.S. Agarwal & Vikas Agarwal – Quick Learning Objective General English
R4. S.P. Bakshi - Objective General English
R 5. Praxis Groups -Campus Recruitment Complete Reference

Additional Readings:

https://www.indiabix.com/online-test/aptitude-test/

https://www.geeksforgeeks.org/aptitude-questions-and-answers/

https://www.hitbullseye.com/

Summer Internship-I

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: Summer Internship-I	**Course Code**	**L-T-P**	**Credits**
	ENSI251	0-0-0	2
Type of Course:	INT		

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

- Real-world Application of Knowledge: Students will apply theoretical concepts from their coursework to solve industry problems or research tasks in a practical setting.
- Hands-on Experience in Professional Environments: Through industry or research internships, students will gain hands-on technical experience, enhancing their problem-solving abilities and technical competencies.
- Collaboration and Networking: Students will work with industry professionals or academic mentors, fostering collaboration and expanding their professional networks.

- Independent Research and Skill Development: The internship encourages independent learning, allowing students to develop new technical skills and conduct research, resulting in a project, case study, or publishable research paper.
- Structured Feedback and Assessment: Continuous feedback from mentors will help students refine their professional skills, culminating in a well-organized internship report and presentation.
- Presentation and Communication: Students will improve their presentation skills by effectively communicating their findings, using visual aids, and demonstrating their understanding of the work completed during the internship.
- Exposure to Emerging Trends: Engaging with current industry or research projects helps students stay updated with emerging trends and technologies, fostering lifelong learning.

COMPETITIVE CODING -I

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: COMPETITIVE CODING-**I**	**Course Code**	**L-T-P**	**Credits**
		3-0-0	NIL
Type of Course:	Audit Course		
Contact Hours	30		
Pre-requisite(s), if any: Fundamentals of programming			

Course Outcomes

CO1	Understanding and applying 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 and analyzing 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
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

Problem Statement	Mapped COs
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
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:

- Understanding Memory Management: Students grasp the concept of memory allocation and deallocation through pointers, gaining insights into how data is stored and accessed in memory.
- Pointer Arithmetic: Learners practice pointer arithmetic to navigate arrays and structures, enhancing their ability to perform low-level data manipulations efficiently.
- Dynamic Memory Allocation: Students experience dynamic memory allocation with functions like malloc, calloc, and free, learning to manage memory dynamically during runtime.
- Pointer and Function Interactions: Students explore how pointers are used to pass arguments by reference, leading to more efficient function calls and manipulation of data within functions.
- Pointer to Pointer Concepts: Learners work with pointer to pointer (double pointers) to understand multi-level indirection and its applications in complex data structures and dynamic memory management.
- Debugging with Pointers: Students enhance their debugging skills by identifying and fixing pointer-related issues such as memory leaks, dangling pointers, and segmentation faults.

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/

COMMUNITY SERVICE

Program Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name:	**Course Code**	**L-T-P**	**Credits**	**Semester**
Community Service	**CS002**	2-0-0	2	3
Type of Course:	CS			
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 Service** 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	

Criteria	Marks
- 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
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 Service
- 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.

Learning Experiences:

Methodologies for Experiential Learning:

- Fieldwork: Hands-on community service.
- Mentorship: Guiding underprivileged students.
- Workshops: Conduct skill-building sessions.
- Tech Solutions: Develop apps and tools for social causes.
- Survey & Research: Analyze community needs through data collection.
- Problem-solving: Apply technology to real-world issues.
- Collaboration: Work with local NGOs and community centers.

Types of Engagement Activities:

Development and Innovation

- Build apps, assistive devices, and smart solutions for community problems.
- Innovate tools for healthcare, agriculture, or education.

Education and Mentorship

- Offer tutoring and skill-development workshops.

- Organize day camps and educational sessions for children.

Community Service

- Volunteer with local charities.
- Support women empowerment and digital awareness programs.

Environmental & Health Initiatives

- Participate in tree plantation, clean-ups, and conservation drives.
- Organize health awareness campaigns and assist in medical camps.

Technology for Social Good

- Use technology to solve community issues.

Develop tools for environmental monitoring and waste management.

SEMESTER: IV

ANALYSIS AND DESIGN OF ALGORITHMS

Program Name	B.Tech (CSE) with specialization in UI/UX		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Analysis and Design of Algorithms	**ENCS202**	4-0-0	4
Type of Course:	**Major**		
Pre-requisite(s), if any: - Data Structure			

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.

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 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.		
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

- Interactive Lectures and Assignments: Utilize lectures and problem-based assignments to understand fundamental algorithm concepts and complexity analysis.
- Hands-On Lab Sessions: Implement and test various algorithms through lab projects to gain practical experience.
- Group Projects: Collaborate on projects to solve complex algorithmic problems and develop advanced techniques.
- Case Studies: Apply algorithmic strategies to real-world problems through case studies and discussions.
- ICT Tools: Use Moodle LMS for course materials and interactive boards for algorithm visualization.
- Continuous Assessment: Engage in regular quizzes, assignments, and project evaluations with detailed feedback.

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

IV) **TopCoder - Algorithm Tutorials**

a. Detailed tutorials and practice problems on a wide range of algorithmic topics.

b. Link: TopCoder - Algorithm Tutorials

ANALYSIS AND DESIGN OF ALGORITHMS LAB

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Analysis and Design of Algorithms Lab	**ENCS256**	0-0-2	1
Type of Course:	Major		
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	Build a web application that solves dynamic programming problems. 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 web crawler simulation that uses Depth First Search (DFS) and Breadth First Search (BFS) algorithms to traverse a given website's pages.	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

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	L-T-P	Credits
Database Management System	**ENCS204**	4-0-0	4
Type of Course:	Major		
Pre-requisite(s), if any: **Nil**			

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 is divided into 4 modules:

a) Introduction

b) Relational Query Languages

c) Transaction Processing and Storage Strategies

d) Advanced Topics and Database 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 explore advanced database concepts such as distributed databases, data warehousing, and data mining.

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: 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: **Database Security**: Authentication, authorization, access control, DAC (Discretionary Access Control), MAC (Mandatory Access Control), RBAC (Role-Based Access Control). **Intrusion Detection**: Techniques and tools, SQL injection prevention. **Advanced Database Topics**: Object-oriented databases, object-relational databases, logical databases, web databases. **Distributed Databases**: Concepts, architecture, data fragmentation, replication, distributed query processing. **Data Warehousing and Data Mining**: Concepts, architecture, OLAP, data preprocessing, data mining techniques.		

Learning Experiences:

- Hands-on Lab Work: Apply DBMS concepts through practical lab assignments with MySQL, PostgreSQL, and MongoDB.
- Interactive Lectures: Use PPTs and interactive boards for engaging lectures; watch video lectures on key topics.
- Problem-Based Assignments: Solve theory assignments focused on database design and query optimization.
- Group Projects: Collaborate on projects to design and implement databases for simulated business cases.
- Case Studies: Analyze industry case studies to understand real-world DBMS applications and challenges.
- Continuous Assessment: Engage in quizzes, assignments, and peer reviews for ongoing evaluation and feedback.

- ICT Tools: Access course materials and resources on Moodle LMS for study and revision.
- Instructor Support: Seek help and feedback from the instructor during office hours and consultations.

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	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	L-T-P	Credits
Database Management System Lab	**ENCS254**	0-0-2	1
Type of Course:	Major		

Defined Course Outcomes

COs	
CO 1	Designing and implementing database schemas using both open-source and commercial DBMS, defining tables, relationships, and integrity constraints
CO 2	Developing and analyze Entity-Relationship diagrams, relational schemas, and enforce normalization techniques to ensure database efficiency and integrity
CO 3	Executing SQL queries for data definition, manipulation, and complex data retrieval, demonstrating proficiency in relational algebra and transaction processing
CO 4	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. 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.	CO2
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

INTRODUCTION TO UI DESIGN

Program Name:	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Introduction to UI Design	**ENSP210**	4-0-0	4
Type of Course:	IDC		

Pre-requisite(s), if any:

Course Perspective. Learning UI design guidelines for different platforms and operating systems. Understanding the principles and fundamentals of UI Design. To be able to learn and get hands on Iconography & typography for interface design. To fundamentals of screen design based on design guidelines and Cross platform screen design. To master with the practical training in UI design for digital screens. The course is divided into 4 modules:

a) Basic elements of UI Design

b) Typography

c) Iconography

d) Introduction to Visual Tools

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

COs	Statements
CO 1	**Identifying** UI design guidelines for different platforms and operating systems.
CO 2	**Explaining** the principles and fundamentals of UI Design
CO 3	**Applying** iconography & typography principles for interface design
CO 4	**Analyzing** the fundamentals of screen design based on design guidelines and cross-platform requirements.
CO 5	**Creating** meaningful interactions through the translation of ideas and concepts into UI

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: Basic elements of UI Design	No. of hours: 6
Content Summary: Introduction to basic elements of visual design – detailed study of color, color wheel, visual hierarchy, legibility and readability, grid, layout		
Unit Number: 2	**Title: Typography**	**No. of hours: 9**
Content Summary: What is typography, Typefaces history and study, Types of fonts - serif and non-serif, Font anatomy, Importance of Typography in modern age UI design, Usage of type for print vs digital, Latest Trends in Typography		
Unit Number: 3	**Title: Iconography**	**Unit Number: 3**
Content Summary: What is iconography, visualization of icons, industry standards and specifications for iconography, designing for various form factors, trends in iconography, User perception about iconography		
Unit Number: 4	**Title: Introduction to Visual Tools**	**Unit Number: 4**
Content Summary: Details of Internet of Things, Augmented reality and virtual reality, ATM, KIOSK		

Learning Experiences:

- Interactive Lectures: Use visual aids and software tools like Adobe XD to introduce design elements like color, grid, and hierarchy. Demonstrate how these principles apply to real-world UI designs.
- Practical Assignments: Assign students to create simple UI layouts, focusing on color schemes, typography, and visual hierarchy. This allows them to practice applying design concepts.
- Typography Workshops: Teach the fundamentals of typography, including font anatomy and trends. Have students explore the impact of typography in digital and print UIs
- Iconography Projects: Students design icons following industry standards, considering user perception and device form factors. This helps them understand visualization and design trends.
- Visual Tool Demonstrations: Introduce tools like IoT, AR/VR, and kiosk interfaces through practical demos. Students create basic prototypes for devices like ATMs and kiosks.

- Instructor Support: Seek help and feedback from the instructor during office hours and consultations.

TextBooks:

1. Graphic Design The New Basics - Ellen Lupton and Jennifer Cole Phillips
2. The Visual Miscellaneous - David Mc Candless
3. The Elements of User Experience" by Jesse James Garrett
4. "Don't Make Me Think" by Steve Krug
5. "About Face: The Essentials of Interaction Design" by Alan Cooper
6. "Designing Interfaces" by Jenifer Tidwell

Additional Readings:

Online Learning Resource:

1. **Coursera - Introduction to UI Design**
    1. This course will provide an understanding of the critical importance of user interface design and key theories and frameworks that underlie the design of most interfaces being used.
    2. Link: Coursera-Introduction to UI Design
2. Udemy
    1. A wide range of courses on UI design, from beginner to advanced levels.
    2. Link: Udemy UI Design Courses
3. **LinkedIn Learning**

    Provides courses on various aspects of UI design, including tools like Sketch, Figma, and Adobe XD.

    Link: LinkedIn Learning UI Design Courses

UI DESIGN LAB

Program Name:	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: UI Design Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP260	0-0-2	1
Type of Course:	IDC		
Pre-requisite(s), if any: Python			

Defined Course Outcomes

COs	
CO 1	**Understand** and Apply the Basic Elements of Visual Design in UI Contexts.
CO 2	**Exploring** and Utilize Various Visual Tools and Technologies in UI Design
CO 3	**Analyzing** and Utilize Typography in Modern UI Design
CO 4	**Evaluating** and Integrate Iconography in User Interfaces

List of Experiments

Ex No	**Experiment Title**	**Mapped CO/COs**
P1	Introduction to Color Theory and Visual Hierarchy	CO1
P2	Creating and Applying Grids and Layouts	CO1
P3	Exploring Typeface History and Font Anatomy	CO3
P4	Designing with Serif and Sans-serif Fonts	CO3
P5	Typography in Print vs Digital Media	CO3
P6	Latest Trends in Typography	CO3
P7	Basics of Iconography	CO4

P8	Visualization and Creation of Icons	CO4
P9	Designing Icons for Various Form Factors	CO4
P10	Industry Standards for Iconography	CO4
P11	Introduction to IoT, AR, and VR in UI Design	CO2
P12	Designing Interfaces for ATMs and Kiosks	CO2
P13	Using Visual Tools for UI Prototyping	CO3
P14	Case Studies of Successful UI Designs	CO1, CO2, CO3, CO4

Software and Development Tools

- Adobe XD
- Sketch
- Figma
- InVision
- Datasets
- Google Fonts dataset
- Icon8 dataset
- FlatIcon dataset

Online Platforms and Communities

- Dribbble
- Behance
- UI Design Daily
- UX Design Community on Reddit
- Virtual Labs and Additional Resources
- Adobe XD Labs
- Figma Community Files
- Sketch Resources

COMMUNICATION & PERSONALITY

DEVELOPMENT

Program Name:	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Life Skills for Professionals - II	**AEC007**	3-0-0	3
Type of Course:	AEC		
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
CO 3	Cultivating essential soft skills required at 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: Personality Improvement	No. of hours: 6
Content Summary: Asking for and giving information, Offering and responding to offers, Requesting and responding to requests, Congratulating people on their success, Asking questions and responding politely, Apologizing and forgiving		
Unit Number: 2	**Title: Ratio & its application**	**No. of hours: 6**
Content Summary: Time & Work, Time & Distance, Train, Boat & Stream, Permutation & combination, Probability		
Unit Number: 3	**Title: Arithmetic**	**No. of hours: 6**
Content Summary: Inequalities, Log, progression, Mensuration, BODMAS		
Unit Number: 4	**Title: Presentation Skills**	**No. of hours: 6**
Content Summary: Presentation Skills, Telephone etiquettes, LinkedIn Profile and professional networking, Video resumes & Mock interview sessions.		
Unit Number: 5	**Title: Leadership skills**	**No. of hours: 6**
Content Summary: Nurturing future leaders, increasing productivity of the workforce, Imparting Self-leadership, Executive leadership.		

