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 2025-26 session)

Approved in the 38th Meeting of Academic Council

Held on 28th June 2025

Table of Contents

S.N.	Content	Page No.
1	Preface	3
2	Categories of Courses	6
3	University Vision and Mission	8
4	About the School	8
5	School Vision & Mission	10
6	About the Program	11
7	Program Educational Objectives (PEO)	12
8	Program Objectives (PO)	12
9	Program Specific Objectives (PSO)	13
10	Career Avenues	14
11	Duration	16
12	Eligibility Criteria for Award of Degree	16
13	Student's Structured Learning Experience from Entry to Exit in the Programme	16
14	Scheme of Studies	25
15	Syllabus	33

1. 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 AI & Robotics, 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

2. Categories of Courses

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

Discipline Specific Elective Courses (DSE): The Discipline Specific Elective (DSE) courses provide advanced, domain-specific knowledge that allows UX/UI students to specialize in emerging areas such as motion design, game UX, information visualization, human-computer interaction, and design for emerging technologies like AR/VR and IoT. These courses foster deeper understanding and innovation by encouraging critical thinking, research orientation, and creative exploration. Through project-based learning and industry-relevant challenges, students develop specialized skills that align with niche career paths in the evolving landscape of user experience and interface design.

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 designed to enrich students' academic journey by fostering humanistic and ethical values, promoting holistic well-being, and cultivating essential 21st-century skills. These courses go beyond the traditional curriculum to equip students with life skills, digital fluency, entrepreneurial mindset, emotional intelligence, and a deeper understanding of self and society. The VACs are aligned with the ethos of truth, righteous conduct, peace, love, non-violence, and universal human values, along with scientific and technological advancements that promote global citizenship.

- Digital and Technological Solutions
- Health, Wellness, Yoga, and Emotional Intelligence
- Personal Growth and Purposeful Living
- Understanding India and Indian Knowledge Systems
- Innovation and Social Responsibility

Community Service (CS): Courses are designed to nurture civic consciousness, empathy, and a sense of social responsibility among students through structured engagement with real-world community needs. These courses aim to instill values of inclusivity, ethical responsibility, and participative citizenship by encouraging students to connect classroom learning with grassroots realities. Through collaborative projects, fieldwork, and reflective practice, students develop critical life skills including teamwork, problem-solving, ethical decision-making, and leadership. The course also emphasizes the importance of contributing to sustainable

development goals and understanding diverse socio-economic contexts, thereby fostering holistic personal and professional growth.

Project: The B.Tech in Computer Science and Engineering (UX/UI) follows a structured project-based learning approach, including **Minor Projects in Semesters 2, 4, and 6**, and **Major Projects in Semesters 7 and 8**. These projects progressively build students' technical, design, and research skills—starting from foundational tasks to advanced, real-world problem solving. Guided by faculty mentors, students work on innovative solutions, with final outcomes potentially leading to **research publications, conference presentations, or patents**, ensuring strong academic and industry readiness.

3. 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

4. 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.

5. 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.

6. 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).

The B.Tech in Computer Science and Engineering (UX/UI) with academic support from ImaginXP is a future-centric, interdisciplinary undergraduate program that blends core computing skills with in-demand design capabilities, preparing students to build intuitive, inclusive, intelligent digital experiences. The program is designed in full alignment with the National Education Policy (NEP) 2020, promoting multidisciplinary learning, flexible curriculum design, and career-ready outcomes.

While students develop strong foundations in computer programming, software engineering, databases, data structures, and web technologies, they also receive deep domain exposure to User Experience (UX) and User Interface (UI) design. The

specialization includes hands-on learning in design thinking, wireframing, visual storytelling, usability testing, AR/VR prototyping, voice interface design, and inclusive product development, supported by global certifications from Microsoft, Google, LinkedIn Learning, Skillshare, and the Interaction Design Foundation.

Throughout the program, students build portfolios through real-world projects, participate in industry mentorship, and are equipped with competencies to excel in roles like UX Designer, UI Developer, Interaction Designer, Product Researcher, and AR/VR Experience Designer.

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

- Future-ready interdisciplinary curriculum blending core computer science with in-demand UX/UI design skills, aligned with the National Education Policy (NEP) 2020.
- Professionally qualified, competent, and committed teaching faculty, supported by academic and industry experts from ImaginXP.
- Industry-enabled curriculum with regular training and mentorship from leading professionals in design and technology domains.
- Consistent interaction with renowned academicians, design practitioners, and technology leaders through workshops, webinars, and guest lectures.

- Project-based and hands-on learning with emphasis on real-world applications, design thinking, wireframing, AR/VR prototyping, usability testing, and inclusive design.
- Techno-pedagogy, research projects, field work, internships, and a continuous and comprehensive evaluation system integrated throughout the course.
- Portfolio development support through studio projects, live industry briefs, and certification-backed learning from platforms like Microsoft, Google, LinkedIn Learning, Skillshare, and the Interaction Design Foundation.
- Access to certification courses, value-added programs, and skill development modules beyond the core curriculum, boosting professional readiness.
- Strong career guidance, counselling, and mentoring ecosystem for success in both personal and professional spheres.
- Support programs for both advanced and slow learners, promoting inclusion, student diversity, and customized learning tracks.
- Dedicated focus on career progression through industry internships, placement training, entrepreneurship exposure, and preparation for higher studies.
- Career pathways include roles like UX Designer, UI Developer, Interaction Designer, Product Researcher, and AR/VR Experience Designer.

7. 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.

8. 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.

9. 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.

10. 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, behaviours, 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.

11. Duration

4 Years (8 Semesters) - Full-Time Program

12. 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 168 to be eligible for award of degree without any backlog.

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

a. Education Philosophy and Purpose:

Learn to Earn a Living: At KRMU we believe in equipping students with the skills, knowledge, and qualifications necessary to succeed in the job market

and achieve financial stability. All the programmes are tailored to meet industry demands, preparing students to enter specific careers and contributing to economic development.

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

b. University Education Objective

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

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

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

environment for sustained intellectual growth, encouraging learners to achieve their full potential.

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

- Inside 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 (Java), 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.

Evaluation Scheme:

Evaluation Components	Weightage
A. Internal Assessments: **1. Continuous Assessment (40 Marks)** **(Minimum 5 components to be used and to be evenly spaced)** Project/ Quizzes/Test/ Assignments and Essays/ Presentations/ Class Participation/Lab Work/ Case Studies/ Reflective Journals	**40 Marks**
2. Mid Term Exam	**20 Marks**
B. External Assessments: End term Examination	**40 Marks**
Total	**100 Marks**

g. Feedback and Continuous Improvement Mechanisms:

K.R. Mangalam University is deeply committed to academic excellence through a robust **feedback and continuous improvement system**. This system is designed to gather comprehensive input from a diverse range of stakeholders, including **students, faculty, alumni, employers, and academic peers**. Feedback is systematically collected and thoroughly

analyzed to identify areas for enhancement in **curricula, teaching methodologies, and academic processes**. Based on the insights gained, actionable measures are formulated and communicated to the appropriate bodies for timely implementation.

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

h. Academic Integrity and Ethics:

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

13.1 University Education Objective

- Focus Employability and Entrepreneurship through Holistic Education

13.2 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.

14. Scheme Of Studies

Program Name	B.Tech (CSE specialization with UI/UX)
Total Credits	176
Total Semesters	8

Program Name	I	II	III	IV	V	VI	VII	VIII	Total Credits
B. Tech CSE-UX/UI	23	22	27	26	23	24	16	14	176

SEMESTER I

S.N	Course Code	Category	Course Title	L	T	P	Credits
1	**ETCCCM101**	**Major**	**Computational Mathematics - I**	**4**	**0**	**0**	**4**
2	**ETCCPP102**	**Major**	**Programming for Problem Solving Using Python**	**3**	**0**	**2**	**4**
3	**ETCCWD103**	**Major**	**Web Dev -I (HTML,CSS, JS Basics)**	**3**	**0**	**2**	**4**
4	**ETCCCH104**	**Major**	**Engineering Chemistry**	**2**	**0**	**2**	**3**
5	**ETCCCP105**	**Major**	**Computer Science Fundamentals & Career Pathways**	**4**	**0**	**0**	**4**
6	**ETSEDP112**	**SEC**	**Design Thinking & Prototyping / Maker Lab: Tinkering with Technology**	**1**	**0**	**2**	**2**
7	**UNVAES001**	**VAC-I**	**Environmental Studies & Disaster Management**	**2**	**0**	**0**	**2**
TOTAL				**19**	**0**	**8**	**23**

SEMESTER II

S.N	Course Code	Category	Course Title	L	T	P	Credits
1	**ETCCCM201**	**Major**	**Computational Mathematics - II**	**4**	**0**	**0**	**4**
2	**ETCCDS202**	**Major**	**Data Structures**	**3**	**0**	**2**	**4**
3	**ETCCWD203**	**Major**	**Web Dev -II (Advanced JS & React)**	**3**	**0**	**2**	**4**
4	**ETCCPH204**	**Major**	**Engineering Physics**	**2**	**0**	**2**	**3**

5	**ETCCPR205**	**Proj**	**Minor Project-I**	**0**	**0**	**4**	**2**
6	**ETSEMT126**	**SEC**	**Maker Lab: Tinkering with Technology/Design Thinking & Prototyping**	**1**	**0**	**2**	**2**
7		**Open Elective**	***Open Elective-I**	**3**	**0**	**0**	**3**
TOTAL				**16**	**0**	**12**	**22**

Note:

- The marks for the Minor Project-I (**ETCCPR205**) will be allocated solely through internal evaluation, with a total of 100 marks. There will be no end-term exams for this project.
- Open Elective: Students can choose one of the electives from the pool of open electives of university.
- Summer Internship (6-8 Weeks)

SEMESTER III

S.N	Course Code	Category	Course Title	L	T	P	Credits
1	**ETCCNT301**	**Major**	**Nand to Tetris-I**	**3**	**0**	**2**	**4**
2	**ETUXDT305**	**DSE**	**Specialization Course-I (Introduction to UI/UX & Design Thinking)**	**3**	**0**	**2**	**4**
3	**ETCCWD303**	**Major**	**Web Dev -III (Node.js &**	**3**	**0**	**2**	**4**

			Express Backend)				
4	**ETCCMS304**	**Major**	**Database Management Systems**	**3**	**0**	**2**	**4**
5	**ETCCDA302**	**Major**	**Design & Analysis of Algorithms**	**3**	**0**	**2**	**4**
6	**AEC**	**AEC**	**Verbal Ability**	**2**	**0**	**0**	**2**
7	**SEC**	**SEC**	**Competitive Coding - I**	**2**	**0**	**0**	**2**
8	**ETCCIN305**	**INT**	**Summer Internship-I**	**0**	**0**	**4**	**2**
9	**CS**	**CS**	**Community Service**	**1**	**0**	**0**	**1**
TOTAL				**20**	**0**	**14**	**27**

Note:

- For the "Summer Internship," students are required to complete a 6-8 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.