Learning Experiences

- Lectures will use PPTs, interactive boards, and video lectures for enhanced engagement.
- Problem-based theory assignments will promote critical thinking and application.
- Project-based lab assignments will allow hands-on, real-world learning.
- Moodle LMS will provide access to course materials, question banks, and model papers.
- Continuous assessments with feedback will track progress and offer personalized guidance.
- Group activities and peer reviews will foster collaboration and teamwork.
- Instructor support will be available for additional feedback and assistance.

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

Additional Readings:

Online Learning Resource:

https://onlinecourses.nptel.ac.in/noc21_hs02/preview

INFORMATION ARCHITECTURE

Program Name:	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Information Architecture	**SEC044**	2-0-0	2
Type of Course:	SEC		
Pre-requisite(s), if any:			

Course Perspective: Understanding Information architecture. Tools and techniques of Information architecture. Hands on using excel as a tool for card sorting. Creating IA for different industries. Learning types and structures and structures of IA.

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

COs	Statements
CO 1	**Defining** the concept of information architecture.
CO 2	**Evaluating** tools and techniques of Information architecture.
CO 3	**Performing** hands-on exercises for card sorting.
CO 4	**Examining** IA for different digital products.
CO 5	**Designing** and learn different types and structures of IA.

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: Creating Task Flows**	**No. of hours: 10**

Content Summary:

Definition, basics to create task flows - Key components of a task flow, including steps, actions, decisions, and outcomes

Techniques for mapping out the steps involved in each task, Task flow analysis, Techniques for analyzing existing task flows to identify inefficiencies and pain points, AS-IS and TO-BE process

Implementing into simple problems, Examining real-world examples of task flow improvements.

Unit Number: 2	**Title: Introduction to Information Architecture**	**No. of hours: 1 0**

Content Summary:

Introduction: Definition, Structure, hierarchy, types of Information architecture

Types of IA structures: Tree structures, flat structures, network structures, Principles and steps of Information Architecture

Steps involved in creating an information architecture, from research to implementation.

Unit Number: 3	**Title: Tools & Techniques of Information Architecture**	**No. of hours: 1 0**

Content Summary:

Introduction: Affinity mapping, Role in organizing information,

Card sorting –Introduction, Types of card sorting: open, closed, hybrid

Techniques of evaluating and methods for testing IA with users to gather feedback, Analysis of Information architecture

Using excel as a tool for card sorting, Activity based.

Unit Number: 4	**Designing Information Architecture for Business Strategy & Exploring Gaps**	**No. of hours:10**

Content Summary:

Designing Information Architecture for enterprise to meet its organizational goals using a tree structure,

Using tree structures to represent complex organizational information, Making the case using the site mapping and content inventory and audit

Techniques for making a compelling case for IA improvements to stakeholders.

Learning Experiences

- Lectures will use PPTs, interactive boards, and video lectures for enhanced engagement.
- Task Flow Assignments: Assign students to map and analyze task flows for simple problems. Have them identify inefficiencies in AS-IS processes and propose improvements in TO-BE processes.
- Information Architecture Workshops: Learn to create IA diagrams, applying steps from research to implementation for digital products.
- Hands-on Card Sorting Activities: Conduct in-class card sorting exercises (open, closed, hybrid) using Excel.
- Group Projects: Organize group projects to design IA for a business strategy, using tree structures and site mapping. Encourage students to explore gaps and propose IA improvements to meet organizational goals.
- Continuous assessments with feedback will track progress and offer personalized guidance.
- Group activities and peer reviews will foster collaboration and teamwork.
- Instructor support will be available for additional feedback and assistance.

TextBooks

1. A Practical Guide to Information Architecture by Donna Spencer
2. The User's Journey: Storymapping Products That People Love by Donna Lichaw
3. User Is Always Right: A Practical Guide to Creating and Using Personas for the Web by Steve Mulder

Additional Readings:

1. **Coursera**

Information Architecture from University of Michigan:

1. This course covers the basics of IA and how to organize and structure content effectively.
2. Link for Information Architecture-Coursera Information Architecture

MINOR PROJECT-II

Program Name:	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: Minor Project-II	**Course Code**	**L-T-P**	**Credits**
	ENSI252	---	2
Type of Course:	Project		
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

- Hands-on Project-Based Learning: Students will work on real-world societal problems, applying their technical skills in data gathering, ETL processes, and innovative problem-solving.
- Critical Thinking through Problem Formulation: Students will critically analyze problems, review literature, and evaluate existing solutions, fostering a deeper understanding of complex societal challenges.
- Collaboration and Peer Support: Group work will encourage peer collaboration, allowing students to exchange ideas and provide feedback to one another, enhancing teamwork and leadership skills.

- Innovation and Entrepreneurship Focus: The project will push students to develop novel ideas and entrepreneurial solutions, promoting creativity and out-of-the-box thinking.
- Continuous Support and Feedback: The course in charge will provide regular feedback and support throughout the project, ensuring students stay on track and improve their work iteratively.

COMPETITIVE CODING -II

Program Name:	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: COMPETITIVE CODING -II	**Course Code**	**L-T-P**	**Credits**
		3-0-0	NIL
Type of Course:	Audit Course		
Pre-requisite(s), if any: Fundamentals of programming & data structure			

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.
CO3	Implementing shortest path algorithms such as Dijkstra's algorithm and Bellman-Ford.
CO4	Exploring dynamic programming concepts, including memorization and tabulation, to solve classic problems.

Unit Number: 1	**Title: Object-Oriented Programming Concepts**	**No. of hours: 8**

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

Unit Number: 2	**Title:** Linked Lists, Stacks and Queues	**No. of hours: 8**

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 - 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:

- Interactive Lectures: Engage students with visual PPTs, encourage questions, and simplify complex concepts.
- Problem-Based Assignments: Assign theory problems and practical lab tasks (e.g., segment trees, shortest path algorithms).
- Continuous Assessment: Regular quizzes, mini-projects, and formative assessments to reinforce learning.
- Peer Collaboration: Group activities, case studies, and peer reviews for diverse perspectives.
- Feedback and Support: Instructors provide timely feedback, and students seek help as needed.
- Life-Long Learning: Foster curiosity, critical thinking, and skills beyond the syllabus.

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, Inheritance, Polymorphism).	CO1
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

Problem Statement	Mapped COs
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
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

Problem Statement	Mapped COs
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
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:	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: Theory of Computation	**Course Code**	**L-T-P**	**Credits**
	ENCS301	3-1-0	4
Type of Course:	Major		
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 Vice-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 Experiences:

- Problem-Solving: Engage in regular exercises designing finite automata and Turing machines.
- Simulations: Use computational tools to simulate automata and visualize operations.
- Critical Thinking: Analyze computation limits, including undecidability and the Halting Problem.
- Mathematical Rigor: Practice theorem proofs, including the Pumping Lemma and Arden's Theorem.
- Computability Case Studies: Explore real-world computational challenges and limits.
- Assessment on Applications: Evaluate through practical conversions and problem-solving tasks.
- Peer Presentations: Present on key topics like the Chomsky hierarchy and DFA/NFA conversions.
- Continuous Feedback: Quizzes and discussions with immediate feedback to track progress.

Text Books:

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

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

III) H.R. Lewis and C.H. Papadimitriou, "Elements of the theory of Computation", Second Edition, Pearson Education.

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

V) 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"

1. **NPTEL - Theory of Computation**
    - Online course by IITs on the fundamentals of the theory of computation.
    - Link: NPTEL-Theory of Computation
2. **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
3. **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:	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: OPERATING SYSTEMS	**Course Code**	**L-T-P**	**Credits**
	ENCS303	4-0-0	4
Type of Course:	Major		
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 managing memory allocation, paging, and virtual memory techniques.

CO 4	**Examining** process synchronization mechanisms and handling deadlocks in an operating system environment.
CO 5	**Developing** distributed operating systems and concurrent systems with a focus on fault tolerance and recovery mechanisms.

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

- Hands-on Simulations: Practice process scheduling and memory management in simulated environments.
- Collaborative Projects: Work in teams to implement CPU scheduling and memory allocation algorithms.
- Problem-Solving: Tackle deadlock prevention and synchronization challenges in labs.
- Interactive Discussions: Engage in class discussions on process synchronization, file systems, and security.
- Application-Based Learning: Apply OS concepts to optimize performance in distributed systems.
- Assessment of Real-World Systems: Evaluate different scheduling and memory management techniques in practical systems.
- Critical Thinking: Solve classical synchronization problems like the producer-consumer and dining philosophers.
- Peer Learning: Share and discuss solutions to OS challenges through group presentations.

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:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name: OPERATING SYSTEMS LAB	**Course Code**	**L-T-P**	**Credits**
	ENCS351	0-0-2	1
Type of Course:	Major		
Pre-requisite(s), if any: Basics of programming			

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 evaluating 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 implementing secure file systems, demonstrating file operations, directory management, and access control mechanisms.

List of Experiments

Ex No	**Experiment Title**	**Mapped CO/COs**
1	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
2	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
3	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

4	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
5	Design and implement solutions for classical synchronization problems such as the Producer-Consumer problem, Readers-Writers problem, and Dining Philosophers problem using semaphores, critical regions, and monitors.	CO2
6	Create a simulation to detect and handle deadlocks in a system. Implement deadlock prevention, avoidance, and detection algorithms. Demonstrate recovery from deadlock scenarios.	CO2
7	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.	CO2
8	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.	CO3
9	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.	CO3
10	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.	CO3
11	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.	CO4
12	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.	CO4

INTRODUCTION TO INTERACTION DESIGN

Program Name:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name: Introduction to Interaction Design	**Course Code**	**L-T-P**	**Credits**
	SEC045	2-0-0	2
Type of Course:	SEC		
Pre-requisite(s), if any:			

Course Perspective: Learning the Importance and scope of Interaction design, User centered design. Design of interactive products Methods of interaction design Tools for interaction design. Get to know futuristic technologies and their implementation in design. The course is divided into 4 modules:

a) Introduction to Interaction Design

b) User Centered Design

c) Design of Interactive products

d) Methods of Interaction Design

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

COs	Statements
CO 1	**Recognizing** the importance and scope of Interaction design, not just for user interfaces, but also for physical products.
CO 2	**Explaining** the relationship between principles of input and feedback.
CO 3	**Demonstrating** proficiency in using relevant software tools and technologies to develop interactive designs.
CO 4	**Critically** evaluating the strengths and weaknesses of different user interfaces based on interaction design.
CO 5	**Applying** the principles of interaction design to create prototypes and wireframes for digital interfaces.

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 Interaction Design**	**No. of hours: 3**
Content Summary: **Interaction in design:** Scope, history of interaction in design, case studies, User Centered Design		

Ergonomics: Physical, cognitive and organizational.

Learning the different methods which includes tools and techniques of interaction design, Understanding micro-interactions.

Differentiating interaction design from related fields like UX design, UI design, and human-computer interaction (HCI).

Unit Number: 2	**Title: User-Centered Design (UCD)**	**No. of hours: 9**

Content Summary:

Introduction: Principles, philosophy Differentiating UCD from other design methodologies.
Process of UCD: Iterative Process, Continuous improvement through user feedback and testing, Case studies showcasing iterative design.
UCD Considers the Whole User Experience - Addressing all aspects of the user's interaction with a product, including emotional and functional experiences. Investment in UCD Pays off, Benefits of UCD and UX, UCD Waterfall process map
UCD process: Detailed diagram with explanation of each stage and its significance.

Unit Number: 3	**Title: Design of Interactive Products**	**No. of hours: 9**

Content Summary:
Ergonomics: Designing for physical comfort and efficiency, Case studies on the impact of ergonomic design on user satisfaction.

Designing for mental processes and cognitive load, Techniques to reduce cognitive strain and enhance usability.