SEMESTER IV

S.N	Course Code	Category	Course Title	L	T	P	Credits
1	**ETCCNT401**	**Major**	**Nand to Tetris-II**	**3**	**0**	**2**	**4**
2	**ETUXIA402**	**DSE**	**Specialization Course-II (User Research & Information Architecture)**	**3**	**0**	**2**	**4**
3	**ETCCCD403**	**Major**	**Cloud Computing & DevOps**	**3**	**0**	**2**	**4**

4	**ETCCJP404**	**Major**	**Object-Oriented Programming with Java**	**3**	**0**	**2**	**4**
5	**OEC**	**Open Elective**	***Open Elective-II**	**3**	**0**	**0**	**3**
6	**AEC**	**AEC**	**Communication & Personality Development**	**2**	**0**	**0**	**2**
7	**SEC**	**SEC**	**Competitive Coding - II**	**2**	**0**	**0**	**2**
8	**ETCCPR405**	**Proj**	**Minor Project-II**	**0**	**0**	**4**	**2**
9	**CS**	**CS**	**Club/Society**	**1**	**0**	**0**	**1**
TOTAL				**20**	**0**	**12**	**26**

Note:

- For the "Minor Project," students will undergo internal evaluation, which will be graded on a scale of 100 marks.
- Open Elective: Students can choose one of the electives from the pool of open electives of University

SEMESTER V

S.N	Course Code	Category	Course Title	L	T	P	Credits
1	**ETUXID505**	**DSE**	**Specialization Course-III (Interaction Design & Prototyping)**	**3**	**0**	**2**	**4**
2	**ETUXDP503**	**DSE**	**Specialization Course-IV (Visual &**	**3**	**0**	**2**	**4**

			Interface Design Principles)				
3	**ETCCOS502**	**Major**	**Operating Systems**	**3**	**0**	**2**	**4**
4	**ETCCMD501**	**Major**	**Principles and Practices of System Design**	**4**	**0**	**0**	**4**
5	**AEC**	**AEC**	**Arithmetic and Reasoning**	**2**	**0**	**0**	**2**
6	**SEC**	**SEC**	**Competitive Coding - III**	**2**	**0**	**0**	**2**
7	**ETCCIN504**	**INT**	**Summer Internship-II (Assessment)**	**0**	**0**	**4**	**2**
8	**VAC**	**VAC-II**	**Value Added Course (VAC)**	**2**	**0**	**0**	**2**
TOTAL				**19**	**0**	**10**	**24**

Note:

- For the "Summer Internship," students must complete a 6-8 weeks 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.

SEMESTER VI

S.N	Course Code	Category	Course Title	L	T	P	Credits
1	**ETUXUD605**	**DSE**	**Specialization Course-V (Advanced UI Development & Handoff)**	**3**	**0**	**2**	**4**
2	**ETUXUT602**	**DSE**	**Specialization Course-VI (Usability**	**3**	**0**	**2**	**4**

			Testing & Accessibility)				
3	**ETCCAF601**	**Major**	**Next-Gen Software Engineering with Agile Frameworks**	**3**	**0**	**2**	**4**
4	**ETCCCN603**	**Major**	**Computer Networks**	**3**	**0**	**2**	**4**
5	**AEC**	**AEC**	**Comprehensive Placement Preparation**	**2**	**0**	**0**	**2**
6	**SEC**	**SEC**	**Competitive Coding - IV**	**2**	**0**	**0**	**2**
7	**ETCCPR604**	**Proj**	**Minor Project-III**	**0**	**0**	**4**	**2**
8	**VAC**	**VAC-III**	**Value Added Course (VAC)**	**2**	**0**	**0**	**2**
TOTAL				**18**	**0**	**12**	**24**

Note:

- For the "Minor Project," students will undergo internal evaluation, which will be graded on a scale of 100 marks.
- Summer Internship (6-8 Weeks)

SEMESTER VII

S.N	Course Code	Category	Course Title	L	T	P	C
1	**ETUXUI703**	**DSE**	**Specialization Course-VII (UI/UX Integration)**	**3**	**0**	**2**	**4**
2	**ETCCTT704**	**Major**	**Generative AI Tools & Techniques**	**3**	**0**	**2**	**4**

3	**ETCCPR701**	**PROJ**	**Major Project-I**	**0**	**0**	**8**	**4**
4	**ETCCIN702**	**INT**	**Summer Internship-III (Assessment)**	**0**	**0**	**4**	**2**
5	**MOOC**	**MOOC-I**	**MOOC (Swayam/ NPTEL/AICTE's ELIS)**	**2**	**0**	**0**	**2**
TOTAL				**8**	**0**	**16**	**16**

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.

Semester VIII

S.N	Course Code	Category	Course Title	L	T	P	C
1	**ETCCIN801**	**INT**	**Summer Internship-IV (Assessment)**	**0**	**0**	**16**	**8**
2	**ETCCPR802**	**PROJ**	**Major Project-II**	**0**	**0**	**8**	**4**
3	**MOOC**	**MOOC-II**	**MOOC (Swayam/ NPTEL/AICTE's ELIS)**	**2**	**0**	**0**	**2**
TOTAL				**2**	**0**	**24**	**14**

Syllabus

SEMESTER: 1

Computational Mathematics-I

Program Name	B.Tech (CSE) with specialization in UI/UX			
Course Name: Computational Mathematics-I	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCCM 101	4-0-0	4	60
Type of Course:	Major			
Pre-requisite(s): Basic knowledge of high school algebra and mathematical reasoning.				

Course Perspective. This course builds the mathematical and computational backbone required for later B.Tech subjects such as Data Structures & Algorithms, Machine Learning, Computer Graphics, Cryptography, and Operating Systems. Students learn core ideas from discrete mathematics, linear algebra, probability & statistics, and calculus/optimization, then implement a *small set* of high-impact algorithms in Python (NumPy/SciPy/SymPy). By the end, they can translate theory into efficient code, reason about correctness, and analyze computational complexity.

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

COs	Statements
CO 1	**Explaining** fundamental mathematical structures used in computation and prove basic properties.
CO 2	**Modeling** real-world engineering problems with linear algebra, probability, and calculus.

CO 3	**Selecting & implementing** essential numerical and statistical methods in Python.
CO 4	**Evaluating** algorithmic correctness, stability, and convergence for practical problems.

Course Outline:

Unit Number: 1	**Discrete Mathematics Essentials**	**No. of hours: 15**

Content:

- Relations, equivalence relations; closures and partial orders
- Recurrence relations (linear homogeneous & Master Theorem)
- Boolean algebra, Karnaugh maps, logic minimization
- Finite-state machines & basic automata links
- Propositional / predicate logic; proof techniques (direct, contradiction, induction)
- Sets, functions, permutations & combinations
- Introductory graph theory (BFS, DFS) and algorithm correctness proofs

Essential Practical Tasks

- **Recurrence Solver & Complexity Checker** – write a function that takes a divide-and-conquer recurrence and predicts asymptotic runtime, then validates it empirically.
- **Graph Traversal Visualizer** – implement BFS & DFS on an input graph and animate the search order (network-analysis use case).

Unit Number: 2	**Linear Algebra for Computation**	**No. of hours: 15**

Content:

- Vector spaces, rank, basis, linear independence
- Matrix algebra; LU & QR factorisation
- Eigenvalues, eigenvectors, spectral decomposition

- Singular Value Decomposition (SVD) and dimensionality reduction (PCA)
- Least-squares and normal equations
- Condition number & numerical stability
- Applications: linear regression, robotics kinematics, PageRank intuition

Essential Practical Tasks

- PCA on Image Data – compute SVD of an image dataset, reconstruct with k components, and plot reconstruction error.
- Least-Squares Linear Regression – implement and benchmark closed-form vs. NumPy linalg.lstsq on calorie-burn vs. heart-rate data.

Unit Number: 3	Title: Probability & Statistics	No. of hours: 15

Content:

- Sampling methods; Central Limit Theorem
- Random variables, expectation, variance
- Common distributions: Bernoulli, Binomial, Poisson, Normal
- Bayes' theorem and naïve Bayes intuition
- Hypothesis testing: z-, t-, chi-square; p-values & confidence intervals
- Correlation, covariance, simple linear regression
- Introduction to Markov chains and steady-state behaviour

Essential Practical Tasks

- A/B Test Simulator – generate click-through data, run a two-sample t-test, output statistical significance and power.
- Markov-Chain Weather Model – build a 2-state chain (sun/rain) from historical data and forecast n-day probabilities.

Unit Number: 4	Calculus Foundations & Gradient-Based Optimization	No. of hours: 15

Content:

- Limits, continuity, differentiation & integration (single variable)
- Multivariable functions; partial derivatives, gradient, Jacobian, Hessian

- Taylor expansion and error terms
- Gradient descent, learning rate, convergence criteria
- Constrained optimisation (Lagrange multipliers – conceptual)
- Back-propagation perspective for neural nets
- Applications: loss-function minimisation, curve-fitting

Essential Practical Tasks

1. **Symbolic vs. Numeric Differentiation** – compute analytical gradient with SymPy and compare to finite-difference approximation for f(x)=xsin⁡xf(x)=x\sin xf(x)=xsinx.
2. **From-Scratch Gradient Descent** – minimise MSE for a univariate linear model; plot loss vs. iterations and discuss convergence.

Learning Experience

Inside Classroom Learning:

1. **Interactive Lectures:** Use slides and board work to explain concepts like relations, recurrence, logic, FSMs, and graph theory.
2. **Hands-on Proof Techniques:** Solve and present problems using direct, contradiction, and induction proofs.
3. **Recurrence Relation Mastery:** Derive and solve linear recurrences using Master Theorem and iterative expansion.
4. **Boolean Logic Sessions:** Perform logic simplification using Karnaugh maps in class.
5. **Graph Theory Exploration:** Implement BFS and DFS during guided lab walkthroughs.
6. **Collaborative Learning:** Group-based whiteboard sessions to solve predicate logic and function-related problems.
7. **Concept Reinforcement:** Weekly quizzes with discussion-based feedback and revision loops.

Outside Classroom Learning Experience

1. **Take-Home Assignments:** Solve recurrence relations, logic proofs, and graph problems independently.
2. **Tool-Based Practice:** Use Python or Jupyter notebooks for recurrence checking and visualizing graph traversals.
3. **Peer Discussion Forums:** Collaborate on logic and automata problems using college LMS or discussion boards.
4. **Self-Guided Projects:** Mini-projects like building a logic-based decision system or simulating a finite state machine.
5. **Online Resources & Simulations:** Use open-source tools or YouTube/OCW content to visualize predicate logic and FSMs.

Textbooks:

- **Mathematics for Computer Science**, Eric Lehman, F. Thomson ("Tom") Leighton, and Albert R. Meyer, Downloadable via MIT OpenCourseWare
- **Mathematics for Machine Learning** – Deisenroth, Faisal & Ong (Cambridge, 2020)
- **Linear Algebra and Learning from Data** – Gilbert Strang (2019)
- **Discrete Mathematics with Applications** – Susanna Epp, 5e (2024)
- **Probability and Statistics for Engineering and the Science**, Jay L. Devore, 10th Edition, Cengage Learning (2020)

Programming for Problem Solving using Python

Programme Name:	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Programming for Problem Solving using Python**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCPP102	3-0-2	4	45
Type of Course:	Major			
Pre-requisite(s), if any: Working Knowledge of any programming language like C will be an added advantage				

Course Perspective: This hands-on course teaches you how to solve real problems with Python—from writing your first script to analysing and visualising data. You will master core syntax, control flow, functions, object-oriented design, file/exception handling, and the use of NumPy, Pandas, and Matplotlib. Through a series of mini-projects you will practise turning ideas into working programs, cleaning and exploring datasets, and presenting clear visual insights.

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

COs	Statements
CO 1	**Writing** and **debugging** basic Python scripts using variables, data types, operators, and strings.
CO 2	**Applying** control flow, functions, and built-in data structures to implement algorithmic solutions.
CO 3	**Designing** modular programs with classes, file I/O, and robust exception handling.
CO 4	**Loading**, **processing**, **analyzing**, and **visualizing** real-world data using NumPy, Pandas, and Matplotlib.

Course Outline:

Unit Number: 1	**Title: Python Foundations & Developer Workflow**	**No. of hours: 12**
Content: • *Pythonic* mindset; REPL vs. script vs. Jupyter Notebook • Installing CPython & Miniconda; creating virtual environments (venv, conda) • VS Code debugging, linting with flake8 / pylint, formatting with black • Variables, numeric/boolean types, strings (slicing, f-strings), basic I/O • Operators, precedence, simple expressions in list comprehensions • Quick Win: create a requirements.txt, run python -m venv .venv, commit initial setup to Git		
Unit Number: 2	**Title: Control Flow, Functions & Data Structures**	**No. of hours: 12**
Content: • Conditionals & multi-branch logic patterns • Loops, comprehension patterns, generator expressions • Functions: positional vs. keyword args, *args/**kwargs, docstrings, type hints • Algorithmic idioms—frequency maps, sliding window, two-pointer, recursion vs. iteration • Built-ins: lists, tuples, sets, dicts; collections.Counter, defaultdict, namedtuple • Complexity snapshot: Big-O of list vs. dict operations • Quick Win: solve 3 HackerRank "warm-up" problems, push to Git branch		
Unit Number: 3	**Title: OOP, Modules, Packaging & Robust Code**	**No. of hours: 12**

Content:

- OOP pillars; classes, dataclasses, private vs. public, property decorators
- Inheritance vs. composition; mix-ins; dunder methods for iteration & comparison
- Building and using packages; __init__.py, namespace packages, relative imports
- Testing & CI intro: unittest / pytest, writing first test, GitHub Actions workflow
- File handling with pathlib; JSON & CSV serialization; intro to SQLite with sqlite3
- Exceptions, custom error classes, logging levels, with statement everywhere
- Quick Win: publish a tiny package to TestPyPI

Unit Number: 4	**Title: Data Acquisition, Analysis & Visualization**	**No. of hours: 9**

Content:

- NumPy essentials: vectorised maths, broadcasting, boolean indexing
- Pandas: DataFrame creation, merging, groupby, time-series resampling, basic plot
- Web data ingestion: requests + simple REST/JSON parsing; rate-limit etiquette
- Exploratory plots in Matplotlib & Seaborn; saving to PNG/SVG, notebook vs. script modes
- Storytelling: selecting the right chart, adding annotations, exporting Markdown reports
- Quick Win: fetch live crypto prices, compute 24 h moving average, push chart to repo wiki

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Unit 1: "Daily Calorie Tracker CLI"** *Real-world context:* many students want a quick way to monitor calorie intake. **Sub-objectives** 1. Collect meal names and calorie values from the user via input() and convert to numeric types. 2. Store entries in lists; compute total and average calories with arithmetic operators. 3. Use comparison operators to warn when the daily limit is exceeded. 4. Format a neatly aligned summary report with f-strings and escape characters. 5. Save the session log to a plain-text file using simple write operations (bonus).	CO 1
2	**GradeBook Analyzer** *Real-world context:* lecturers need quick statistics on student marks. **Sub-objectives** 1. Read a set of marks (entered or loaded from a small CSV) into lists and dictionaries. 2. Implement functions to calculate class average, median, highest and lowest scores. 3. Use loops and conditionals to assign letter grades and count grade distribution. 4. Demonstrate list comprehensions for filtering pass/fail students.	CO 2

	5. Present results in a formatted table and allow repeated analysis until the user exits.	
3	**Library Inventory Manager"** *Real-world context:* the campus library wants a simple console app to track books. **Sub-objectives** 1. Define a Book class with attributes (title, author, ISBN, status) and __str__. 2. Create methods to issue, return, and display book details; override as needed. 3. Persist the catalogue in a JSON file; load it safely at program start. 4. Handle missing-file and I/O errors with try–except–finally. 5. Provide a menu-driven CLI that lets staff add, search, or update books.	CO 3
4	**Weather Data Visualizer** *Real-world context:* students analyse local climate trends for a sustainability report. **Sub-objectives** 1. Load a real CSV of daily weather data into a Pandas DataFrame. 2. Clean missing values, compute monthly averages with group-by operations. 3. Create line, bar, and scatter plots in Matplotlib; arrange subplots in one figure. 4. Use NumPy to calculate mean, max, min, and standard deviation for temperature. 5. Export the final plots as PNGs and summarise insights in a Markdown report.	CO 3
5	**Campus Energy-Use Dashboard" – facilities team wants insights on building electricity consumption.**	CO 4

	• Data ingestion: read multiple monthly CSV files into DataFrames; handle missing/ corrupt files with exceptions. • Core logic: write functions & loops to calculate daily, weekly, and building-wise totals; store results in lists/dicts. • OOP layer: design Building and Meter Reading classes with methods for aggregation and reporting. • Visual output: create at least three charts (trend line, comparative bar, peak-hour scatter) in a single Matplotlib figure. • Persistence & presentation: save cleaned data to a new CSV, export the figure, and print a concise textual executive summary—all from one Python script	

Learning Experiences

Inside Classroom Learning:

1. Interactive Coding Exercises: Students practice writing Python programs in real-time with in-class coding challenges related to loops, data types, and control structures.
2. Live Code Reviews: Students submit code for peer review, receiving constructive feedback on maintaining clean code principles.
3. Hands-on File Handling Tasks: Students work on file manipulation tasks, exploring real-world scenarios using CSV, JSON, and regular expressions.
4. Mini-Projects: Small projects like building a basic text-based game to apply Python basics and data structures.
5. Web Scraping Practice: Implement web scraping in class to extract data from websites using Python libraries.
6. Error-Handling Workshops: Collaboratively debug code with exception-handling techniques, promoting troubleshooting skills.

7. Function Design Practice: Group exercises in designing clean, efficient functions using lambda, map, and filter functions.

Outside Classroom Learning:

1. Online Coding Competitions: Participation in coding challenges on platforms like LeetCode or HackerRank to practice clean code.
2. Industry Guest Lectures: Attending talks by professionals about Python's application in cybersecurity and real-world coding practices.
3. Group Web Scraping Projects: Students collaborate on web scraping projects using real-world data to extract insights from websites.
4. Database Connectivity Assignments: Real-time project work with MySQL databases, creating systems to fetch and insert data.
5. Open-Source Contributions: Students contribute to open-source Python projects, adhering to clean coding standards.
6. Hackathons: Participation in hackathons focused on building Python-based solutions for cybersecurity challenges.
7. Documentation Writing: Students practice writing clear, concise documentation for their Python projects, reinforcing clean coding habits

Text and Reference Book

Eric Matthes, "Python Crash Course: A Hands-On, Project-Based Introduction to Programming," 3rd Edition, No Starch Press, 2023. ISBN 978-1-71850-264-2.

Web Dev -I (HTML ,CSS, JS Basics)

Program Name	B.Tech (CSE) with specialization in UI/UX			
Course Name: Web Dev -I (HTML ,CSS, JS Basics)	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCWD103	3-0-2	4	45
Type of Course:	Major			
Pre-requisite(s): Familiarity with basic computer operations and logical thinking				

Course Perspective. This beginner-level course is designed for BSc Computer Science students to learn the fundamentals of web development. It covers HTML for structuring web pages, CSS for styling and layout, and basic JavaScript for simple interactivity. Students will work with real-world mini-projects and gain confidence in building responsive websites using simple, structured code.

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

COs	Statements
CO 1	Creating structured and accessible web pages using HTML5.
CO 2	Analyzing style and layout pages using basic CSS including Flexbox.
CO 3	Writing simple JavaScript programs using functions, loops, and arrays.
CO 4	Building a complete basic website using HTML, CSS, and JavaScript.

Course Outline:

Unit Number: 1	**Title: HTML Basics**	**No. of hours: 10**
• Introduction to HTML and its structure. • Basic tags: headings, paragraphs, links, images, lists. • Tables and Forms: basic usage and attributes.		

- Semantic elements: header, footer, section, nav.
- Tools: Using VS Code and Live Server.

Mini project: Simple personal webpage with sections and contact form.

Unit Number: 2	**Title: CSS Fundamentals**	**No. of hours: 15**

Content:

- CSS syntax and selectors.
- Styling text, colors, backgrounds, and borders.
- Box Model: padding, margin, border.
- Basic Layouts using Flexbox.
- Responsive design with media queries.
- Simple hover effects and transitions.

Mini project: Responsive personal portfolio layout

Unit Number: 3	**Title: JavaScript Essentials**	**No. of hours: 10**

Content:

- Introduction to JavaScript and use in HTML.
- Variables, data types, operators.
- Control structures: if, else, loops.
- Functions and basic scope.
- Arrays and basic operations.

Mini project: Simple interactive quiz using prompt and alert.

Unit Number: 4	**Title: Final Project**	**No. of hours: 10**

Content:

- Build a responsive multi-section webpage.
- Use HTML for structure, CSS for design, and basic JS for interaction.
- Incorporate form, images, navigation, and simple JS logic.
- Project themes: Portfolio, Event Page, Product Showcase.

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Simple Web Page with HTML (Unit 1)** **Objective:** Learn HTML structure and basic tags. **Tasks:** • Create a personal info page using HTML • Include sections: About, Hobbies, Contact • Use headings, paragraphs, lists, images, and links • Create a basic contact form with name and message fields	CO 1
2	**Styled Portfolio Page (Unit 2)** **Objective:** Apply CSS to style and layout your page. **Tasks:** • Style the HTML page using colors, fonts, spacing • Use **Flexbox** for header, body, and footer layout • Add media queries for mobile responsiveness • Implement hover effects for links and buttons	CO 2
3	**JavaScript Quiz App (Unit 3)** **Objective:** Write your first logic-based JavaScript application. **Tasks:** • Create a quiz using prompt() and alert() • Store questions in an array • Loop through questions and compare answers • Show total score with a final message • Provide a menu-driven CLI that lets staff add, search, or update books.	CO 3
4	**Final Responsive Website (Unit 4)**	CO 3

	Objective: Build a mini project combining all learned technologies. **Tasks:** • Design a one-page responsive website • Use **semantic HTML**, **Flexbox CSS layout**, and **basic JS** • Add a navigation bar, image gallery, and contact form • Ensure the layout is mobile-friendly and includes basic interactivity	
5	**Capstone: Mini Project Website** **Objective:** Design and build a full small website to represent a real-world theme. **Choose One Theme:** • Personal Portfolio • Online Bookstore • Event Page (e.g., College Fest) • Food Menu Display • Local Service Listing (e.g., Tutors, Plumbers)	CO 4

Requirements:

- Multiple sections (Home, Services/Projects, Contact)
- Clean layout using **HTML5 + CSS3**
- Responsive design using **Flexbox** and **media queries**
- Add basic interactivity (e.g., quiz, form validation, calculator, toggle themes) using JavaScript
- Well-commented code and organized folder structure

Deliverable:

- Fully functional, responsive, and visually structured mini-site
- Submit all files (HTML, CSS, JS) in a zipped folder

Learning Experiences

Inside Classroom Learning*:*

1. **Live Coding Sessions:** Students build web pages in real-time using HTML tags, CSS styling, and basic JavaScript logic under instructor guidance.
2. **Component-Based Page Building:** Create structured page sections like headers, navigation bars, and forms using semantic HTML and CSS box model.
3. **CSS Styling Challenges:** Apply Flexbox and layout properties to style web pages with custom fonts, colors, and spacing.
4. **JavaScript Snippet Walkthroughs:** Solve in-class tasks using loops, conditionals, and functions to enable interactivity (e.g., form validation, image sliders).
5. **Mini Web Projects:** Develop small web applications like personal portfolio, landing pages, or interactive quiz apps.
6. **Peer Code Reviews:** Students review each other's HTML/CSS/JS code for structure, responsiveness, and proper indentation.
7. **Browser DevTools Practice:** Use Chrome/Firefox Developer Tools to inspect elements, debug CSS, and test JavaScript behavior.