Considering the interaction between systems and organizational structures, Designing systems that fit seamlessly into users' workflows

Unit Number: 4	**Methods of Interaction Design**	**No. of hours: 6**

Tools and Techniques- Overview of tools used in interaction design: wire framing tools

Prototyping software and techniques: Storyboarding, sketching, and user journey mapping.

Understanding Micro interactions - The role of micro-interactions in enhancing user experience and examples of effective micro-interactions in everyday products.

Designing Micro interactions - Principles and best practices for creating engaging micro-interactions with case studies of successful micro-interactions in digital products.

Learning Experiences:

- Problem-Solving: Engage students in exercises differentiating interaction design from UX, UI, and HCI. Have them analyze micro-interactions and their role in user-centered design.
- Simulations: Use tools like wireframing software and prototyping platforms to simulate interaction design workflows, emphasizing physical and cognitive ergonomics.
- Critical Thinking: Assign case studies that examine iterative user-centered design processes, focusing on user feedback, cognitive load, and ergonomic efficiency in product design.
- Hands-on Prototyping: Conduct practical sessions where students use storyboarding, sketching, and journey mapping tools to design interactive products and micro-interactions.
- Peer Presentations: Encourage students to present on key topics like user-centered design methodologies, showcasing examples of successful micro-interactions and iterative design processes.
- Continuous Feedback: Provide regular quizzes, design critiques, and feedback during prototyping activities, focusing on the application of interaction design principles.

Textbooks

1. About Face 3: The Essentials of Interaction Design- by Robert Reimann,Alan Cooper, David Cronin
2. Designing Interactions (The MIT Press)- by Bill Moggridge

DESIGN THINKING

Program Name:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name: Design Thinking	**Course Code**	**L-T-P**	**Credits**
	ENSP315	4-0-0	4
Type of Course:	IDC		
Pre-requisite(s), if any: Introduction to UX Design			

Course Perspective. Get to know what design thinking and wicked problem is. To learn to generate new ideas. To grasp the methods of the design thinking 5d process. To comprehend and effectively use the tools and techniques to solve wicked problems. To apprehend the application of design thinking with case studies. The course is divided into 4 modules:

a) Introduction to Design Thinking

b) Design Philosophies

c) Case Studies in Design Thinking

d) Design Framework

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

COs	Statements
CO 1	Learning to generate new ideas.
CO 2	Defining the methods of the design thinking 5D process.
CO 3	Examining the tools and techniques of solving wicked problems.
CO 4	Designing and applying the method of design thinking with case studies.
CO 5	Evaluating and exploring what design thinking and wicked problems are.

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 Design Thinking**	**No. of hours: 10**

Content Summary:

Introduction: Definition of Design thinking and how it has evolved to solve wicked problems around the world, four pillars of wicked problems, Deep dive into the Design process followed by Designers around the world

Complexity: Understanding the multifaceted nature of wicked problems. Uncertainty: Navigating the unknowns and unpredictability's.

Conflict: Managing differing perspectives and interests.

Constraints: Working within limitations and restrictions, Differentiate between design thinking and traditional problem-solving methods.

Unit Number: 2	**Title: Case studies in Design thinking**	**No. of hours: 10**

Content Summary:

Getting to know the real-world applications and success stories of different industries, Understanding the value of real-world examples in learning design thinking,

Healthcare: How design thinking is used to improve patient experiences and healthcare delivery.

Technology: Innovative solutions in tech, including user-centered software development.

Education: Designing better learning experiences and environments. Social Innovation: Addressing social issues through design thinking.

Unit Number: 3	**Title: Design Frameworks**	**No. of hours: 1 0**

Content Summary:

AARRR framework - What is the AARRR framework and its components (Acquisition, Activation, Retention, Referral, Revenue), How the AARRR framework is used in product and service design

Customer Experience Index (CX Index) -Definition and importance of the Customer Experience Index. How to measure and improve customer experience using the CX Index

Google's HEART framework, Social Impact Metrics, IDEO, Standford, ImaginXP 5D Process Explained with case studies		
Unit Number: 4	**Innovation & Creativity**	**No. of hours: 10**
Content Summary: Introduction: Innovation, Creativity, Difference between innovation and creativity Dynamics of creative thinking, becoming creatively fit as an individual, creative insight, idea generation, learn what is innovation and how leading organization across the world are implementing innovation, Role of creativity and innovation in organizations, idea evaluation, creativity in teams		

Learning Experiences:

- Problem-Solving: Engage students in exercises to differentiate design thinking from traditional problem-solving, focusing on navigating wicked problems through the design process.
- Case Study Analysis: Explore real-world design thinking applications in healthcare, technology, and social innovation. Students will analyze how design thinking transforms industries.
- Framework Application: Assign hands-on activities where students apply frameworks like AARRR, HEART, and CX Index to design products and improve customer experience.
- Creative Workshops: Conduct sessions focused on idea generation, creative insights, and evaluating innovations. Students will practice brainstorming and refining innovative solutions.
- Peer Presentations: Have students present on case studies and design frameworks, such as IDEO and Stanford's 5D process, demonstrating practical design thinking applications.
- Continuous Feedback: Provide ongoing assessments through quizzes, case study discussions, and feedback on creative and innovative project work, ensuring students grasp key concepts**.**

Textbooks

1. Designing for Digital Age: How to create human-centered products and services - Kim Goodwin
2. The design of everyday things - Don Norman
3. Gamestorming: A Playbook for Innovators, Rulebreakers, and Changemakers- by Dave Gray, Sunni Brown, James Macanufo
4. Sprint- Jake Knapp
5. Change by Design - Tim Brown

Additional Readings:

1. **Virtual Lab on Prototyping Tools:** Explore an online platform like InVision or Marvel App for creating and testing interactive product prototypes, providing a hands-on experience in UI/UX design.

2. **Customer Journey Mapping Tool:** Utilize an online tool such as Lucidchart or Microsoft Visio to create detailed customer journey maps, helping to visualize the customer experience and identify key interaction points.

3. **ROI Calculator Online:** Access an interactive Return on Investment (ROI) calculator that allows students to input data and see the potential financial impact of product innovations and decisions.

4. **Virtual Workshop on Stakeholder Analysis:** Engage with a virtual workshop or webinar focused on stakeholder analysis techniques, providing insights into managing and understanding stakeholder expectations in product development.

DESIGN THINKING LAB

Program Name:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name: Design Thinking Lab	**Course Code**	**L-T-P**	**Credits**
	ENSP365	0-0-2	1
Type of Course:	IDC		
Pre-requisite(s), if any:			

Defined Course Outcomes

COs	
CO1	**Demonstrating** knowledge of Metaverse Technologies and Tools
CO2	**Applying** technical Skills in Virtual and Augmented Reality Development
CO3	**Analyzing** and Evaluating Metaverse Applications
CO4	**Designing** and **Creating** Immersive Experiences in the Metaverse

List of Experiments

Ex. No	**Experiment Title**	**Mapped CO/COs**
1	Create a value proposition canvas for a new product idea, focusing on customer needs and product benefits.	CO1, CO3
2	Conduct a feature mapping session for an existing product to enhance its design or functionality.	CO1, CO3
3	Develop an ROI (Return on Investment) map for a proposed innovation to evaluate its potential financial benefits.	CO1, CO3

4	Analyze a competitor's product using design thinking tools to identify potential areas for improvement in your own product.	CO2, CO3, CO4
5	Map the customer journey for a service or product, identifying key touchpoints and opportunities for enhancement.	CO2, CO3
6	Identify and map key stakeholders for a project and analyze their influence and interests related to the product.	CO3
7	Design and prototype a basic wearable device that incorporates user experience (UX) elements.	CO1, CO3, CO4
8	Conduct usability testing on a series of prototypes to gather user feedback and identify areas for improvement.	CO1, CO3, CO5
9	Modify an existing product to enhance its accessibility for users with disabilities.	CO1, CO3
10	Facilitate an ideation session to generate new product ideas using design thinking methodologies.	CO1, CO2
11	Create detailed user personas for a new technology product targeted at varying market segments.	CO1, CO3
12	Integrate design thinking tools into developing a business model canvas for a startup idea.	CO1, CO2, CO3
13	Design and develop a strategic product design plan for a prototype, including market analysis and product positioning.	CO1, CO3, CO4
14	Simulate the process of product lockdown, determining final features and design specifications under constraints.	CO2, CO3, CO4
15	Create comprehensive UI design documentation for a software application.	CO3, CO4
16	Prepare detailed design delivery documentation to accompany a digital product's release.	CO3, CO4
17	Analyze a successful product in the market and understand how design thinking contributed to its development and success.	CO2, CO4, CO5
18	Hold a design critique session where students present their designs and receive feedback based on design thinking principles.	CO1, CO2, CO3
19	Engage in an iterative design process where initial designs are continuously refined based on user feedback and new insights.	CO1, CO3

20	Organize a cross-functional team exercise to develop a product concept that includes input from design, engineering, and marketing.	CO1, CO3, CO4
21	Assess a product design for sustainability and propose design modifications to enhance its environmental friendliness.	CO1, CO3
22	Develop a quick prototype for a digital application focusing on user experience design principles.	CO1, CO3, CO4
23	Craft a marketing strategy for a new product using design thinking to identify unique selling points and market needs.	CO1, CO2, CO3
24	Organize a showcase where students present their strategic product designs and prototypes to an audience of peers and industry professionals.	CO3, CO4
25	Design an experiment to establish a feedback loop for a product in the beta phase to gather continuous user insights.	CO1, CO3, CO5

ARITHMETIC AND REASONING SKILLS

Program Name:	**B.Tech (CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Life Skills for Professionals -III	**AEC008**	3-0-0	3
Type of Course:	AEC		
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:

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 Summary: Vedic Maths, Classification of Numbers and Divisibility Rule, Percentage, Ratio and Proportion		
Unit Number: 2	**Title: Fundamentals of Logical Reasoning**	**No. of hours: 6**
Content Summary: Blood Relations, Direction Sense, Coding Decoding		
Unit Number: 3	**Title: Elementary Quantitative Skills**	**No. of hours: 18**
Content Summary: Simple and Compound Interest, Average, Partnership, Time and Work, Time Speed & Distance		
Unit Number: 4	**Title: Advanced Quantitative Skills**	**No. of hours: 6**
Content Summary: Permutation & Combination, Probability		

Learning Experiences

- Sharpen Logic: Solve reasoning puzzles to enhance problem-solving skills.
- Improve Non-Verbal Reasoning: Tackle visual puzzles to boost analytical thinking.
- Build Employability Skills: Practice resume writing, interviews, and group discussions.
- Enhance Critical Thinking: Apply logic to evaluate situations and make decisions.
- Develop Teamwork: Work in groups to improve collaboration and resolve conflicts.
- Strengthen Leadership: Practice leadership and effective communication.
- Boost Digital Literacy: Use online tools relevant to the workplace.

Text Book References:

R1. Guha Abhijit: Quantitative Aptitude for Competitive Examinations, Tata McGraw Hill Publication

R2. Quantitative Aptitude by R.S. Aggarwal

R3. Verbal & Non-Verbal Reasoning by R.S. Aggarwal

Additional Readings:

https://www.indiabix.com/online-test/aptitude-test/

https://www.geeksforgeeks.org/aptitude-questions-and-answers/

https://www.hitbullseye.com/

Summer Internship-II

Program Name:	B.Tech(CSE) with specialization in UI/UX		
Course Name: Summer Internship-II	Course Code	L-T-P	Credits
	ENSI351	0-0-0	2
Type of Course:	INT		
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:

- Real-World Application: Students apply theoretical concepts learned in their coursework to practical problems in industry or research settings, providing a hands-on approach to learning.
- Technical Skill Development: Through their internships, students acquire and demonstrate new technical skills relevant to computer science and engineering, enhancing their practical knowledge and expertise.
- Independent Research Projects: Students undertake research projects, which include problem identification, literature review, methodology design, and data analysis, fostering independent research skills.

- Professional Report Preparation: Students compile detailed reports documenting their internship experiences, project outcomes, and research findings, emphasizing report structure, clarity, and comprehensiveness.
- Presentation Skills: Students prepare and deliver presentations on their internship projects, focusing on clear communication, effective use of visual aids, and engaging delivery.
- Professional and Soft Skills Enhancement: Students develop key professional skills such as teamwork, time management, and adaptability, as well as soft skills like communication and problem-solving.
- Continuous Feedback and Support: Regular feedback sessions are held to guide students, address challenges, and support their learning throughout the internship, ensuring a continuous improvement process.
- Peer Collaboration: Students collaborate with peers to exchange insights, provide feedback on each other's work, and support one another in their internship projects and presentations.
- Industry Exposure: Students gain practical experience and insights into industry practices, preparing them for future career opportunities and enhancing their understanding of professional environments.

- Commitment to Lifelong Learning: Students are encouraged to engage in continuous skill development and stay updated with emerging trends and technologies in computer science and engineering, promoting lifelong learning.

USABILITY TESTING

Program Name:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name: **Usability Testing**	**Course Code**	**L-T-P**	**Credits**
	ENSP317	4-0-0	4
Type of Course:	IDC		
Pre-requisite(s), if any:			

Course Perspective. Students will be able to Design vector artwork. Able to prepare graphics for web and print. To implement useful keyboard shortcuts. Learn illustrator the way a professional would use it. Practice everything you learn during the course.