Outside Classroom Learning:

1. **Web Design Case Studies:** Analyze popular websites (e.g., Netflix, Zomato) for layout strategies, responsiveness, and UI/UX best practices.
2. **CodePen/JSFiddle Practice:** Build mini UI components and animations outside class using online sandbox environments.
3. **Responsive Redesign Tasks:** Redesign basic web pages for mobile, tablet, and desktop views using media queries.
4. **Online Tutorials & Practice:** Complete structured exercises on platforms like freeCodeCamp, W3Schools, or MDN Web Docs.
5. **Open Web Projects:** Collaborate on GitHub to build simple group websites (e.g., college club sites or event pages).

6. **Web Accessibility Review:** Evaluate and improve accessibility features such as alt text, semantic tags, and contrast ratios using online tools like WAVE or Lighthouse.

Textbooks:

1. *Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics* (5th Edition), Robbins, Jennifer Niederst., O'Reilly Media, 2018. ISBN: 978-1491960202
2. HTML & CSS: Visual QuickStart Guide – 9th Ed., Elizabeth Castro, Bruce Hyslop, Peachpit Press (Pearson), 2021, 978-0136581444

ENGINEERING CHEMISTRY

Department:	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: Engineering Chemistry	**Course Code** : **ETCCCH104**	**L –T- P**	**Credits**	**Contact Hours**
		2-0-2	3	30
Type of Course:	**Major**			
Pre-requisite(s), if any: Basic knowledge of high school-level chemistry, including atomic structure, chemical bonding, and periodic classification.				

Course Perspective: Students should have a basic understanding of chemistry, including energy changes, and calorific value, is essential for analysing fuels. Familiarity with electrochemistry, including redox reactions, electrochemical cells, and electrode potential, will help in understanding battery technology and fuel cells. Awareness of environmental science, particularly water pollution, water treatment methods, and sustainable energy sources, will help in relating theoretical concepts to real-world applications.

Course Outcomes (CO)

COs	Statements
CO1	**Understanding** the methods for water hardness and the basics of boiler water treatment and analyzing fuel calorific values and battery.
CO2	**Explaining** the characteristics and working principles of different battery types and types of fuels.
CO3	**Determining** the differences between batteries and fuel cells and classify fuel cell types based on their components.

CO4	**Identifying** between hard and soft water, solve the related numerical problems on water purification and emf and its significance in industry and daily life.

Course Outline:

Unit Number: 1	**Title: Water technology**	**No. of hours: 6**

Content Summary:

- **Introduction, water analysis**: Hardness-determination by EDTA method, Treatment of boiler feed water: Internal treatment (Phosphate, Colloidal and Calgon conditioning).
- **External treatments**: Ion exchange and lime-soda process, Zeolite processes. Boiler scales formation and ill effects, methods of prevention of scales. Numerical problems

Unit Number: 2	**Title: Chemical Fuels**	**No. of hours: 8**

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, Knocking in IC engine, its ill effects and prevention of knocking. Anti-knocking agent: Leaded and unleaded petrol.
- **Gaseous fuels**: LPG, CNG and their applications.

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

Content Summary:

- **Introduction -** Galvanic cell, electrode potential, EMF of the cell and cell representation. Batteries and their importance,

- **Classification of batteries-** primary, secondary and reserve batteries with examples. Battery characteristics - voltage, capacity, energy density, power density, energy efficiency, cycle life and shelf life.
- **Construction, working and applications of**: Ni-Cd, and Lithium-ion battery. Working principle of an electric vehicle.

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

Content Summary:

- Basic concepts of polymer,
- Types of polymers,
- Thermoplastic & thermosetting plastics,
- Preparation and application of some industrially important polymers (Natural rubber, Buna S, Buna-N, Neoprene, Isoprene, Nylon-6, nylon-6,6 , Decron and Terylene).
- Conducting and biodegradable polymers.

Lab Experiments

Ex. No	**Experiment Title**	**Mapped CO/COs**
1	Determination of temporary and permanent hardness in water sample using EDTA.	CO1
2	Determination of alkalinity in the given water sample.	CO1
3	Determination of viscosity of given liquid.	CO2
4	Determination of surface tension of given liquid.	CO3
5	Preparation of Phenol-formaldehyde resin.	CO2
6	Preparation of Urea-formaldehyde resin.	CO3
7	Determination of chloride content in water sample.	CO2
8	Estimation dissolved oxygen (DO) content in the given water sample by Winkler's method.	CO4
9	Determination of iron content in the given solution by Mohr's method.	CO3

10	To determine the calorific value of a given fuel sample using a bomb calorimeter.	CO4
11	Preparation of a nickel complex $[Ni(NH_3)_6]Cl_2$ and estimation of nickel by complexometric titration.	CO2
12	Synthesis of drug like Aspirin, /Paracetamol etc.	CO4

Learning Experiences

Inside Classroom Learning:

1. **Interactive Concept Lectures:** Use visual aids and models to explain atomic structure, periodic properties, electrochemistry, polymers, and water chemistry in an engaging manner.
2. **Demonstration-Based Learning:** Live demonstrations and video-based visualizations of chemical phenomena like corrosion, electrolysis, and polymerization reactions.
3. **Numerical Problem Solving:** Practice numericals on Nernst equation, pH calculations, water hardness, and polymer molecular weights during class hours.
4. **Group Discussions:** In-class team-based activities to debate and analyze topics such as green chemistry, renewable energy sources, and advanced materials.
5. **Case Study Integration:** Analyze case studies of industrial failures or innovations, such as corrosion in bridges or polymer use in aerospace.
6. **Quick Quizzes and Flash Recaps:** Conduct short quizzes and instant polls using tools like Kahoot or Google Forms for topic reinforcement.
7. **Whiteboard Reactions:** Students draw and explain chemical reactions and structures of compounds like polymers and electrochemical cells collaboratively.

Outside Classroom Learning:

1. **Lab-Based Experiments:** Hands-on experiments like water analysis (hardness, alkalinity), polymer synthesis, and electrochemical cell testing with detailed lab reports.
2. **Virtual Chemistry Simulations:** Use platforms like PhET, ChemCollective, or Labster for simulating chemical reactions and instrumentation virtually.
3. **Industrial Exposure Visits:** Visit to water treatment plants, battery manufacturing units, or chemical industries to observe real-world chemistry applications.
4. **Research Poster Creation:** Create informative posters on green chemistry, nanotechnology in healthcare, or smart materials for intra-college exhibitions.
5. **Online Learning Modules:** Engage with SWAYAM/NPTEL content on fuel cells, corrosion control, or spectroscopic techniques.
6. **Mini Research Projects:** Topic-focused projects such as corrosion behavior in local water pipelines or analysis of eco-friendly polymers.
7. **Science Communication Blogs:** Write simplified blogs or infographics on engineering chemistry topics to communicate science to the general audience.

Text Books:

1. **Materials Science and Engineering: An Introduction;** William D. Callister and David G. Rethwisch; John Wiley & Sons, 10th Edition, 2020; **ISBN-13 (10th Ed.):** 978-1119405620
2. **Engineering Chemistry;** P.C. Jain and Monica Jain; Dhanpat Rai Publishing Company (P) Ltd.; 20th Edition.
3. **Electrochemical Methods: Fundamentals and Applications**; Allen J. Bard and Larry R. Faulkner; John Wiley & Sons; 3rd Edition; 2000; SBN 13(3rdEd.): 978-0471043720.

Computer Science Fundamentals & Career Pathways

Program Name	B.Tech (CSE) with specialization in UI/UX			
Course Name: **Computer Science Fundamentals & Career Pathways**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCCP105	4-0-0	4	60
Type of Course:	Major			
Pre-requisite(s): None				

Course Perspective. This course is designed to introduce first-year B.Tech Computer Science students to the fundamentals of computing, the digital ecosystem, and career pathways in modern computer science. The course combines foundational computing principles, hands-on Linux usage, computational thinking, career awareness, and exposure to essential tools and technologies used in the industry. Through practical sessions, reflection-based activities, and personalized exploration tasks, students will build a solid technical and professional base for the rest of their academic journey.

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

COs	Statements
CO 1	Describing foundational computing concepts and apply computational thinking to solve simple real-world problems.
CO 2	Demonstrating proficiency in navigating Linux environments and using essential shell commands and tools.
CO 3	Identifying and explore emerging technologies and relate them to career domains in computer science.

CO 4	Utilizing modern programming and collaboration tools effectively for learning and development.

Course Outline:

Unit Number: 1	**Title: Foundations of Computer Science & Computational Thinking**	**No.of hours: 15**

Content:

- **Evolution of computing:** milestones, hardware generations, key contributors
- **Components of a computer system**: CPU, memory, storage, input/output, buses
- **Number systems**: decimal, binary, octal, hexadecimal conversion
- **Character encoding schemes:** ASCII, Unicode, UTF-8
- **Software types:** system software, application software, utilities
- **Algorithms and Flow of Data**: basic structure of an algorithm, input–process–output model
- **Computational Thinking Framework**: Abstraction, Decomposition, Pattern Recognition, Algorithm Design
- **Flowcharts and Pseudocode**: drawing and interpreting basic flowcharts, writing pseudocode for simple tasks
- Introduction to digital ethics, privacy, responsible use of technology

Unit Number: 2	**Title: Basics of Linux and Open-Source Tools**	**No. of hours: 12**

Content:

- Importance of Linux in modern computing: hosting, cloud, AI/ML, cybersecurity
- Linux architecture basics: kernel, shell, filesystem
- Installing and accessing Linux: Ubuntu on WSL/VirtualBox
- Basic shell commands: navigation, file operations, viewing content, system status
- ls, cd, pwd, cp, mv, rm, cat, more, less, clear, mkdir, rmdir
- User and file permission management: chmod, chown, whoami
- Process monitoring: top, ps, kill

- Network and download tools: ping, wget, curl
- Introduction to shell scripting: writing simple bash scripts
- Philosophy of open-source: community, contribution, licenses (MIT, GPL)

Unit Number: 3	Title: Emerging Technologies & Personalized Career Exploration	No. of hours: 12

Content:

- Introduction to major computing domains
- Real-world applications and case studies from India and abroad

Exploratory Task:

  - Choose one technology domain
  - Research its trends, job roles, required skills
  - Present findings in the form of a short blog post, infographic, or 3-min video

- Mapping university subjects to career domains (e.g., DSA → Backend Dev, Linux → DevOps)
- Role of interdisciplinary computing (e.g., CS in healthcare, agriculture)
- Showcasing alumni examples, Indian startup stories, and student innovation models.

Unit Number: 4	Title: Tools for Programming, Learning, and Collaboration	No. of hours: 12

Content:

- IDEs for development:
  - Installing and using Visual Studio Code
  - Introduction to Replit and Jupyter Notebooks
- Version control with Git and GitHub:
  - Basic commands: init, add, commit, push, clone
  - Creating and managing repositories
  - Using .gitignore, README files
  - GitHub Classroom or Portfolio project
- Markdown syntax for technical documentation
- Online coding and learning platforms:
  - HackerRank, LeetCode, W3Schools, Kaggle basics
- Note-taking & productivity tools: Notion, Obsidian
- Team collaboration tools: MS Teams, Slack, Discord, Trello

Unit Number: 5	Title: Career Planning, Certifications & Industry Readiness	No. of hours: 9

Content:

- Goal setting: SMART goals for academic and career planning
- Industry-recognized certifications
- Building a professional digital identity
- Participating in:
  - Hackathons and coding contests
  - Open-source contributions
  - Internships and community programs (e.g., GSoC, MLH, Smart India Hackathon)
- Importance of a growth mindset, peer networking, and time management

Learning Experiences

Inside Classroom Learning:

1. **Interactive Lectures:** Core CS topics like hardware/software systems, OS, networking, and programming logic are taught through real-world analogies and digital aids.
2. **CS Career Mapping Workshops:** Group activities to explore domains such as Software Development, Data Science, Cybersecurity, AI/ML, and UX Design.
3. **Tech Timeline Discussions:** Analyze the evolution of computing, from early machines to quantum computing, through timelines and innovation spotlights.
4. **Career Talk Sessions:** Discussions on job roles, internship paths, and roadmap breakdowns for various career tracks in tech.
5. **Resume & LinkedIn Profile Labs:** Hands-on sessions to build strong technical resumes and optimize LinkedIn for job/internship visibility.
6. **Mock Interviews & HR Readiness:** Simulated technical and behavioral interviews with feedback on articulation, confidence, and technical clarity.

7. **Ethics in Tech Debates:** Classroom debates on data privacy, AI bias, tech responsibility, and open-source ethics.

Outside Classroom Learning:

1. **Alumni/Industry Expert Talks:** Guest sessions with professionals from startups, MNCs, and academia sharing career journeys and field insights.
2. **Virtual Lab Exploration:** Explore virtual environments simulating OS behavior, basic networking, and computer architecture concepts.
3. **Tech Career Research Projects:** Students research emerging fields (e.g., Blockchain, Cloud Computing) and present scope, roles, and required skills.
4. **Portfolio Website Development:** Build and deploy a personal web portfolio to showcase skills, projects, and career objectives.
5. **Hackathon & Bootcamp Participation:** Engage in events focused on problem-solving, coding, team collaboration, and tech exposure.
6. **Online Certification Modules:** Complete courses on Coursera, Google Career Certificates, or IBM SkillsBuild to gain domain-specific credentials.
7. **Peer Mentoring Circles:** Form study groups or peer mentoring chains to prepare for aptitude, coding rounds, and domain clarity.

Textbooks:

1. Computer Science: An Overview – J.G. Brookshear & D. Brylow, Pearson Education , 2021
2. The Linux Command Line: A Complete Introduction, William E. Shotts Jr., 2019, 13th Edition, *ISBN-13:* 978-1593279523.

Design Thinking & Prototyping

Program Name	B.Tech (CSE) with specialization in UI/UX			
Course Name: **Design Thinking & Prototyping**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		1-0-2	2	15
Type of Course:	SEC			
Pre-requisite(s): Basic understanding of problem-solving and a willingness to engage in collaborative, user-centered exploration.				

Course Perspective. This course equips first-semester engineering students with critical problem-solving skills using Design Thinking methodology. Through hands-on studio and lab sessions, students will apply iterative prototyping, user-centered design principles, and digital tools to create meaningful solutions addressing real-world problems on campus.

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

COs	Statements
CO 1	Identifying user needs and frame design challenges through empathy-driven research.
CO 2	Generating multiple innovative solutions utilizing creative ideation techniques.
CO 3	Developing functional digital prototypes using Figma.
CO 4	Performing usability evaluations and effectively analyze feedback.
CO 5	Presenting and communicating design ideas persuasively, demonstrating iterative improvement.

Course Outline:

<table>
<tr><td>Unit Number: 1</td><td>Title: Empathy and Problem Framing</td><td>No. of hours: 5</td></tr>
<tr><td colspan="3">Content
▪ Introduction to Design Thinking (Definition, Importance, IDEO Shopping Cart case study)
▪ Empathy techniques (Interviews: structured vs. unstructured, active listening, Observation: direct, contextual inquiry)
▪ Creation of Empathy Maps (capturing thoughts, feelings, actions, pain points)
▪ Defining User Personas (demographics, behaviors, goals, pain points, motivations)
▪ Problem statements formulation (POV and How Might We (HMW) statements)
Activities/Projects:
▪ IDEO Shopping Cart analysis workshop
▪ Conducting interviews and observations (2-3 participants per student)
▪ Developing empathy maps and detailed user personas
▪ Crafting precise Point-of-View statements and How Might We questions</td></tr>
<tr><td>Unit Number: 2</td><td>Title: Ideation and Paper Prototyping</td><td>No. of hours: 3</td></tr>
<tr><td colspan="3">Content:
• Creative Ideation strategies (SCAMPER method: Substitute, Combine, Adapt, Modify, Put to another use, Eliminate, Reverse)
• Mind Mapping (generating diverse and structured ideas)
• Rapid prototyping principles (paper wireframes, basic UI components)
• Feedback mechanisms (structured critique, peer feedback loops)

Activities/Projects:
• Conduct team brainstorming session resulting in 8–10 varied solution ideas
• Selection of best ideas based on feasibility, desirability, viability</td></tr>
</table>

- Creation of detailed paper prototypes illustrating selected solutions
- Facilitating structured feedback sessions

Unit Number: 3	Title: Digital Prototyping with Figma	No. of hours: 2

Content:

- Overview of digital prototyping tools (Figma interface overview, components, frames)
- Developing mid-fidelity prototypes (button interactions, navigation design, prototyping interactions and linking)
- Collaborative workflows in digital prototyping (team sharing, feedback loops, version control basics)

Activities/Projects:

- Hands-on lab exercises introducing essential Figma tools and prototyping basics
- Team collaboration to build a functional clickable mid-fidelity prototype
- Mid-project pitch highlighting user problems, personas, and initial prototype designs

Unit Number: 4	Title: Testing, Iteration, and Presentation	No. of hours: 5

Content:

- Principles of usability testing (setting test objectives, user tasks creation)
- Conducting heuristic evaluations (Nielsen's usability heuristics, user feedback analysis)
- Fundamentals of UI/UX design (consistency, affordances, visual aesthetics, Gestalt principles)
- Iterative design process (implementing feedback, iterative prototype improvement)
- Effective storytelling and presentation techniques

Activities/Projects:

- Organizing peer-led usability testing sessions utilizing heuristic evaluation checklists

- Iterative refinements of prototypes based on systematic user feedback
- Developing comprehensive final presentation materials
- Executing final showcase presentations demonstrating polished and functional prototypes

Learning Experiences

Inside Classroom Learning:

- **Empathy Mapping Exercises:** Students conduct interviews and observations to build empathy maps and define user personas for real-world problems.
- **Problem Framing Workshops:** Hands-on activities to reframe vague challenges into actionable problem statements using "How Might We" questions.
- **Ideation Sessions:** Group brainstorming with tools like mind mapping, SCAMPER, and 6-3-5 method to generate creative design solutions.
- **Storyboarding & User Journey Mapping:** Visualize end-user experiences through storyboards and journey maps for better problem understanding.
- **Low-Fidelity Prototyping:** Students create quick prototypes using paper, cardboard, or digital wireframing tools to visualize solutions.
- **In-Class Design Jams:** Timed design sprints where teams ideate and prototype around a given theme or problem.
- **Feedback & Iteration Loops:** Peer and instructor feedback sessions to test assumptions and refine prototype iterations.

Outside Classroom Learning:

1. **Field Observation & User Interviews:** Students interact with target users in real-world settings to identify pain points and validate needs.
2. **Figma/Adobe XD Prototyping Practice:** Develop interactive digital prototypes using UI/UX design tools to simulate real user flows.
3. **Design Thinking Case Study Reviews:** Analyze case studies of companies like IDEO, Apple, or Airbnb applying design thinking to solve complex challenges.

4. **Rapid Prototyping Challenges:** Participate in online or community design challenges to build fast, iterative solutions under constraints.
5. **Hackathons & Designathons:** Join multi-disciplinary events to apply the complete design thinking cycle in collaborative team settings.
6. **Peer Testing in Public Spaces:** Conduct informal usability testing with target users to gather insights and identify design improvements.
7. **Reflection Journals:** Maintain a journal documenting each phase of the design thinking process, personal insights, and learnings.

Textbooks/References

1. The Design Thinking Toolbox: A Guide to Mastering the Most Popular and Valuable Innovation Methods; Michael Lewrick, Patrick Link, Larry Leifer; John Wiley & Sons, Inc., 1st Edition, 2020, **ISBN-13:** 978-1119629191.

2. Stanford d.school Bootcamp Bootleg
    - IDEO Design Kit (https://www.designkit.org/)
    - Don Norman, The Design of Everyday Things
    - Steve Krug, Don't Make Me Think
3. Figma Resources:
    - Figma Education Resources
    - FreeCodeCamp Figma Crash Course (YouTube)

Evaluation Scheme (with Rubrics):

- Studio Work & Participation (20%): Regularity, quality of discussions, in-class assignments (ideation boards, journey maps).
- Midterm Project Pitch (20%): Clarity of problem definition, innovativeness of solutions, depth of empathy research, initial paper prototype.
- Final Prototype (40%): Complexity and interactivity of the digital prototype, quality of iterative documentation.
- Final Presentation & Viva (20%): Effectiveness of presentation, storytelling clarity, response to questions, individual contribution reflection.

SEMESTER: 2

Computational Mathematics – II

Program Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Computational Mathematics - II**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCCM201	4-0-0	4	60
Type of Course:	Major			
Pre-requisite(s): Single variable calculus, Matrices, Differentiation and Integration, Basics of Python				

Course Perspective. This course equips engineering students with practical mathematical and computational skills to solve real-world problems in modeling, simulation, data science, and automation. Emphasis is placed on numerical integration, differential equations, probability, and optimization techniques implemented using Python.

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

COs	Statements
CO 1	Explaining the fundamental numerical, probabilistic and optimization ideas that underpin scientific computation.
CO 2	Applying ordinary differential equations (ODEs) to model time-dependent engineering systems and simulate their behavior using numerical methods.
CO 3	Analyzing data to infer statistical properties, quantify information and uncertainty, and evaluate hypotheses through appropriate statistical testing methods.
CO 4	Designing and implementing complete Python workflows that solve engineering design, control or data-science problems and document them to professional standards.

Course Outline:

Unit Number: 1	Title: Differential-equation modelling	No. of hours: 15

Content:

- First-order ODEs: separable, exact, linear, Bernoulli; qualitative analysis.
- Second-order constant-coefficient ODEs; damping and forcing.
- Initial-value vs boundary-value problems; stiff systems and implicit Runge–Kutta ideas.
- Brief glimpse of PDE prototypes (heat, wave) to show the continuum.