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

CO1	**Defining** and analyze and critique the design of interactive products

CO2	**Recognizing** the concept of micro-interactions in detail products
CO3	**Analyzing** findings from a usability test
CO4	**Implementing** various tools and techniques of Usability testing
CO5	**Designing** and executing the process to conduct end-to-end usability testing on real life digital products

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: Process of Usability Testing**	**No. of hours: 10**
Content Summary: Define Usability testing, Types of testing, Learning the steps to test different types of products/service/methods planning, executing, information gathering and documentation, case studies, heuristic evaluation and its importance, 10 Laws of Heuristic Evaluation, Understanding Heuristics through practical examples		
Unit Number: 2	**Title: Usability testing for Digital products**	**No. of hours: 10**
Content Summary: Learn how to create questionnaires, test cases and test moderation. Preparing for the testing of products, understanding people's psychology and Behaviour, How to create questionnaires that gather useful insights. Types of questions (e.g., open-ended, closed-ended) and their uses, Techniques for moderating usability tests		
Unit Number: 3	**Title: Tools & Techniques of Usability Testing**	**No. of hours: 10**
Content Summary: Usability testing methodologies – task-based user testing, A/B testing		

Designing and analyzing A/B tests, lab-based user testing, remote user testing, moderated & un-moderated user testing. Tools: Google Optimizer, Hotjar, Using Hotjar for heatmaps, session recordings, and user feedback.		
Unit Number: 4	**Analyzing and Reporting Usability Testing**	**No. of hours: 3**
Content Summary: Techniques for analyzing qualitative data from usability tests, Identifying themes and patterns in user feedback, Structure and components of a usability test report Techniques for visualizing usability test data, Using charts, graphs, and other visuals to convey findings, How to effectively communicate usability test results to stakeholders. Tailoring communication to different audiences (e.g., designers, developers, executives).		

Textbooks:

- The Adobe Photoshop CC Book for Digital Photographers - Scott Kelby
- Adobe Illustrator CC Classroom in a Book (2017 release) - Brian Wood

References

- Measuring the User Experience: Collecting, Analyzing, and Presenting Usability Metrics (William Albert and
- Thomas Tullis)
- Practical Guide to Usability Testing - Author: Joseph S. Dumas
- Usability Inspection Methods - Authors: Jakob Nielsen
- Usability Engineering - Author: Jakob Nielsen
- Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests - Author: Jared Spool, Jeffrey Rubin, Dana Chisnell

COMPETITIVE CODING -III

Program Name:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
COMPETITIVE CODING -III		3-0-0	NIL
Type of Course:	Audit Course		
Pre-requisite(s), if any: Data Structure			

Course Outcomes

CO1	Analyzing and writing SQL queries to retrieve, modify, and optimize data in relational databases.
CO2	Designing and 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	Implementing and evaluating graph algorithms for traversing, searching, and finding shortest paths in complex graph structures.

Unit Number: 1	**Title: SQL & PL/SQL**	**No. of hours: 8**

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

Unit Number: 2	**Title:** Height Balanced Tree Concepts	**No. of hours: 8**

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)

Unit Number: 3	**Title: Greedy Design Strategy and Dynamic Programming**	**No. of hours: 8**

Content:

Greedy Algorithms: Definition and Characteristics, Greedy Choice 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	

Problem Statement	**Mapped COs**
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
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

Problem Statement	Mapped COs
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 after each deletion.	CO6
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

Problem Statement	**Mapped COs**
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
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:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name: **Computer Organization & Architecture**	**Course Code**	**L-T-P**	**Credits**
	ENCS302	3-1-0	4
Type of Course:	Major		
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 basic concepts of computer architecture and data representation.
CO 2	**Applying** memory hierarchy and caching techniques in computing systems.
CO 3	**Analyzing** processor performance and optimization strategies.
CO 4	**Evaluating** input/output systems and storage technologies for improved system

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 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**.**		

Text Books:

I) David A. Patterson and John L. Hennessy ,"Computer Organization and Design: The Hardware/Software Interface",5th Edition, Elsevier

II) Mano M. Morris, "Computer System Architecture", Pearson.

III) CarlHamache, "Computer Organization and Embedded Systems", 6th Edition, McGraw Hill Higher Education

IV) "Computer Architecture and Organization", 3rd Edition by John P. Hayes, WCB/McGraw-Hill

V) William Stallings "Computer Organization and Architecture: Designing for Performance", 10th Edition, Pearson Education

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:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name: **Computer Networks**	**Course Code**	**L-T-P**	**Credits**
	ENCS304	4-0-0	4
Type of Course:	Major		
Pre-requisite(s), if any: Basics of Communication Engineering and Data structures			

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 of computer networks, including data communication components, network topologies, and the OSI model.
CO 2	**Applying** data link layer protocols and techniques for error detection, error correction, and flow control in practical networking scenarios.
CO 3	**Analyzing** network layer functions, including logical addressing (IPv4, IPv6), address mapping, and routing protocols, to design and troubleshoot network architectures.

CO 4	**Evaluating** and **implementing** application layer protocols such as DNS, HTTP, and FTP, and assess their performance and security in network environments.

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: 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

- Interactive Lectures: Utilize lecture PPTs and interactive teaching boards to present core concepts and facilitate real-time discussions on data communication, network layers, and protocols.
- Hands-on Labs: Engage in project-based lab assignments that involve configuring network devices, implementing protocols, and analyzing network traffic to gain practical experience.
- Problem-Based Assignments: Complete problem-based theory assignments that challenge students to apply network layer and data link layer concepts to solve real-world networking issues.
- Collaborative Group Work: Participate in group activities and peer reviews to design and evaluate network architectures, fostering teamwork and collaborative problem-solving skills.
- ICT Tools: Access course materials, submit assignments, and engage in online discussions through Moodle LMS, ensuring seamless communication and resource sharing.
- Continuous Assessment: Regularly assess understanding through quizzes, assignments, and model question papers, with timely feedback provided to track progress and guide improvement.
- Video Lectures: Watch video lectures on complex topics such as QoS and advanced network protocols to reinforce learning and clarify difficult concepts.
- Support and Feedback: Receive additional support and feedback from the course instructor, with encouragement to seek help as needed to ensure a comprehensive learning experience.

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:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Computer Networks Lab	**ENCS352**	0-0-2	1
Type of Course:	Major		
Pre-requisite(s), if any:			

Defined Course Outcomes

COs	
CO 1	**Applying** fundamental networking concepts and techniques to develop and analyze network topologies, protocols, and error detection mechanisms.
CO 2	**Designing** and implementing network protocols and architectures for efficient data communication and management in various environments.
CO 3	**Utilizing** advanced networking techniques to implement, monitor, and optimize communication systems for real-time and multimedia applications.
CO 4	**Integrate** IoT devices and develop smart systems using networking principles for automation and efficient data management.

List of Experiments

Ex. No	**Experiment Title**	**Mapped CO/COs**
1	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.	CO 1
2	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.	CO 1
3	Develop a network simulator to analyze packet delay, loss, and end-to-end throughput. Implement various routing algorithms and measure their impact on network performance under different traffic conditions.	CO1

4	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
5	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
6	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
7	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
8	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
9	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
10	Implement a DNS and DDNS simulation to demonstrate domain name resolution and dynamic updates. Create a simple client-server application that queries and updates the DNS records.	CO4
11	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

Web Development with HTML/CSS/Java Script/Python

Program Name:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name: **Web Development with HTML/CSS/Java Script/Python**	**Course Code**	**L-T-P**	**Credits**
	ENCS358	4-0-0	4
Type of Course:	Major		
Pre-requisite(s), if any: Python			

Course Perspective. This course is designed to provide students with a comprehensive understanding of web development using HTML, CSS, JavaScript, and Python. It aims to equip students with the skills necessary to build and maintain modern web applications, focusing on both front-end and back-end development. By the end of the course, students will be able to create dynamic, responsive, and interactive websites, as well as manage server-side operations using Python.

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

COs	Statements
CO 1	Understanding and apply the fundamentals of HTML and CSS to create structured and styled web pages.
CO 2	Utilizing JavaScript to add interactivity and dynamic content to web pages.
CO 3	Developing and integrating server-side applications using Python and relevant frameworks.
CO 4	Implementing responsive design principles to ensure websites are accessible on various devices and screen sizes.
CO 5	Combining front-end and back-end technologies to build full-stack web 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 HTML and CSS	**No. of hours: 10**
Content: Basics of HTML: Structure, elements, attributes, HTML5 semantic elements, Forms and input validation Introduction to CSS: Syntax, selectors, and properties, CSS Box Model, positioning, and layout techniques Responsive design with media queries, CSS frameworks (e.g., Bootstrap)		
Unit Number: 2	**Title:** JavaScript for Web Development	**No. of hours: 10**
Content: JavaScript basics: Syntax, variables, and operators Control structures: Conditionals and loops, Functions and scope DOM manipulation and events, AJAX and asynchronous programming JavaScript libraries (e.g., jQuery) and frameworks (e.g., React)		
Unit Number: 3	**Title:** Server-side Development with Python	**No. of hours: 10**
Content: Introduction to server-side programming, Setting up a Python development environment Basics of Python: Syntax, data types, and control flow Building web applications with Flask/Django Working with databases (SQL and NoSQL),RESTful API development User authentication and security practices		
Unit Number: 4	**Title:** Full-stack Web Development	**No. of hours: 10**
Content: Integrating front-end and back-end components, Handling form data and user input Session management and state maintenance, Implementing real-time communication (e.g., WebSockets)		

Deploying web applications to cloud platforms, Performance optimization and debugging techniques Case studies and project work

Textbooks:

HTML and CSS: Design and Build Websites" by Jon Duckett

Reference Books:

R 1. "JavaScript: The Good Parts" by Douglas Crockford

R 2. "Eloquent JavaScript: A Modern Introduction to Programming" by Marijn Haverbeke

R 3. "Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics" by Jennifer Niederst Robbins

R 4. "Flask Web Development: Developing Web Applications with Python" by Miguel Grinberg

R 5. "Python Crash Course: A Hands-On, Project-Based Introduction to Programming" by Eric Matthes

R 6. "JavaScript and JQuery: Interactive Front-End Web Development" by Jon Duck

WEB DEVELOPMENT LAB

Program Name:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name: **Web Development Lab**	**Course Code**	**L-T-P**	**Credits**
	ENCS358	4-0-0	4
Type of Course:	Major		
Pre-requisite(s), if any: Python			

Defined Course Outcomes

COs	
CO 1	**Developing** and designing web pages using HTML and CSS.
CO 2	**Implementing** interactivity and dynamic behavior in web pages using JavaScript.
CO 3	**Creating** server-side applications and handle databases using Python.
CO 4	**Integrating** front-end and back-end components to build full-stack web applications.

Lab Experiments

S.N	Lab task	Mapped CO/COs
1	Project 1: Personal Portfolio Website • Problem Statement: Develop a personal portfolio website using HTML, CSS, and basic JavaScript. The website should showcase the user's skills, projects, and contact information. Implement basic web page structure and styling to create a professional-looking portfolio. Project 2: Basic Company Landing Page • Problem Statement: Create a landing page for a fictitious company using HTML and CSS. The landing page should include sections such as an introduction, services, testimonials, and contact information. Ensure proper use of HTML tags and CSS for layout and design.	CO1
P2	**Project 3: Interactive Quiz Application** Problem Statement: Develop an interactive quiz application where users can answer multiple-choice questions. Use JavaScript to handle the quiz logic, including scoring and displaying the results. Implement features like timer, question navigation, and real-time feedback. **Project 4: Dynamic To-Do List** Problem Statement: Create a dynamic to-do list application where users can add, edit, and delete tasks. Use JavaScript to update the list in real-time without refreshing the page. Implement features like task prioritization, search, and filtering.	CO2
3	**Project 5: Simple Blog Platform** **Problem Statement**: Develop a simple blog platform using Flask where users can create, read, update, and delete blog posts. Implement user authentication to manage access to the platform. Use a SQLite database to store blog posts and user data.	CO3

	Project 6: Contact Management System **Problem Statement**: Create a contact management system using Flask where users can store and manage their contacts. Implement CRUD operations (Create, Read, Update, Delete) for contacts and ensure data is stored in a SQLite database. Include user authentication for secure access.	
4	**Project 7: E-Commerce Website** **Problem Statement**: Develop an e-commerce website where users can browse products, add items to a shopping cart, and complete purchases. Use Flask for the backend to manage product data, user accounts, and orders, and JavaScript (along with a frontend framework like React or Vue) to create a dynamic and responsive user interface.	CO4

VISUAL DESIGN TOOLS LAB

Program Name:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name: **Visual Design Tools Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP368	0-0-4	2
Type of Course:	Minor		
Pre-requisite(s), if any: UI Desihn			

Course Outcomes

CO1	Understanding the basics of visual design principles.
CO2	Applying visual design tools for web development.
CO3	Creating visually appealing web pages.
CO4	Evaluating and improving web design based on user feedback and usability principles.

List of Experiments

S. No.	**Experiment**	**Mapped CO/COs**
1	Introduction to Visual Design Tools	CO 1
2	Working with Color Theory	CO 1, CO 2
3	Typography in Web Design	CO 2
4	Layout Design Principles	CO 1, CO 3
5	Creating Responsive Designs	CO 3
6	Using Grid Systems	CO 2, CO 3
7	Image Editing and Optimization	CO 2

8	Designing with CSS	CO 2, CO 3
9	Introduction to Adobe XD	CO 2
10	Prototyping with Figma	CO 2, CO 3
11	User Interface Design	CO 1, CO 3
12	Creating Interactive Prototypes	CO 3, CO 4
13	Design Systems and Components	CO 2
14	Web Accessibility	CO 3, CO 4
15	Usability Testing	CO 4
16	User Experience (UX) Design Principles	CO 1, CO 4
17	Advanced CSS Techniques	CO 2, CO 3
18	Animations and Transitions in Web Design	CO 2, CO 3
19	Creating SVG Graphics	CO 2
20	Introduction to JavaScript for Designers	CO 2, CO 3
21	Building a Design Portfolio	CO 1, CO 4
22	Case Study: Successful Web Designs	CO 1, CO 4
23	Real-world Design Project 1	CO 3, CO 4
24	Real-world Design Project 2	CO 3, CO 4
25	Final Design Review and Presentation	CO 1, CO 4
P1	Redesigning a Local Business Website	CO 3, CO 4
P2	Developing a Personal Portfolio Site	CO 1, CO 3
P3	Creating an Interactive Web Application	CO 2, CO 3
P4	Designing a Non-profit Organization Site	CO 3, CO 4
P5	Building a Community Forum Platform	CO 3, CO 4

Reference Books

1. " "Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability" by Steve Krug
2. "Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Guidelines" by Jeff Johnson
3. "Responsive Web Design with HTML5 and CSS" by Ben Frain
4. "Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics" by Jennifer Niederst Robbins

(DISCIPLINE SPECIFIC ELECTIVE -I)

WIREFRAMING & PROTOTYPING

Program Name:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Wireframing & Prototyping	**ENSP318**	4-0-0	4
Type of Course:	**DSE**		
Pre-requisite(s), if any:			

Course Perspective. Students will practice learning the tools required to design wireframes and prototypes. Design wireframes on paper and translate paper concepts into digital wireframes. Understand and practice the techniques involved in designing digital wireframes for UI Platforms. Understand and practice the techniques involved in designing digital wireframes for HMI and other digital screens. Understand and practice the techniques involved in creating digital prototypes. Tools to be taught – AxureRP, invision. The course is divided into 4 modules:

a) Basic guidelines of Wireframing

b) Designing wireframes on paper

c) Designing wireframes on Axure/Invision

d) Designing digital wireframes for different UI platforms

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

COs	Statements
CO1	Defining the concepts of Innovation and Creativity
CO2	Recognizing the of types of innovation, role of innovators and innovation settings in organizations

CO3	Evaluating and implementing innovation management tool in different industries
CO4	Designing and formulating researchstudy for executing innovation management to diversified businesses for enhancing user experience.

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: Basic guidelines of Wireframing and Prototyping**	**No. of hours: 12**
Content Summary: Introduction to wireframes, understanding responsive design, primary, secondary and utility navigation, content, inline links, indexes, search, Understanding responsive design principles. How to design wireframes that adapt to different screen sizes, what is Prototyping, when do we need it, understanding rapid prototyping, Types of Prototypes, Overview of wireframing and prototyping digital tools.		
Unit Number: 2	**Title: Designing wireframes on paper**	**No. of hours: 12**
Content Summary: Header, footer, sidebar, navigation systems, use of whitespace, How to use whitespace to improve readability and focus, web fonts and typography, Designing wireframes with different typographic styles, Create paper-based wireframes for a given scenario, Designing sidebars for better navigation, placement and usability considerations		
Unit Number: 3	**Title: Designing wireframes and prototypes on Figma and AdobeXD**	**No. of hours: 12**
Content Summary: Creating visual mockups, What are visual mockups and their importance in the design process? Creating visual mockups in Figma and AdobeXD, Techniques for designing interfaces that require scrolling. Best practices for vertical and horizontal scrolling, whitespace to style a form, scrolling, introduction to clickable prototypes using Figma and AdobeXD, importing and exporting assets.		

Unit Number: 4	Designing digital wireframes for different UI platforms	No. of hours: 12
Content Summary: Practical hands-on demonstration of paper-based wireframes and clickable prototypes using digital tools, Techniques for maintaining design consistency across different platforms, Creating responsive prototypes that adapt to different devices, Live demonstration of creating digital wireframes and prototypes, Hands-on practice sessions to reinforce learning.		

Text Books:

1. Communicating Design: Developing Web Site Documentation for Design and Planning, Book by Daniel M. Brown
2. UI/UX Sketchbook For Wireframing And Prototyping: Big Size Edition Light Version

References

1. Mobile UI/UX Sketchbook: Wireframing and Prototyping Notebook for UI/UX Designers, Students, Mobile
2. App Developers, and Hobbyists App Developer Notebooks

WIREFRAMING & PROTOTYPING LAB

Program Name:	B. Tech (Computer Science and Engineering)		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Wireframing & Prototyping Lab	**ENSP370**	0-0-2	1
Type of Course:	(Discipline Specific Elective -I)		
Pre-requisite(s), if any: Strong programming skills, particularly in Python.			

Proposed Lab Experiments

Defined Course Outcomes

COs	
CO1	Define the concepts of Innovation and Creativity
CO2	Recognize the of types of innovation, role of innovators and innovation settings in organizations, Innovation management and its 4 pillars
CO3	Evaluate and implement innovation as a culture Innovation management tool in different industries
CO4	Design and formulate research study for executing innovation management to diversified businesses for enhancing user experience.

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	Sketch Basic Wireframe: Students will manually sketch a simple webpage layout including essential sections such as headers, footers, and a content area to understand basic wireframe structures.	CO1, CO5
2	Responsive Wireframe Creation: Create a wireframe that adjusts its layout from desktop to mobile view, focusing on responsive design principles to enhance user interface adaptability.	CO1, CO4
3	Navigation Wireframe: Design detailed wireframes focusing on primary, secondary, and utility navigation systems, enhancing site navigability and user experience.	CO1, CO5
4	Content Layout Experiment: Develop wireframes that efficiently organize various types of content to improve readability and engagement.	CO1, CO5
5	Typography in Wireframes: Experiment with different web fonts and typography in wireframe designs to understand visual impact and readability.	CO1, CO5
6	Use of Whitespace: Explore the strategic use of whitespace in wireframes to enhance visual clarity and user experience.	CO1, CO5
7	Inline Links and Indexes: Create wireframes that include inline links and index structures, focusing on enhancing the usability and accessibility of the content.	CO1, CO5
8	Search Functionality Wireframe: Design a wireframe that includes an effective and easy-to-use search bar, enhancing user interaction and functionality.	CO1, CO5
9	Clickable Prototype on Paper: Construct a paper wireframe that includes simulated clickable areas to understand interactive design elements before digital implementation.	CO1, CO5
10	Axure Basics: Learn and apply basic functionalities of Axure to create digital wireframes, focusing on tool proficiency and digital design skills.	CO2, CO5
11	Invision Experimentation: Use Invision to convert static wireframes into interactive prototypes, understanding the process of enhancing user engagement through interactivity.	CO2, CO5

12	Form Styling with Whitespace: Design a form in Axure or Invision, using whitespace creatively to style and structure the form for improved user experience.	CO1, CO5
13	Importing Assets: Practice importing external assets into Axure and Invision, learning how to enhance wireframes with external visuals and data.	CO2, CO5
14	Creating Hotspots: Develop skills in creating interactive hotspots within digital wireframes using Axure or Invision, enhancing interactive prototype functionality.	CO2, CO5
15	Mobile App Wireframe: Design a wireframe specifically for mobile applications, focusing on touch interactions and mobile-specific design principles.	CO1, CO5
16	Web Wireframe Detailed Design: Create a detailed wireframe for a complex website, emphasizing thorough planning and user-centric design strategies.	CO1, CO5
17	Wireframe for iOS: Craft wireframes adhering to iOS design guidelines, focusing on platform-specific conventions and user expectations.	CO1, CO5
18	Wireframe for Wearables: Design wireframes for wearable technology, considering unique challenges such as small screen sizes and context-specific usability.	CO1, CO5
19	Prototype Evaluation and Feedback: Conduct a session for peer reviews of clickable prototypes to gather feedback and refine design approaches based on user input.	CO4, CO5
20	Typography and Readability Test: Test different typography settings in wireframes to determine their impact on readability and user experience, guiding better design decisions.	CO1, CO5
21	Comparative Study of Wireframing Tools: Compare functionalities and usability of various wireframing tools like Balsamiq, Sketch, and Adobe XD, fostering an understanding of tool selection based on project needs.	CO2, CO4
22	Cross-Platform Wireframe Consistency: Design and evaluate wireframes across different platforms to ensure consistent user experience and interface adaptability.	CO1, CO5
23	Interactive Elements Integration: Integrate interactive elements such as drop-down menus and modal windows in wireframes to understand dynamic content handling.	CO1, CO5

24	Header and Footer Variations: Explore various design approaches for headers and footers in wireframes to understand their impact on user navigation and site aesthetics.	CO1, CO5
25	Sidebar Design and Functionality: Develop multiple wireframe variants featuring different sidebar designs to evaluate their effectiveness and usability in web layouts.	CO1, CO5

METAVERSE & ITS APPLICATIONS

Program Name:	**B.Tech(CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
METAVERSE & ITS APPLICATIONS	**ENCS320**	4-0-0	4
Type of Course:	(Discipline Specific Elective -I)		
Pre-requisite(s), if any:			

Course Perspective: Metaverse is an online learning platform designed to help students learn cutting-edge technologies and develop web-enabled technology skills such as virtual reality, voice recognition, artificial intelligence, and more. It is an open-source learning platform created by the Silicon Valley-based company, Metaverse Technologies. Metaverse courses provide students with an introduction to various emerging technologies and skills. Students can learn about various topics such as AI machine learning, virtual reality development, and augmented reality development. Through the use of Metaverse courses, students can gain in-depth knowledge about the various technologies and their applications. The course is divided into 4 modules:

a) Introduction to Mataverse

b) Technologies and Frameworks in the Metaverse

c) Applications of the Metaverse

d) Building for the Metaverse

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

CO1	**Understanding** the concept, evolution, and key features of the Metaverse
CO2	**Identifying** the technologies and frameworks used in the Metaverse
CO3	**Analyzing** and assessing the applications of the Metaverse in various domains
CO4	**Evaluating** the role and impact of different technologies in the Metaverse.
CO5	**Designing** and creating Metaverse applications by applying appropriate design principles

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 Metaverse**	**No. of hours: 4**

Content Summary: Overview of the Metaverse concept, Evolution and history of the Metaverse, Characteristics and key features of the Metaverse, Comparison between Virtual Reality (VR), Augmented Reality (AR), and the Metaverse, The Metaverse vs. Web 3.0, Advantages and Challenges of the Metaverse, ,Opportunities in the development of Mataverse.		
Unit Number: 2	**Title: Technologies and Frameworks in the Metaverse**	**No. of hours: 8**
Content Summary: Types of the Metaverse, Devices to access the metaverse, Extended reality (XR) in the metaverse, Virtual Reality (VR) technologies and devices, Augmented Reality (AR) technologies and devices, Blockchain technology and its role in the Metaverse, Cryptocurrencies and digital assets in the Metaverse, Artificial Intelligence (AI) and Machine Learning (ML) in the Metaverse,		
Unit Number: 3	**Title: Applications of the Metaverse**	**No. of hours: 8**
Content Summary: Social interactions and virtual communities in the Metaverse, Gaming and entertainment in the Metaverse, Education and training in the Metaverse, Healthcare and telemedicine in the Metaverse, Business and commerce in the Metaverse, Art, creativity, and digital ownership in the Metaverse.		
Unit Number: 4	**Title: Building for the Metaverse**	**No. of hours: 8**
Content Summary: Design principles for Metaverse applications, User experience (UX) and user interface (UI) considerations in the Metaverse, Development frameworks and tools for creating Metaverse applications, Security and privacy considerations in the Metaverse, Future trends and possibilities in the Metaverse.		