Hands-on

- Use scipy.integrate.solve_ivp with event detection to simulate spring-mass–damper and RLC circuits.
- Plot phase portraits for a predator–prey (Lotka–Volterra) model and classify equilibria.

Unit Number: 2	Title: Numerical computation & linear algebra	No. of hours: 15

- Floating-point error, stability and conditioning.
- Root-finding: bisection, secant, Newton, fixed-point iteration.
- Numerical differentiation (finite-difference, Richardson) and adaptive integration (Gaussian quad, Romberg).
- Linear systems: LU, Cholesky, Jacobi/Gauss–Seidel; eigen-analysis and singular-value decomposition.
- Dimensionality-reduction with low-rank SVD or PCA.

Hands-on

- Empirically explore derivative error vs step size and illustrate catastrophic cancellation.
- Code Newton and secant methods "from scratch"; benchmark against SciPy.
- Compress a grayscale image to 95 % retained energy via SVD.
- Solve a sparse flow-network problem that appears in logistics or VLSI routing.

Unit Number: 3	Title: Probability, statistics & information theory	No. of hours: 15

Content:

- Random variables and distributions: Bernoulli, Binomial, Poisson, Gaussian, Exponential.
- Law of Large Numbers, Central Limit Theorem, confidence intervals; z, t and χ^2 tests.
- Maximum-likelihood estimation, Bayesian updating, basic Monte-Carlo simulation.
- Entropy, cross-entropy, Kullback–Leibler divergence, mutual information.
- Naïve Bayes; bias-variance trade-off.

Hands-on

- Bootstrap the mean daily output of a wind-farm and build 95 % CIs.
- Train and test a spam–ham Naïve Bayes classifier on an open e-mail corpus.
- Estimate KL divergence between two sensor-noise models and discuss which sensor to deploy.
- Simulate A/B-testing with real-time visualization of p-value trajectories.

Unit Number: 4	**Title: Optimization & data-driven modelling**	**No. of hours: 15**

Content:

- Convex sets/functions; unconstrained minimisers: gradient descent, Newton, BFGS, Adam.
- Constrained problems: Lagrange multipliers, KKT conditions; linear and quadratic programming via CVXPY.
- Linear and logistic regression with L1/L2 regularisation; interpreting coefficients.
- Snapshot of meta-heuristics (genetic algorithms, particle-swarm) and multi-objective Pareto concepts.

Hands-on:

- Fit and cross-validate ridge vs lasso models on an energy-efficiency dataset.
- Formulate a minimum-weight truss design subject to stress limits in CVXPY.
- Visualize the loss landscape of a two-parameter regression and animate gradient-descent paths.
- Use a simple GA to optimize solar-panel tilt angles over a year of irradiance data.

Learning Experiences

- **Classroom Learning**
  - Interactive Lectures: Visual aids and simulations to teach ODEs, numerical methods, and statistical models.
  - Concept Clarity: Step-by-step explanations of topics like Runge–Kutta, KL divergence, and optimization techniques.
  - Live Coding & Demos: Use tools like scipy, cvxpy, and matplotlib to demonstrate real-time problem solving.
  - Group Work & Case Studies: Apply concepts to real-world problems like sensor selection, spam detection, and A/B testing.
  - Problem-Solving Practice: In-class coding and theoretical exercises on differential equations, root-finding, and regression.
- **Outside Classroom Learning**
  - Hands-On Projects: Simulate models like spring–mass systems, heat rods, and Lotka–Volterra equations.
  - Data Analysis Tasks: Train classifiers, apply PCA/SVD, bootstrap datasets, and visualize optimization paths.
  - Collaborative Assignments: Group projects on energy-efficient design, circuit simulations, and routing problems.
  - Self-Learning Resources: Access to notebooks, tutorials, and online discussion platforms for deeper understanding.

Textbooks:

- **Advanced Engineering Mathematics; Erwin Kreyszig;** *Edition:* 10th ; John Wiley & Sons; *ISBN-13 :* 978-0470458365 (10th); 978-1394319466 (11th)
- **Python Programming and Numerical Methods: A Guide for Engineers and Scientists** ;Quingkai Kong, Tim Siauw & Alexandre M. Bayen ;Academic Press, 2021

Data Structures

Program Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Data Structures**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCDS202	3-0-2	4	45
Type of Course:	Major			
Pre-requisite(s):	Basic Programming Knowledge			

Course Perspective. This course provides a comprehensive theoretical foundation in Data Structures and Algorithms, essential for building efficient and scalable software solutions. Students will explore fundamental data structures, understand their underlying principles, and analyze their performance using various complexity measures. With a strong emphasis on concepts relevant to campus placements and real-world industrial applications, the course utilizes Python to illustrate these ideas, enabling practical application of theoretical knowledge without deep language-specific programming.

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

COs	**Statements**
CO 1	**Analyzing** algorithm efficiency and selecting appropriate data structures.
CO 2	**Implementing** and manipulating fundamental linear data structures.
CO 3	**Applying** diverse sorting algorithms and evaluating their performance.
CO 4	**Designing** and utilizing non-linear data structures and hashing techniques.
CO5	**Developing** integrated solutions combining multiple data structures and algorithms

Course Outline:

Unit Number: 1	Title: Foundations & Algorithmic Analysis	No. of hours: 10

Content Summary:

- **Data Structures & ADTs:** Definition, need, common operations.
- **Algorithmic Analysis:** Time & Space Complexity (Big O, Omega, Theta), best/average/worst cases, common complexities (O(1) to O(n!)).
- **Algorithm Design Paradigms:** Brief conceptual overview of Divide & Conquer, Greedy, Dynamic Programming.
- **Recursion:** Concept, Call Stack, Base Case, examples.

Unit Number: 2	Title: **Linear Data Structures**	No. of hours: 10

Content Summary:

- **Arrays:** Definition, 1D/2D, static/dynamic, operations. *Real-world: Basic data storage.*
- **Linked Lists:** Singly, Doubly, Circular; core operations (traversal, insertion, deletion, searching). *Real-world: Dynamic data, browser history.*
- **Stacks (LIFO):** ADT, Push, Pop, Peek. *Real-world: Function calls, expression evaluation.*
- **Queues (FIFO):** ADT, Enqueue, Dequeue. *Real-world: Task scheduling, BFS.*

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

- **Sorting Introduction:** Comparison vs. Non-Comparison, Stable vs. Unstable, In-place vs. Out-of-place.
- **O(N^2) Sorts:** Bubble, Selection, Insertion. (Logic & basic complexity)
- **O(NlogN) Sorts:**
    - **Merge Sort:** Divide & Conquer, Merging, O(NlogN) time, O(N) space, stable.
    - **Quick Sort:** Divide & Conquer, Partitioning (pivot), O(NlogN) average time, O(logN) average space.

- **Heap Sort:** Using Heap structure, O(NlogN) time, O(1) space.

- **Non-Comparison Sorts (Conceptual):** Counting Sort, Radix Sort.

Unit Number: 4	Title: Non-Linear & Advanced Data Structures	No. of hours: 15

Content Summary:

- **Trees:** Terminology, Binary Tree types, Traversals.
- **Binary Search Trees (BSTs):** Properties, core operations.
- **Heaps:** Min/Max Heap properties, Heapify, operations. *Real-world: Priority Queues.*
- **Balanced Trees (Conceptual):** AVL/Red-Black Trees, B-Trees. *Real-world: Database indexing.*
- **Graphs:** Terminology, Representations (Adjacency Matrix/List).
- **Graph Traversal:** BFS and DFS algorithms, applications. *Real-world: Networks, routing.*
- **Hashing:** Concept, Hash Function, Collision Resolution. *Real-world: Hash Maps, Caching.*
- **Tries (Prefix Trees):** Concept, applications. *Real-world: Autocomplete, spell check.*
- **Important Industry & Recent Data Structures (Brief Conceptual Intro):**
    - **Bloom Filters:** Probabilistic set membership.
    - **Spatial Data Structures:** Quadtrees, K-D Trees.
    - **Fenwick Tree / Segment Tree:** Efficient range queries/updates.
    - **Skip Lists:** Probabilistic alternative to balanced trees.

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab Task 1: Algorithmic Efficiency & Recursive Problem Solving** **Problem Statement:** You are tasked with analyzing the efficiency of algorithms and implementing recursive solutions for common computational problems. **Sub-tasks:** **1. Factorial & Fibonacci Analysis:** a. Write a recursive Python function to calculate the factorial of a number n. b. Write a recursive Python function to calculate the nth Fibonacci number. c. For both functions, analyze and write down their time and space complexity using Big O notation. Discuss why one might be less efficient than the other for larger inputs. **2. Tower of Hanoi Implementation:** a. Implement the classic Tower of Hanoi problem using recursion. b. Trace the sequence of moves for N=3 disks. c. Determine the time complexity of the Tower of Hanoi algorithm. **3. Recursive Search Analysis:** a. Consider a recursive implementation of binary search on a sorted list. b. Analyze and justify its time complexity using recurrence relations (conceptual, not formal proof).	CO 1

2	**Lab Task 2: Linear Data Structures: Implementation and Application**
	Problem Statement: Implement and demonstrate the core operations of various linear data structures, understanding their practical implications.
	Sub-tasks:
	• **Dynamic Array Simulation:**
	  ○ Implement a simplified DynamicArray class that mimics Python's list behavior, including append() (handling resizing, e.g., doubling capacity when full) and pop() (from end).
	  ○ Analyze the amortized time complexity of append().
	• **Linked List Operations:**
	  ○ Implement a SinglyLinkedList class with methods for insert_at_beginning(), insert_at_end(), delete_by_value(), and traverse().
	  ○ Extend it to a DoublyLinkedList and implement insert_after_node() and delete_at_position().
	• **Stack & Queue from Scratch:**
	  ○ Implement a Stack ADT using your SinglyLinkedList implementation as the underlying storage. Include push(), pop(), and peek() methods.
	  ○ Implement a Queue ADT using your SinglyLinkedList implementation as the underlying storage. Include enqueue(), dequeue(), and front() methods.
	  ○ Discuss the time complexity of each operation for your linked-list based Stack and Queue.
	• **Application: Parentheses Checker:**
	  ○ Use your Stack implementation to check for balanced parentheses in a given string (e.g., "({[]})").
		CO 2