Text Books:

T1: "The Fourth Transformation: How Augmented Reality & Artificial Intelligence Will Change Everything" by Robert Scoble and Shel Israel

References:

- R1: "Virtual Reality and Augmented Reality: Myths and Realities" by Mel Slater, Anthony Steed, and Yiorgos Chrysanthou
- R2: "Blockchain Basics: A Non-Technical Introduction in 25 Steps" by Daniel Drescher
- R3: "The Metaverse: A Scientific Journey through Virtual Reality, Artificial Intelligence, and the Internet of Things" by Angelica Maria Tactay and Thanh Thuy Vo

Additional Readings:

1. Metaverse for Beginners: Udemyhttps://opensource.com/article/22/6/open-source-metaverse
2. Metaverse 101 : Ultimate Metaverse Course For Beginners | Udemy

METAVERSE AND ITS APPLICATIONS LAB

Program Name	**B.Tech(CSE) with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Metaverse and its Applications Lab	**ENSP372**	0-0-2	2
Type of Course:	(Discipline Specific Elective -I)		
Pre-requisite(s), if any:			

Course Outcomes

COs	
CO1	**Demonstrating** knowledge of Metaverse Technologies and Tools
CO2	**Applying** technical Skills in Virtual and Augmented Reality Development
CO3	**Analyzing** and Evaluate Metaverse Applications
CO4	**Designing** and Creating Immersive Experiences in the Metaverse

Proposed Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	Explore popular VR tools and environments such as Unity3D, Unreal Engine, or Google VR SDK.	CO1
2	Set up the development environment and familiarize yourself with the VR interface.	CO2
3	Design and create a simple VR environment using Unity3D or Unreal Engine.	CO2

4	Add objects, textures, and interactive elements to the environment.	CO2
5	Test the VR environment using a compatible VR headset.	CO2
6	Develop a simple VR game using Unity3D or Unreal Engine.	CO2
7	Create a VR simulation for training purposes, such as a virtual laboratory or a virtual driving simulator.	CO2
8	Develop a VR application for educational purposes, such as virtual classrooms, historical reconstructions, or language learning.	CO2
9	Develop an AR application that uses image recognition to overlay virtual content on recognized images.	CO2
10	Design and implement a VR application for therapeutic or rehabilitation purposes, such as phobia treatment or physical therapy.	CO2
11	Develop a web-based telemedicine application that enables remote patient-doctor consultations.	CO2
12	Design and develop a virtual store or marketplace within a 3D environment.	CO2
13	Create an AR game where virtual objects or characters are overlaidon the real-world environment.	CO2
14	Use Unity3D to create a VR environment, incorporating 3D models, textures, and interactive elements. Test the VR experience using Google Cardboard or other compatible VR headsets.	CO2
15	Develop an AR application using Unity3D and Vuforia. Use Vuforia's image recognition capabilities to overlay virtual content on recognized images or markers. Test the application on a mobile device with a camera.	CO4
16	Develop a simple blockchain-based application for the Metaverse using Ethereum and Solidity. Implement smart contracts to manage digital assets or ownership within the virtual environment.	CO4
17	Train an AI model using TensorFlow to recognize and classify objects within an AR or VR environment. Use the trained model to create interactive experiences or	CO2

	intelligent behaviors within the Metaverse.	
18	Build an AR application focused on educational content. Develop interactive experiences that provide supplementary information or simulations related to specific subjects.	CO3
19	Create an AR simulation that incorporates physics- based interactions between virtual objects and the real world.	CO3
20	Implement an AR application that tracks and applies virtual effects or filters to the user's face in real-time. Use facial tracking libraries such as ARKit or ARCore to accomplish this.	CO4
21	Use open-source design tools like Figma or Sketch to create mockups and wireframes for a Metaverse application.	CO4
22	Use open-source security tools like OWASP ZAP or Burp Suite to perform security assessments and vulnerability scanning on a Metaverse application.	CO4
23	Develop a Metaverse application using an open- source platform like Decentraland or Mozilla Hubs	CO3
24	Conduct user experience testing sessions with participants to gather feedback on the application's usability, navigation, and overall user experience	CO3
25	Analyze the feedback and iterate on the design and functionality of the Metaverse application based on the user input.	CO3

Textbooks and Reading Materials

1. **"Learning Virtual Reality: Developing Immersive Experiences and Applications for Desktop, Web, and Mobile" by Tony Parisi**

    - This book provides a comprehensive introduction to the development of VR applications across various platforms including Unity3D. It is great for students starting to explore popular VR tools and environments, as well as for designing and testing simple VR environments.

2. **"Unity Virtual Reality Projects" by Jonathan Linowes**

    - Explore a range of VR projects using Unity3D that include creating environments, adding interactive elements, and developing VR games. This resource is ideal for those looking to dive deep into Unity3D for creating both VR and AR applications.

3. **"Augmented Reality: Principles and Practice" by Dieter Schmalstieg and Tobias Hollerer**

    - This book offers a solid foundation in AR, including designing applications that use image recognition to overlay virtual content. It covers both the theoretical aspects and practical implementations of AR, making it suitable for experiments involving AR game development and educational applications.

4. **"Virtual Reality for Game Developers" by Jennifer Dawe and Matthew Farber**

    - This text covers the specifics of developing VR games, including using engines like Unreal and Unity3D. It's particularly useful for understanding the integration of interactive elements and testing with VR headsets.

Software and Development Tools

1. **Unity3D:**

    - Usage: This is a versatile game engine that supports both VR and AR development. It can be used to create VR environments, games, simulations, and AR applications. It integrates with various VR platforms like Oculus Rift and HTC Vive, as well as AR platforms like ARKit and ARCore. It's also useful for experiments 1, 3, 6, 7, 8, 14, 15, and 23.

2. **Unreal Engine:**

    - Usage: Known for its high-fidelity graphics capabilities, Unreal Engine is ideal for creating complex VR environments and simulations. It supports VR development for platforms like PlayStation VR, HTC Vive, and Oculus Rift. It's suitable for experiments 1, 3, 6, 12, and 7.

3. **Vuforia:**

    - Usage: This is an augmented reality software development kit (SDK) that is particularly good for AR applications that require image recognition. It works well with Unity3D and is suitable for creating AR applications that overlay digital content over the real world, as described in experiments 9 and 15.

INTRODUCTION TO VIRTUAL REALITY AND AUGMENTED REALITY

Program Name:	**B. Tech CSE with specialization in UI/UX**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Introduction to Virtual Reality and Augmented Reality	**ENSP304**	4-0-0	4
Type of Course:	DSE		
Pre-requisite(s), if any:			

Course Perspective: Students will be able to have an understanding on futuristic technologies to be able to practice and implement technologies in new ideas. To be able to implement after understanding different platforms. The course is divided into 4 modules:

a) Designing for AR

b) Designing for VR

c) Introduction to Internet of Things

d) Project

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

CO1	Defining and analyzing the concept of augmented and virtual reality in depth with the help of different examples.
CO2	Recognizing understand the process of implementing virtual and augmented reality through case studies
CO3	Defining and interpreting Internet of things (IOT) and its application in different industry domains
CO4	Demonstrating and implementing various tools and techniques of Usability testing

CO5	Designing and creating futuristic technologies solutions for different industries utilizing varied

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: Designing for AR**	**No. of hours: 15**
Content Summary: What is augmented reality, Examples, Case studies on augmented reality, implementing augmented reality in different industry domains		
Unit Number: 2	**Title: Designing for VR**	**No. of hours: 15**
Content Summary: What is virtual reality, Examples, Case studies on virtual reality, implementing augmented reality in different industry domains		
Unit Number: 3	**Title: Introduction to Internet of Things**	**No. of hours: 15**
Content Summary: What is Internet of things, Examples, Case studies on IOT, Implementing IOT in different industry domains		
Unit Number: 4	**Title: Project**	**No. of hours: 8**
Content Summary: Project Based		

References

1. R1: "Designing for Wearables: Effective UX for Current and Future Devices- by Scott Sullivan (Author)
2. R2: Designing Bots: Creating Conversational Experiences- by Amir Shevat

3. R3: Designing for Emerging Technologies: UX for Genomics, Robotics, and the Internet of Things- by Jonathan Follett

4. M.K. Bhuyan, Computer Vision and Image Processing: Fundamentals and Applications, CRC Press, USA.

VIRTUAL REALITY AND AUGMENTED REALITY LAB

Program Name:	**B. Tech CSE with specialization in UI/UX**		
Course Name: **Virtual Reality and Augmented Reality Lab**	**Course Code**	**L-T-P**	**Credits**
	ENSP374	0-0-2	1
Type of Course:	DSE		
Pre-requisite(s), if any: (1) Linear Algebra and (2) programming in python			

Defined Course Outcomes

COs	
CO1	**Demonstrating** knowledge of Metaverse Technologies and Tools
CO2	**Applying** technical Skills in Virtual and Augmented Reality Development
CO3	**Analyzing** and Evaluate Metaverse Applications
CO4	**Designing** and Creating Immersive Experiences in the Metaverse

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	Explore popular VR tools and environments such as Unity3D, Unreal Engine, or Google VR SDK.	CO1
2	Set up the development environment and familiarize yourself with the VR interface.	CO2
3	Design and create a simple VR environment using Unity3D or Unreal Engine.	CO2
4	Add objects, textures, and interactive elements to the environment.	CO2

5	Test the VR environment using a compatible VR headset.	CO2
6	Develop a simple VR game using Unity3D or Unreal Engine.	CO2
7	Create a VR simulation for training purposes, such as a virtual laboratory or a virtual driving simulator.	CO2
8	Develop a VR application for educational purposes, such as virtual classrooms, historical reconstructions, or language learning.	CO2
9	Develop an AR application that uses image recognition to overlay virtual content on recognized images.	CO2
10	Design and implement a VR application for therapeutic or rehabilitation purposes, such as phobia treatment or physical therapy.	CO2
11	Develop a web-based telemedicine application that enables remote patient-doctor consultations.	CO2
12	Design and develop a virtual store or marketplace within a 3D environment.	CO2
13	Create an AR game where virtual objects or characters are overlaidon the real-world environment.	CO2
14	Use Unity3D to create a VR environment, incorporating 3D models, textures, and interactive elements. Test the VR experience using Google Cardboard or other compatible VR headsets.	CO2
15	Develop an AR application using Unity3D and Vuforia. Use Vuforia's image recognition capabilities to overlay virtual content on recognized images or markers. Test the application on a mobile device with a camera.	CO4
16	Develop a simple blockchain-based application for the Metaverse using Ethereum and Solidity. Implement smart contracts to manage digital assets or ownership within the virtual environment.	CO4
17	Train an AI model using TensorFlow to recognize and classify objects within an AR or VR environment. Use the trained model to create interactive experiences or intelligent behaviors within the Metaverse.	CO2
18	Build an AR application focused on educational content. Develop interactive experiences that provide supplementary information or simulations related to specific subjects.	CO3
19	Create an AR simulation that incorporates physics- based interactions between virtual objects and the real world.	CO3
20	Implement an AR application that tracks and applies virtual effects or filters to the user's face in real-time. Use facial tracking libraries such as ARKit or ARCore to accomplish this.	CO4
P1	Use open-source design tools like Figma or Sketch to create mockups and wireframes for a Metaverse application.	CO4

P2	Use open-source security tools like OWASP ZAP or Burp Suite to perform security assessments and vulnerability scanning on a Metaverse application.	CO4
P3	Develop a Metaverse application using an open- source platform like Decentraland or Mozilla Hubs	CO3
P4	Conduct user experience testing sessions with 1. participants to gather feedback on the application's usability, navigation, and overall user experience	CO3

MINOR PROJECT-III

Program Name:	**B. Tech CSE with specialization in UI/UX**		
Course Name: Minor Project-III	**Course Code**	**L-T-P**	**Credits**
	ENSI352	---	2
Type of Course:	Project		
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 -IV

Program Name	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: COMPETITIVE CODING -IV	**Course Code**	**L-T-P**	**Credits**
		3-0-0	NIL
Type of Course:	Audit Course		
Pre-requisite(s), if any: Fundamentals of programming			

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 implementing ACID transactions in SQL and NoSQL.
CO4	Solving algorithmic problems using advanced techniques and applying string matching algorithms in coding challenges.

Unit Number: 1	**Title: Foundations and Advanced Concepts in System Design**	**No. of hours: 8**

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.

Unit Number: 2	**Title: Advanced Database concepts**	**No. of hours: 8**

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**.**

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
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

S. No.	Problem Statement	Mapped CO
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
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

- **Engagement through Lecture PPTs**: Students will grasp key concepts of system design, database indexing, and algorithmic techniques through well-structured lecture presentations, ensuring clarity and retention.

- **Problem-Based Theory Assignments**: Assignments will focus on real-world challenges like load balancing and caching strategies, allowing students to apply theoretical knowledge and develop problem-solving skills.
- **Project-Based Lab Work**: Students will design and implement system blueprints, databases, and caching mechanisms through hands-on lab assignments, enhancing practical skills in a collaborative environment.
- **Comprehensive Question Bank**: A diverse question bank covering the entire syllabus will allow students to practice systematically and prepare thoroughly for exams and competitive coding interviews.
- **Model Question Papers and Assessments**: Continuous assessments through quizzes, model papers, and coding challenges will test students' understanding and application of complex concepts like bit manipulation, string matching, and scalability techniques.
- **Support & Feedback System**: Instructors will provide timely feedback on assignments and projects. Students will have access to the course in charge for additional support, ensuring they can clarify doubts and improve performance.
- **Use of ICT Tools & Interactive Boards**: Moodle LMS will host all course materials, assignments, and video lectures, enabling students to access resources anytime. Interactive teaching boards will be used for live demonstrations, enhancing participation.
- **Video Lectures for Critical Topics**: Pre-recorded video lectures on advanced topics like ACID transactions and multithreading will offer students the flexibility to learn at their own pace and review difficult concepts.

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

(Discipline Specific Elective-II)
UX DESIGN FOR FUTURISTIC TECHNOLOGIES - HMI

Program Name:	**B. Tech (CSE with specialization in UI/UX)**		
Course Name: UX Design for Futuristic Technologies - HMI	**Course Code**	**L-T-P**	**Credits**
	ENSP425	4-0-0	4
Type of Course:	(Discipline Specific Elective-II)		
Pre-requisite(s), if any:			

Course Perspective. This course focuses on the design principles and methodologies required for developing user interfaces for futuristic technologies, particularly Human-Machine Interfaces (HMI). It covers the essentials of user experience (UX) design, emphasizing the need for creating intuitive, efficient, and engaging interfaces for advanced technological systems.

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

COs	Statements
CO 1	Understanding the fundamental principles and theories of UX design for Human-Machine Interfaces.
CO 2	Applying user-centered design methodologies to create intuitive and effective interfaces for futuristic technologies.

CO 3	Evaluating and critiquing HMI designs based on usability, accessibility, and user satisfaction criteria.
CO 4	Developing prototypes and conducting usability testing to iterate and improve HMI designs.

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 HMI**	**No. of hours: 12**
Content Summary: What is HMI? Who Uses HMI? Common Uses of HMI, Basic components and functionalities of HMI, Evolution of HMI technology over the years, Overview of industries that utilize HMI technology (e.g., manufacturing, automotive, healthcare). Specific applications of HMI in various sectors, What is the Difference Between HMI and SCADA? Integration of HMI and SCADA in industrial systems.		
Unit Number: 2	**Title: Trends in HMI Technology**	**No. of hours: 12**
Content Summary: Understanding the different technologies of HMI, Past trends and current technologies, High- Performance HMIs, Touch Screens and Mobile Devices, Remote Monitoring, Edge-of-Network and Cloud HMIs Case studies in detail, Characteristics and benefits of high-performance HMIs, Use of touch screens and mobile devices in HMI systems.Practical examples of touch screen and mobile HMI applications.		
Unit Number: 3	**Title: Futuristic HMI's**	**No. of hours: 12**
Content Summary: Understanding the current trends, exploring ways to implement Augmented Reality (AR) and Virtual Reality (VR) to visualize manufacturing functions. How AR and VR can enhance manufacturing functions, Potential applications of AR and VR in HMI systems, Case studies on AR and VR implementation in manufacturing.		
Unit Number: 4	**Title: Human Factors and Ergonomics in HMI**	**No. of hours: 12**
Content Summary:		

Introduction to human factors engineering, Importance of considering human factors in HMI design, Managing cognitive load in HMI systems, Enhancing user experience through ergonomic design, Key principles of ergonomic design for HMI systems, Importance of accessibility in HMI systems, Techniques for designing accessible HMIs.

Learning Experiences:

- Lecture PPTs will provide a clear foundation on UX design principles specifically for futuristic technologies, with a focus on Human-Machine Interaction (HMI) concepts and their practical applications.

- Problem-based theory assignments will engage students in exploring and applying key HMI frameworks, cognitive models, and interaction techniques in the context of next-generation technologies, such as AI, AR/VR, and robotics.

- Project-based lab assignments will offer hands-on experience in designing and prototyping HMI systems, including creating user interfaces for smart devices, immersive environments, and wearable technologies.

- Question banks will reinforce key UX design and HMI concepts, helping students review important topics such as gesture-based interactions, voice interfaces, and adaptive UI for various devices.

- Model question papers will simulate exam conditions, preparing students for assessments on topics such as usability testing, user-centered design for HMI, and ethical considerations in designing for emerging technologies.

- Continuous assessment will offer ongoing feedback through quizzes, peer critiques, and instructor evaluations, allowing students to track their progress in understanding the challenges and innovations in UX for futuristic HMI systems.

- Support & Feedback will be readily available, encouraging collaboration and discussions on cutting-edge HMI trends, case studies, and the role of user experience in enhancing the interaction between humans and machines.
- Mapping /Alignment of COs with POs and PSOs

Textbooks:

1. Digital Anatomy - Applications of Virtual, Mixed and Augmented Reality by Jean-François Uhl, Joaquim

2. Jorge, Daniel Simões Lopes, Pedro F Campos

References

1. Technology-Augmented Perception and Cognition by Tilman Dingler, Evangelos Niforatos
2. Learn Human-Computer Interaction - Solve human problems and focus on rapid prototyping and validating solutions through user testing by Christopher Reid Becker
3. Emotions in Technology Design - From Experience to Ethics by Rebekah Rousi, Jaana Leikas, Pertti Saariluoma

INTRODUCTION TO COMPUTER VISION

Program Name:	**B. Tech (CSE with specialization in UI/UX)**		
Course Name: Introduction to Computer Vision	**Course Code**	**L-T-P**	**Credits**
	ENSP427	0-0-2	1
Type of Course:	Minor (Discipline Specific Elective - II)		
Pre-requisite(s), if any: Nil			

Course Perspective. This course aims to introduce students to the fundamental concepts and techniques in computer vision. It covers various topics such as image processing, feature extraction, object recognition, and computer vision applications. The course is designed to provide both theoretical foundations and practical skills necessary for solving real-world vision problems. The course is divided into 4 modules:

a) Basic Concepts of Image Formation

b) Image Restoration and Coloring

c) Image Compression and Segmentation

d) Object Representation and Computer Vision Techniques

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 implementing projects that integrate multiple image processing algorithms to solve complex 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 Computer Vision**	**No. of hours: 10**

Content: Overview of computer vision, history, and applications. Basic image processing techniques.		
Unit Number: 2	**Title : Image Processing**	**No. of hours: 12**
Content: Techniques for image enhancement, filtering, edge detection, and image segmentation.		
Unit Number: 3	**Title: Feature Extraction**	**No. of hours: 10**
Content: Keypoint detection, descriptors, feature matching, and local feature extraction techniques.		
Unit Number: 4	**Title: Object Detection and Recognition**	**No. of hours: 12**
Content: Object detection methods, classification, neural networks in vision, and practical applications.		

References

1. Computer Vision: Algorithms and Applications by Richard Szeliski
2. Computer Vision and Image Processing: Fundamentals and Applications by Manas Kamal Bhuyan
3. Gonzalez Rafael C. and Woods Richard E., Digital Image Processing, New Delhi: Prentice– Hall of India.
4. M.K. Bhuyan, Computer Vision and Image Processing: Fundamentals and Applications, CRC Press, USA

GRAPHICS DESIGN

Program Name:	**B. Tech (CSE with specialization in UI/UX)**		
Course Name: Graphics Design	**Course Code**	**L-T-P**	**Credits**
	ENSP429	4-0-0	4
Type of Course:	Minor (Discipline Specific Elective - II)		
Pre-requisite(s), if any:			

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

COs	**Statements**
CO 1	Understanding the fundamental concepts of graphic design, including typography, color theory, and layout.
CO 2	Developing proficiency in using graphic design software tools such as Adobe Photoshop, Illustrator, and InDesign.
CO 3	Creating visually compelling designs for various media, including print, web, and social media.
CO 4	Applying design principles to develop branding and marketing materials.
CO 5	Criticallying analyze and provide constructive feedback on design work.

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 Graphic Design**	**No. of hours: 8**
Content: Overview of graphic design History and evolution of graphic design Basic principles of design: balance, contrast, emphasis, movement, proportion, rhythm, and unity Introduction to typography and color theory		
Unit Number:2	**Title: Graphic Design Software Tools**	**No. of hours: 10**
Content: **Introduction to Adobe Photoshop** Basic tools and techniques, Layers, masks, and adjustments **Introduction to Adobe Illustrator** Vector graphics and tools, Creating logos and illustrations **Introduction to Adobe InDesign** Layout design and tools, Creating brochures and magazines		
Unit Number:3	**Title: Design for Different Media**	**No. of hours: 1-**
Content: Print design: posters, flyers, and business cards, Web design: website layout and elements, Social media graphics: creating engaging content, Motion graphics: basics of animation		
Unit Number:4	**Title: Advanced Design Projects**	**No. of hours: 12**
Content: Branding: creating a brand identity, Marketing materials: designing advertisements and promotional items, Portfolio development: compiling and presenting design work , Critique sessions: peer review and feedback on projects		

Text Book:

- T1: Adobe Photoshop Classroom in a Book by Andrew Faulkner, Conrad Chavez
- T2: Adobe InDesign Classroom in a Book by Kelly Kordes Anton, Tina DeJarld
- T3: The Non-Designer's Design Book by Robin Williams

References

- Graphic Design: The New Basics by Ellen Lupton and Jennifer Cole Phillips
- Thinking with Type by Ellen Lupton

(DISCIPLINE SPECIFIC ELECTIVE - III)

DESIGN THINKING FOR PRODUCT MANAGEMENT

Program Name:	**B. Tech (CSE with specialization in UI/UX)**		
Course Name: **Design Thinking for Product Management**	**Course Code**	**L-T-P**	**Credits**
	ENSP431	4-0-0	4
Type of Course:	(Discipline Specific Elective - III)		
Pre-requisite(s), if any:			

Course Perspective. Understanding the cycle of product design. Be able to find and execute the technology required. Understanding the importance of product management. To be able to execute the cycle of product management. The course is divided into 4 modules:

a) Introduction to Product Lifecycle management

b) Product development platform

c) Product Lifecycle Things

d) Project

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

COs	Statements
CO1	Defining and understanding Product Lifecycle and Product Lifecycle Management
CO2	Identifying and understanding identify the phases of Product lifecycle
CO3	Examining Product life cycle stages, Benefits, areas of PLM

CO4	Understanding Phases of product lifecycle and corresponding technologies
CO5	Designing the Product Life Cycle using numerous instruments of Product Development 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.

Course Outline:

Unit Number: 1	**Title: Business UX**	**No. of hours: 12**
Content Summary: Understanding How a UX approach can help any business, The Business Value of UX Design, Strategy building, Aspects of key guidelines in UX business, values and emotions of user Behavior and cognitive psychology of market and business, Design policies, Importance of understanding business requirements, Discovering business goals		
Unit Number: 2	**Title: Stakeholder and Competitive Analysis**	**No. of hours: 12**
Content Summary: Importance of understanding business requirements, discovering business goals, Internal and external stakeholders, stakeholder analysis, stakeholder interviewing, meeting stakeholder expectations and feedback, Direct and Indirect Competitors, Competitor Analysis and its practice, Steps to Conduct Competitor analysis, Parameters to conduct competitor analysis		
Unit Number: 3	**Title: Design Management**	**No. of hours: 12**
Content Summary: What is design management, Taking Charge of Processes and People, The Evolution of Design Management, Areas of Design Management, Why Does Design Management Matter, Where Does Design Management Fall Within Businesses? Value Proposition Canvas, Creating a UX Roadmap		

Unit Number: 4	**Zeplin and Jira and Introduction to Product lifecycle management**	**No. of hours: 12**
Content Summary: Learning how to develop and deliver documentation using Zeplin and How to communicate well and assign tasks among and within teams using Jira, what is Product Lifecycle Management (PLM)? What is the Product Life Cycle? Product life cycle stages, Benefits, areas of PLM.		

Learning Experiences:

- Lecture PPTs will offer a detailed introduction to the principles of Design Thinking, guiding students through its stages—Empathize, Define, Ideate, Prototype, and Test—within the context of product management.

- Problem-based theory assignments will engage students in applying Design Thinking methodologies to real-world product challenges, focusing on customer needs, pain points, and innovative solutions that align with business goals.

- Project-based lab assignments will provide hands-on experience in developing product concepts, including user research, journey mapping, rapid prototyping, and user testing to iterate and improve product ideas.

- Question banks will help students thoroughly review core Design Thinking concepts and their application to product lifecycle management, ensuring they understand how to incorporate user-centered design into business strategies.

- Model question papers will simulate exam conditions, preparing students for assessments on topics like market research, product innovation, and decision-making frameworks that leverage Design Thinking principles in product management.

- Continuous assessment will provide ongoing feedback through interactive workshops, peer evaluations, and instructor critiques, helping students track their progress in understanding how Design Thinking drives successful product development.

- Support & Feedback will be readily available, with opportunities for group collaboration and mentoring to enhance practical application, encouraging students to explore case studies of innovative product solutions created using Design Thinking.

Text Books:

1. The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses by Eric Ries

References

1. Fundamentals of User-Centered Design by Still and Crane
2. UX Strategy: How to Devise Innovative Digital Products that People Want: Jaime Levy

ADOBE PHOTOSHOP

Program Name:	**B. Tech (CSE with specialization in UI/UX)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Adobe Photoshop	**ENSP433**	4-0-0	4
Type of Course:	(Discipline Specific Elective - III)		
Pre-requisite(s), if any: Nil			

Course Perspective: This course enhances the ability to create accurate masks and image effects, retouch images, work with video files, automate repetitive tasks, and integrate with other Adobe applications. The course is divided into 4 modules:

a) Introduction to Adobe Photoshop

b) Photo corrections

c) Working with Selections

d) Layer and channels

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

COs	Statements
CO1	Recalling and identifying the basic concepts, tools, and functionalities in Adobe Photoshop
CO2	Applying various photo correction techniques to enhance and modify images.
CO3	Analyzing different selection tools in Adobe Photoshop, to make precise selections and manipulate selected areas.
CO4	Analyzing and evaluating the usage of masks and channels for advanced image editing.
CO5	Designing the Product Life Cycle using numerous instruments of Product Development 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.