3	**Lab Task 3: Sorting Algorithms: Performance & Trade-offs** **Problem Statement:** Implement and compare the performance of various sorting algorithms on different datasets to understand their efficiency characteristics. **Sub-tasks:** **1. Implementation of Core Sorts:** a. Implement InsertionSort(). b. Implement MergeSort() (including the merge helper function). c. Implement QuickSort() (choose a partitioning scheme and pivot selection strategy, e.g., last element as pivot). **2. Performance Measurement:** a. Write a utility to measure the execution time of a sorting algorithm. b. Generate datasets: i. A randomly ordered list of 1000, 5000, and 10000 integers. ii. An already sorted list of 1000, 5000, and 10000 integers. iii. A reverse-sorted list of 1000, 5000, and 10000 integers. c. Run each implemented sorting algorithm on these datasets and record the execution times. **3. Analysis and Comparison:** a. Create a brief report (or structured output) comparing the observed performance of InsertionSort, MergeSort, and QuickSort for different input types and sizes.	CO 3

	b. Discuss how the theoretical time complexities (O(N2), O(NlogN)) align with your experimental results, especially noting Quick Sort's worst-case behavior if encountered. c. Comment on the stability and in-place nature of each implemented sort.	
4	**Lab Task 4: Non-Linear Data Structures: Tree, Graph, and Hash Applications** **Problem Statement:** Apply Tree, Graph, and Hashing concepts to build functional components for various data management and search tasks. **Sub-tasks:** • **Binary Search Tree (BST) Operations:** o Implement a BST class with methods for insert(), search(), delete() (handle all cases: no child, one child, two children), and inorder_traversal() (to print sorted elements). o Test your BST with a series of insertions and deletions. • **Graph Traversal:** o Represent a small, directed, weighted graph (e.g., 5-7 nodes, 8-10 edges) using an adjacency list. o Implement Breadth-First Search (BFS) starting from a given node. o Implement Depth-First Search (DFS) starting from a given node. o Print the traversal order for both BFS and DFS. • **Hash Table with Collision Handling:** o Implement a HashTable class using separate chaining for collision resolution. o Include insert(key, value), get(key), and delete(key) methods.	CO 3

	○ Choose a simple hash function (e.g., modulo operator) and demonstrate its usage. ○ Insert several key-value pairs, including some that cause collisions, and demonstrate retrieval and deletion.	
5	**Lab Task 5: Capstone Project: Simple Social Network Explorer** **Problem Statement:** Design and implement a simplified social network explorer that allows users to manage profiles, connect with others, and discover relationships. **Sub-tasks:** **1. User Profile Management (Hashing & Arrays/Lists):** a. Use a HashTable (or Python dictionary internally, conceptually linked to Hashing) to store user profiles, where the user ID (string) is the key and profile details (name, age, interests, etc.) are the value. b. Implement functions to add_user(), get_user_profile(user_id), and update_user_profile(user_id, new_details). **2. Friendship Network (Graphs):** a. Represent the "friendship" or "follower" relationships between users using an Adjacency List (Graph data structure). b. Implement functions to add_friendship(user1_id, user2_id) (bidirectional for friends, unidirectional for followers), remove_friendship(), and get_friends_of_user(user_id). **3. Pathfinding & Relationship Discovery (Graph Traversal):** a. Implement a BFS algorithm to find the "shortest path" (minimum degrees of separation) between two users. Return the path as a list of user IDs.	CO 4

	b. Implement a DFS algorithm to find all "friends of friends" up to a certain depth. **4. Recommendation/Suggestion (Sorting & Trees - Conceptual):** a. **Conceptual Task:** Imagine users have "interest" tags (e.g., ['sports', 'movies', 'tech']). If a system were to recommend new friends, how might you use **Hashing** to find users with similar interests and then **sort** them based on the number of common interests? (No need to implement the full recommendation engine, just discuss the DS/Algo approach.) b. **Optional (Bonus):** If you wanted to quickly find users within a certain geographical proximity (assuming user profiles include coordinates), which advanced data structure (from Unit 4) would be most suitable and why? **5. User Interface (Basic Text-based):** a. Provide a simple text-based interface for interacting with your social network (e.g., command-line prompts to add users, add friendships, find paths, print profiles).	

Learning Experience

Classroom Learning

- **Concept Building:** Interactive lectures on arrays, linked lists, stacks, trees, and graphs with real-life examples.
- **Visual Learning:** Use flowcharts and recursion trees to explain sorting and searching algorithms.
- **In-Class Coding:** Live Python demos of sorting algorithms, recursion, and basic data structures.
- **Problem Solving:** Weekly quizzes, dry runs, and time-complexity analysis of common algorithms.

Lab-Based Learning

- **Hands-on Practice:** Implement linear/non-linear structures (e.g., Stack, Queue, BST, HashTable) from scratch.
- **Recursive Thinking:** Solve problems like Fibonacci, Tower of Hanoi, Binary Search using recursion.
- **Algorithm Comparison:** Measure and compare sorting algorithms on different datasets.
- **Capstone Project:** Build a mini social network system integrating hashing, graphs, and traversals.

Beyond Classroom

- **Assignments & Challenges:** Weekly coding tasks and mini-projects for applying concepts.
- **Collaborative Learning:** Group discussions, peer code reviews, and code-along sessions.
- **Online Practice:** Practice problems on coding platforms to strengthen placement readiness.

Text Book:

Data Structures & Algorithms in Python; Author: Fabio Neves; **Publisher:** Packt Publishing; **Publication Year:** 2021; **ISBN-13:** 978-1801815170

Web Dev -II (Advanced JS & React)

Program Name	B.Tech (CSE) with specialization in UI/UX			
Course Name: Web Dev -II (Advanced JS & React)	Course Code	L-T-P	Credits	Contact Hours
	ETCCWD203	3-0-2	4	45
Type of Course:	Major			

Course Perspective. In this second-level web course you will master modern JavaScript features and build full single-page applications with React. You will learn ES6+ syntax, asynchronous programming, testing, and tooling, then dive into React's component model, hooks, state management, routing, performance tuning, and deployment. Every concept is paired with live coding and mini-projects, so by the end you will have a production-ready React app and the skills for a junior React / Front-End Engineer role.

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

COs	Statements
CO 1	Applying advanced ES6+ JavaScript syntax, modules, and async patterns in real projects.
CO 2	Building reactive user interfaces with React functional components and hooks.
CO 3	Managing complex state, routing, and data-fetching in single-page applications.
CO 4	Optimizing, test, and deploy React apps using industry tools and CI/CD workflows.
CO5	Producing a portfolio-quality SPA that follows accessibility and performance best practices

Course Outline:

Unit Number: 1	Title: Modern JavaScript Deep-Dive	No. of hours: 12

- ES6+ essentials: let/const, template literals, destructuring, default & rest parameters, spread syntax.
- Advanced functions: arrow functions nuances, higher-order functions, closures, currying.
- Modules & tooling: ES modules, import/export, bundlers (Vite/Webpack basics), linting with ESLint/Prettier.
- Asynchronous JS: Promises, async/await, error handling, fetch & AbortController, parallel vs sequential flows.
- Testing fundamentals with Jest & mocking fetch calls.

Hands-on focus: daily kata converting legacy ES5 code to ES6+, writing async utilities, and green-bar tests.

Unit Number: 2	Title: **React Fundamentals & Component Architecture**	No. of hours: 10

Content Summary:

- Setting up with Vite/Create-React-App, project structure, hot reloading.
- JSX syntax and rendering rules.
- Functional vs class components; props, state, event handling.
- Core hooks: useState, useEffect, useRef, useMemo for simple caching.
- Styling strategies: CSS Modules, Styled-Components, Tailwind overview.
- Component communication patterns (prop drilling vs composition).
- **Hands-on focus:** build a multi-component UI (dashboard, cards, modal) and refactor class → function + hooks.

Unit Number: 3	Title: State Management, Routing & Advanced Hooks	No. of hours: 12

- Global state with Context API and useReducer.
- Custom hooks for reuse (form, fetch, pagination).
- React Router v6: nested routes, dynamic params, protected routes.

- Data fetching libraries: React Query or SWR for caching & mutations.
- Intro to Redux Toolkit or Zustand for larger apps.
- Auth flow basics (JWT storage, guarded routes).

Hands-on focus: integrate REST API, build forms with React-Hook-Form, and implement context-based dark/light theme switcher.

Unit Number: 4	Title: Performance, Testing & Deployment	No. of hours: 11

Content Summary:

- Performance profiling: React DevTools, flame-charts, why-did-you-render.
- Optimisation: memoisation (React.memo, useCallback, useMemo), code-splitting with React.lazy, image optimisation.
- Accessibility in React: semantic HTML, ARIA, keyboard traps, Lighthouse checks.
- Testing React UI: React Testing Library, snapshot vs behaviour testing, CI with GitHub Actions.
- Build & deploy: environment variables, static hosting (Netlify/Vercel), basic PWA setup, versioned releases.

Hands-on focus: measure and improve bundle size; set up automated tests and a deploy pipeline.

Lab Experiments

1. **Lab 1 (Unit 1) – "ES6+ Portfolio Generator CLI"** **CO1**
    - Scaffold a Node-based CLI that reads JSON resume data.
    - Use ES modules and async file I/O to generate static HTML.
    - Implement CLI flags for theme selection.
    - Add unit tests for the generator functions.
    - Publish the package to a private GitHub repo with README usage guide.
2. **Lab 2 (Unit 2) – "Personal Finance Tracker – Core UI"** **CO2**
    - Spin up a React project with Vite.

- Design dashboard, transaction list, and add-expense modal using functional components.
- Manage component-level state with useState.
- Persist dummy data in localStorage.
- Apply responsive styling with Tailwind or CSS Modules.

3. **Lab 3 (Unit 3) – "Real-Time Chat App with Context & Routing" CO3**
   - Create channel list and chat window routes with React Router.
   - Store current user and messages in Context + useReducer.
   - Build a custom hook for WebSocket simulation (polling JSON).
   - Add login form using React-Hook-Form and guarded routes.
   - Display unread badges, theme switch, and optimistic UI updates.
4. **Lab 4 (Unit 4) – "E-Commerce SPA Performance Audit" CO4**
   - Fork a starter React store and benchmark bundle size.
   - Implement code-splitting, lazy images, and memoised product cards.
   - Write accessibility tests (alt text, tab order) and unit tests for cart logic.
   - Configure GitHub Actions to run tests and deploy preview build to Netlify.
   - Document improvements and before/after metrics in README.
5. **Lab 5 – Capstone (All Units) – "Smart-City Events Dashboard" CO4**
   - Plan feature list and Kanban board (GitHub Projects).
   - Fetch real event data from a public API; cache requests with React Query.
   - Implement map view, search & filters, favourites stored in Context.
   - Optimise with code-splitting, PWA manifest, and service worker offline support.
   - Deploy to Vercel and present Lighthouse scores ≥ 90 for performance & accessibility.

Learning Experience

Classroom Learning

- Concept-Oriented Live Coding: Teach ES6+ features, React components, hooks, and state using real-time demos.
- Project-Based Learning: Build small UIs, implement routing, context, and API integration in class.
- Performance & Deployment: Cover optimisation techniques, testing with React Testing Library, and deploying to Vercel/Netlify.

Lab-Based Learning

- ES6+ CLI generator, React UI for finance tracker, chat app with context & routing.
- Optimise and test an e-commerce SPA; capstone project on Smart-City dashboard with PWA features.
- Tools Used: React, Vite, Tailwind, ESLint, GitHub Actions, Lighthouse.

Beyond Classroom

- Portfolio Projects: Guide students to deploy and showcase apps.
- Peer Learning: Code reviews, Git workflows, and team collaboration.
- Industry Readiness: Practice CI/CD, real APIs, and accessibility standards.

Text book:

- **Learning React: Modern Patterns for Developing React Apps;** Alex Banks & Eve Porcello; O'Reilly Media; 5th Edition, 2024 ; **ISBN-13:** 978-1098132924

Engineering Physics

Program Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Engineering Physics**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCPH204	2-0-2	3	30
Type of Course:	Major			

Course Perspective. This course is designed to provide B.Tech CSE students with a fundamental understanding of key physics principles that drive modern computing and communication technologies. The course focuses on practical applications of semiconductor physics and optics, ensuring students grasp how these concepts are integrated into real-world computing devices like processors, memory units, fiber-optic networks, and display technologies. By emphasizing applied learning over theoretical derivations, students will develop an intuitive understanding of physics-driven advancements in computer science and electronics.