Course Outline:

Unit Number: 1	**Title: Introduction to Adobe Photoshop**	**No. of hours: 10**

Content:
Raster vs vector scanning, creating new image, saving files for print, saving files for web/screen, working for adobe bridge, introduction: using the tool, option bar and other panel, undoing actions, customizing the workspace, tools
panel overview

Unit Number: 2	Title: Photo corrections	No. of hours: 10

Content:
Strategy for retouching, resolution and image size, Adjusting the color in Camera Raw, Straightening and cropping the image in Photoshop, Replacing colors in an image, Adjusting saturation with the Sponge tool, Repairing areas with the Clone Stamp tool, Using the Spot Healing Brush tool, Using content- aware fill, Applying the Unsharp Mask filter.
Mini project: Use the Spot Healing Brush, Clone Stamp, or Content-Aware Fill to remove any unwanted elements or blemishes in the photos.

Unit Number: 3	Title: Working with Selections	No. hours: of 8

Content:

About selecting and selection tools, Using the Quick Selection tool, Moving a
selected area, Manipulating selections, Using the Magic Wand tool, Selecting with the lasso tools, Rotating a selection, Selecting with the Magnetic Lasso

Unit Number: 4	Title: Layer and channels	No. of hours: 10

Content:

Content: Layers, Using the Layers panel, Rearranging layers, Applying a gradient to a layer, Applying a layer style, Flattening and saving files. Working with masks and channels, Creating a mask, Refining a mask, Creating a quick mask, Manipulating an image with Puppet Warp, Working with channels.
Mini project: Create a documentation or presentation showcasing the original
images, the layered composition, and a brief description of the layer and channel techniques applied

References

- "Adobe Photoshop Classroom in a Book" by Adobe Creative

- Photoshop for Beginners: Learn the Basics of Photoshop in Under 10 Hours!" by Nathaniel Dodson
- Photoshop Elements 2024 For Dummies" by Barbara Obermeier

INTRODUCTION TO FIGMA

Program Name:	**B. Tech (CSE with specialization in UI/UX)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Introduction to Figma	**ENSP435**	4-0-0	4
Type of Course:	(Discipline Specific Elective - III)		
Pre-requisite(s), if any:			

Course Perspective. The course provides an in-depth understanding of Figma, a collaborative interface design tool, enabling students to design, prototype, and gather feedback all in one place.

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

COs	Statements
CO 1	Understanding the basics of Figma and its interface
CO 2	Developing skills to create and manipulate design elements using Figma
CO 3	Learning to prototype interactive elements and create user flows
CO 4	Collaborating with team members in real-time using Figma's collaborative features
CO 5	Applying Figma's design systems and components to ensure consistency and efficiency in design work

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 Figma**	**No. of hours: 8**
Content: Overview of Figma, Understanding the Figma interface, Basic tools and features		
Unit Number:2	**Title: Designing with Figma**	**No. of hours: 1 0**
Content: Creating and managing design files, Working with shapes and text, Using layers and groups		
Unit Number:3	**Title: Prototyping in Figma**	**No. of hours: 8**
Content: Creating interactive prototypes, Linking screens and adding transitions, User testing and feedback collection		
Unit Number:4	**Title: Advanced Figma Features**	**No. of hours: 10**
Content: Design systems and components, Collaboration and version control, Exporting assets and handoff to developers		

Text Books:

1. Figma Essentials: The Ultimate Guide to Mastering Figma by John Doe.
2. Collaborative Interface Design with Figma by Jane Smith.

Reference Book:

- Prototyping with Figma: Techniques and Best Practices by Robert Brown.

Additional Readings:

1. The Design Systems Handbook by InVision.
2. Figma Documentation and Tutorials available on the official Figma website.
3. Online articles and case studies on collaborative design tools and their impact on team productivity.

Portfolio Development & Review

Program Name:	**B. Tech (CSE with specialization in UI/UX)**		
Course Name: Portfolio Development & Review	**Course Code**	**L-T-P**	**Credits**
	SEC046	2-0-0	2
Type of Course:	SEC		
Pre-requisite(s), if any: NA			

Course Perspective. To learn the importance of portfolio and creating a UX Portfolio for career and Internship recruitment. To prepare students for internship and placement recruitments. To boost students' confidence in applying for different roles in UX/UI in different companies. To ensure the right set of information reach the recruiters' to profile about the student, that brings the best out their skills and strengths.

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

COs	Statements
CO1	Defining and analyzing the concept of designing the portfolio resume
CO2	Recognizing and understanding the process of creation of portfolio
CO3	Defining and interpreting product life cycle things
CO4	Demonstrating and implementing various tools and techniques of portfolio designing
CO5	Designing and creating futuristic technologies solutions for designing different portfolio on different design UX/Roles

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 Portfolio Development**	**No. of hours: 6**
Content Summary: Understanding the role of a portfolio in professional development, how a well-crafted portfolio can enhance job prospects and career growth, Overview of different types of portfolios (e.g., personal, academic, professional), Examples and case studies of successful portfolios in various fields. Identifying the key components of a portfolio (e.g., cover page, table of contents, resume, work samples, project descriptions, reflections).		
Unit Number: 2	**Title: : Collecting and Curating Work Samples**	**No. of hours: 6**
Content Summary: Emphasizing the importance of quality work over the quantity of samples, Reviewing and refining selected work samples for inclusion. Techniques for visually documenting work samples (e.g., photos, screenshots, diagrams), Importance of reflection in portfolio development. Writing reflective pieces that provide insights into personal and professional growth.		
Unit Number: 3	**Title: Digital Portfolio Creation**	**No. of hours: 9**
Content Summary: Overview of popular platforms and tools for creating digital portfolios (e.g., Behance, Dribbble, WordPress, Wix). Criteria for selecting the right platform based on individual needs, Principles of UX design for an effective digital portfolio, Ensuring ease of navigation and user-friendly interface. Principles of visual design, including layout, color schemes, typography, and branding. Creating a consistent and professional look and feel. Best practices for publishing and promoting the digital portfolio. Strategies for driving traffic to the portfolio and reaching the target audience.		
Unit Number: 4	**Title: Reviewing and Refining the Portfolio**	**No. of hours: 9**
Content Summary: Techniques for analyzing feedback from peers, mentors, and industry professionals. Identifying areas for improvement based on feedback. Implementing changes and improvements to enhance the portfolio. Continuous iteration and refinement of portfolio content and design. Final steps for polishing and proofreading the portfolio.		

Text Books:

1. Show Your Work!: 10 Ways to Share Your Creativity and Get Discovered by Austin Kleon
2. Building Design Portfolios: Innovative Concepts for Presenting Your Work by Sara Eisenman

3. The Portfolio: An Architecture Student's Handbook by Igor Marjanović and Katerina Rüedi Ray
4. Digital Portfolio: Build an Impressive Online Portfolio and Land Your Dream Job by Steve Johnson

Summer Internship-III

Program Name:	**B. Tech (CSE with specialization in UI/UX)**		
Course Name: Summer Internship-III	**Course Code**	**L-T-P**	**Credits**
	ENSI451	---	2
Type of Course:	INT		
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 6th semester and before the commencement of the 7th 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**
- 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

Applied Programming and Problem-Solving Skills for Campus Interviews

Program Name:	**B. Tech (CSE specialization with UI/UX)**		
Course Name: Applied Programming and Problem-Solving Skills for Campus Interviews	**Course Code**	**L-T-P**	**Credits**
		---	2
Type of Course:	MOOC		
Pre-requisite(s), if any: NA			

Objective: The Boot Camp (Training and Placement) module is a comprehensive course designed to equip final-year B. Tech, BCA, MCA, and B. Sc students with the necessary skills and knowledge to excel in campus placement drives. All students are required to pass the individual components to receive the final marks out of 100 and earn 2 credits for the Practical Training Module (Bootcamp training) in their course structure. Students must obtain specific **free certifications** from Infosys Springboard (https://infytq.onwingspan.com/web/en/page/home).

Course Outcomes (CO)

COs	Statements
CO1	Applying data structures and algorithms to solve complex programming problems.
CO2	Implementing object-oriented programming principles and developing robust Java applications.
CO3	Designing and managing databases efficiently using advanced SQL and database management techniques.
CO4	Demonstrating readiness for campus placements through successful participation in mock interviews, AMCAT assessments, and obtaining certifications from Infosys Springboard, showcasing industry-relevant skills and knowledge

SESSION WISE DETAILS		
Module: 1	Data Structures and Algorithms - Part 1	No. of hours: 30

		(Online, Self-Paced)
Content Summary: Module 1 covers foundational data structures including arrays, strings, and linked lists. It introduces key operations and practical applications. This foundational knowledge is crucial for advancing to more complex data structures.		
Module: 2	Data Structures and Algorithms - Part 2	No. of hours: 30 (Online, Self-Paced)
Content Summary: Module 2 focuses on advanced data structures such as stacks, queues, trees, and graphs. It covers essential operations, real-world applications, and their role in solving complex problems. Understanding these structures is vital for effective problem-solving and algorithm optimization.		
Module: 3	Object Oriented Programming	No. of hours:46 (Online, Self-Paced)
Content Summary: covers the fundamental concepts of OOP, including classes, objects, inheritance, polymorphism, and encapsulation. It emphasizes designing and implementing software using these principles to enhance code reusability and maintainability.		
Module: 4	Programming using Java	No. of hours: 113 (Online, Self-Paced)
Content Summary: basics of Java, including syntax, data types, operators, and control structures. It covers object-oriented principles specific to Java, such as classes, objects, inheritance, and polymorphism, along with advanced topics like exception handling and file I/O. The module aims to build strong foundational skills in Java for developing robust applications.		
Module: 5	Database management Systems (part I)	No. of hours: 64 (Online, Self-Paced)
Content Summary: fundamental concepts of database systems, including database models, relational databases, and SQL. It covers key topics such as entity-relationship modeling, normalization, and basic query operations. The module provides a solid foundation in designing and managing databases, focusing on effective data retrieval and manipulation using SQL.		
Module: 6	Database management Systems (part II)	No. of hours: 40 (Online, Self-Paced)

Content Summary: advanced database concepts, including transaction management, concurrency control, and database security. This module covers complex SQL queries, stored procedures, and triggers, as well as performance optimization techniques. It focuses on enhancing your skills in managing and optimizing large-scale databases, ensuring data integrity, and implementing robust security measures.		
Module: 7	Aptitude Exam	No. of hours: Online
Module: 8	Independent Evaluation through 3rd party	No. of hours: Offline
Module: 9	Softs Skills	No. of hours: Online
Module: 10	MOCK Interview	No. of hours: Hybrid
Reference Books: ▪ "Introduction to Algorithms" by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein ▪ "Cracking the Coding Interview" by Gayle Laakmann McDowell ▪ "Elements of Programming Interviews" by Adnan Aziz, Tsung-Hsien Lee, and Amit Prakash ▪ "Head First Java" by Kathy Sierra and Bert Bates ▪ "Database System Concepts" by Abraham Silberschatz, Henry F. Korth, and S. Sudarshan ▪ "Introduction to the Theory of Computation" by Michael Sipser ▪ "Programming Challenges: The Programming Contest Training Manual" by Steven S. Skiena and Miguel A. Revilla ▪ "The Algorithm Design Manual" by Steven S. Skiena ▪ "Algorithms" by Robert Sedgewick and Kevin Wayne ▪ "Effective Java" by Joshua Bloch		

Evaluation Criteria:

Module	Link	Hrs	Marks
Data Structures and Algorithms - Part 1	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01254096991326208001065_shared/overview	30	10

Data Structures and Algorithms - Part 2	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01276673846938828834 48_shared/overview	38	10
Object Oriented Programming	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01254097227492556810 63_shared/overview	46	10
Programming using Java	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01288046454761881634 7_shared/overview	113	10
Database management Systems (part I)	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01275806667282022456_shared/overview	64	10
Database management Systems (Part II)	https://infytq.onwingspan.com/web/en/app/toc/lex_auth_01276730056291942 41_shared/overview	40	
Aptitude Exam	Online Sessions to be conducted by KRMU	NA	10
AMCAT (Aspiring Minds Computer Adaptive Test)	To be organized by KRMU through External agency	NA	20
Softs Skills	Online Sessions to be conducted by KRMU	NA	10
MOCK Interview	To be organized by KRMU	NA	10
			100

*Please Note: No end term evaluations will be done

Semester VIII

Program	B.Tech (CSE specialization with UI/UX)		
Course Name: **Industrial Project/R&D Project/Start-up Project**	**Course Code**	**L-T-P**	**Credits**
	ENSI452		**12**
Type of Course:	**SEC**		
Pre-requisite(s), if any:			

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 that aligns with the principles of the **National Education Policy (NEP) 2020**.

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.

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. 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.