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

COs	**Statements**
CO 1	Understanding fundamental concepts of optics and semiconductor physics relevant to computing technologies.
CO 2	Applying the principles of semiconductors and optical communication in understanding the working of modern electronic and computing devices.
CO 3	Analyzing the role of semiconductor materials and optical components in computing and data storage systems.
CO 4	Evaluating different semiconductor devices and optical systems to assess their impact on computing advancements.

Course Outline:

Unit Number: 1	Title: Semiconductor Physics for Electronic and Computing Devices	No. of hours: 10

Content Summary:

- **Basics of Semiconductors:** (Qualitative Analysis) Introduction to semiconductors, Concept of Energy bands, intrinsic & extrinsic semiconductors, carrier concentration, temperature dependence
- **PN Junction Diode:** (Qualitative Analysis) Working principle, Formation of depletion region, I-V characteristics, applications in LEDs, photodiodes, and solar cells
- **Basics of Transistors:** (Qualitative Analysis) Introduction to transistors, construction and working of transistor, Biasing in transistors. Types of transistors: BJT and MOSFET, simple applications in electronic circuits (concept of amplification and switching)

Unit Number: 2	Title: Physics in Everyday Computing Devices	No. of hours: 6

Content Summary:

- **Display Technologies:** (Qualitative Analysis) Principles and working of CRT, LCD, LED, and OLED screens, comparison of technologies
- **Sensors in Computing:** (Qualitative Analysis) Working principles and applications of touchscreens, fingerprint sensors, and biometric sensors

Unit Number: 3	Title: Optical Physics in Modern Computing	No. of hours: 10

- **Interference and Diffraction:** Basic Principles of interference and diffraction and their types, concept of path and phase difference, Conditions of obtaining bright and dark pattern in interference and diffraction (Qualitative Analysis) and their applications in CD/DVD storage.
- **Holography:** (Qualitative Analysis) Basic Principles of holography, working of holographic storage (Recording and reconstruction of hologram), applications in data security and 3D imaging.
- **Lasers in Computing:** (Qualitative Analysis) Introduction to Lasers, Its properties, principles of laser operation, applications in barcode scanners, optical storage, and laser printers.

Unit Number: 4	Title: Optical Communication and Fiber Optics	No. of hours: 4

Content Summary:

- **Optical Fibers:** (Qualitative Analysis) Structure, working principle, total internal reflection, Basic Terminologies-Critical angle, Acceptance Angle, Acceptance Cone, Numerical Aperture.
- **Fiber Optic Communication:** Role of optical fibres in internet and data transmission, advantages over traditional cabling.

Lab Experiments

ExNo.	Experiment Title	Mapped CO
1.	To determine the wavelength of sodium light using Newton`s ring apparatus.	**CO1**
2.	To determine the wavelength of prominent lines of mercury by plane diffraction grating.	**CO1**
3.	To determine the refractive index of the material of the prism for the given colours (wavelengths) of mercury light with the help of spectrometer.	**CO2**
4.	To determine the specific rotation of cane sugar solution with the help of half shade polarimeter.	**CO3**
5.	To determine the wavelength of He-Ne LASER using transmission diffraction grating.	**CO3**
6.	To study the forward and reverse bias characteristics of a PN junction diode.	**CO2**
7.	To investigate the voltage regulation properties of a Zener diode in breakdown mode.	**CO3**
8.	To study the Input and Output Characteristics of an NPN Transistor (CE Configuration).	**CO4**
9.	To determine the moment of inertia of a flywheel about its own axis of motion.	**CO2**

10.	To determine the value of acceleration due to gravity using Kater`s pendulum.	**CO4**
11.	To plot a graph between the distance of the knife edge from the centre of gravity and the time of the bar pendulum. From the graph, find the acceleration due to gravity and the radius of gyration of the bar.	**CO4**

Learning Experience

Classroom Learning

- Teach semiconductor devices, optics, and lasers through real-world tech applications (LEDs, sensors, fiber internet).
- Use diagrams, simulations, and demo videos to explain concepts like interference, diffraction, and transistor working.
- Reinforce understanding with short quizzes, problem-solving, and concept-based discussions.

Lab-Based Learning

- Conduct hands-on experiments with optical instruments, diodes, transistors, and pendulums.
- Emphasize accurate observation, graph plotting, and practical verification of theory.
- Develop technical handling and data interpretation skills through guided lab work.

Beyond Classroom Learning

1. Assign mini research or presentation topics (e.g., OLEDs, biometric sensors, holography).
2. Encourage use of virtual labs and simulations for revision and deeper exploration.
3. Promote participation in science clubs, quizzes, and visits to tech labs or industries.

Text books/References:

1. S.O. Kasap – *Principles of Electronic Materials and Devices*, 4th Edition, McGraw Hill Education, 2018, 4th edition
2. Arthur Beiser – Concepts of Modern Physics, 7th Edition, McGraw Hill Education (India), 2017
3. Ajoy Ghatak – Optics, 6th Edition, McGraw Hill Education (India), 2017

Minor Project-I

Program Name	B.Tech (CSE) with specialization in UI/UX		
COURSE NAME: Minor Project-I	Course Code	L-T-P	Credits
	ETCCPR205	0-0-4	2
TYPE OF COURSE:	Project		
PRE-REQUISITE(S), IF ANY: NA			

Course Perspective:

The objective of Minor Project-I for the B. Tech (Computer Science & Engineering) 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. Minor project should encourage 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: 12-16 weeks.

Project must focus on following aspects:

Standard Operating Procedure (SOP)

1. **Purpose**

Minor Project-I immerses second-year students in problem discovery, research, and critical analysis of a real societal challenge addressable via computing. From the 2025-26 session onward, every step—topic selection, mentoring, submission, feedback, grading, and reporting—will be executed and audited inside Projexa.

This SOP unifies academic requirements with Projexa's digital workflow to guarantee transparency, consistency, and accreditation-ready records.

2. Scope

- Applies to: All B.Tech CSE students registered for Minor Project-I.
- Excludes: Implementation-heavy Minor Project-II (covered by a separate SOP).
- Duration: 12–14 teaching weeks (one full semester block).

3. Learning Outcomes (LO)

LO	Student will be able to…	Evidence Captured by Projexa
LO-1	Identify a specific, socially relevant computing problem	"Problem Synopsis" form
LO-2	Conduct & critique a structured literature review	Lit-Review PDF + Gap-Matrix worksheet
LO-3	Analyse & synthesize existing solutions, exposing gaps	Mid-term viva recording + mentor comments
LO-4	Document & present findings in professional formats	Auto-generated tech report + slide decks
LO-5	Operate a project-management platform ethically & professionally	Timestamped activity log, on-time submissions

4. Projexa: Core Functions Used in Minor Project-I

Module	**Purpose**
Team Workspace	Topic discussion, mentor chat, file vault
Milestone Engine	Proposal → Mentor Approval → Mid-Review → Final Review
Rubric Builder	Digitised grading templates for mentor & PEC
Analytics Dashboard	Real-time progress, CO/PO attainment, mentor load

Integrity Ledger	Log of late submissions, plagiarism flags, change requests

5. Roles & Responsibilities

Role	Key Responsibilities	Projexa Permissions
Student Team (2–4)	Draft synopsis, upload artefacts, attend vivas, act on feedback, complete reflection survey	Create files, comment, view deadlines
Project Mentor (Faculty)	Weekly guidance, approve milestones, grade mentor components, impose ±3 effort modifier	Approve/Reject, rubric scoring, notes
Project Evaluation Committee (PEC) (3 faculty including Coordinator)	Evaluate Synopsis, Mid-term, Final; moderate mentor marks; resolve disputes	Rubric scoring, moderation tools
Project Coordinator	Configure rubrics & deadlines, monitor cohorts, author reports, manage change-requests	Admin dashboard, deadline override
Dept. Admin	Oversight, accreditation data exports, technical ticket escalation	Read-only analytics, export

6. Semester Timeline (12–14 Weeks)

Week	Status Change in Projexa	Student Deliverable	Mentor / PEC Action
0	Team formation	—	Verify teams
1	Draft → Submitted	1-slide Idea Pitch	Feasibility comment
2	Draft → Submitted	Problem Synopsis (2 pp)	Phase A rubric (Mentor 5 / PEC 15)
3	Mentor-Approved	Revised synopsis (if required)	Mark "Synopsis Approved"

4	—	Literature-Review Dossier + Gap-Matrix	Inline feedback
5	—	Logbook entries	Progress check
6	Mid-Review	Mid-term Deck + Viva	Phase B rubric (Mentor 10 / PEC 20)
7–8	—	Deep-dive analysis, data gathering	Weekly mentor comments
9	—	Draft Tech Report	Mentor inline edits
10	Mentor-Approved	Revised draft report	Set "Ready for Final" flag
11	—	Demo video (≤3 min) rehearsal	Dry-run feedback
12	Final Review	Final Deck + Report	Phase C rubric (Mentor 10 / PEC 30)
13	—	Scholarly evidence (paper / competition)	Phase D score (0–10)
14	Closed	Reflection survey	Grade release, closure

7. Deliverables & Format Standards

Artefact	Mandatory Format	Upload Location
Idea Pitch	1-slide PDF	Proposal module
Problem Synopsis	PDF (Dept template)	Proposal module
Literature Review	PDF + XLS Gap-Matrix	Docs upload
Mid-term Slides	PPT/PDF (12–15 slides)	Presentation module
Logbook	Auto-captured	Activity Log
Draft & Final Report	IEEE 2-column PDF (8–10 pp)	Report upload
Demo Video	MP4 link (YouTube Unlisted / Drive)	Media tab
Scholarly Output	PDF of submission or award certificate	Evidence upload

8. Evaluation Scheme (100 Marks)

Phase	Timing	Total	Mentor	PEC	Criteria (digital rubric)
A Synopsis	Week 2-3	20	5	15	Problem relevance, objective clarity, presentation quality, Q&A
B Mid-term	Week 6	30	10	20	Lit-review depth, gap analysis, methodology soundness, modern tools, logbook rigour
C Final	Week 12	40	10	30	Findings vs objectives, societal impact, report quality, viva professionalism
D Scholarly / Outreach	Week 13	10	—	10	5 pts for manuscript/competition entry +5 bonus for acceptance/award (max 10)
Continuous Effort Modifier	Whole semester	±3	Mentor	—	Consistent diligence (+) or chronic non-compliance (–)

Rubrics contain 4 performance levels (Excellent, Good, Satisfactory, Poor); descriptors are stored in Projexa's Rubric Builder.

9. Grading & Publication

1. Weighted calculation auto-executes when PEC submits final rubric.
2. Students see marks & comments but cannot edit rubrics.
3. A random 10 % sample is second-marked by another PEC member for moderation.
4. Pass requirement: ≥ 50 % overall AND ≥ 40 % in each Phase A-C.
5. Grades are posted to the LMS via Projexa API within 72 h of final review.

Maker Lab: Tinkering with Technology

Program Name	B.Tech (CSE) with specialization in UI/UX			
Course Name: **Maker Lab: Tinkering with Technology**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		1-0-2	2	15
Type of Course:	SEC			

Course Perspective. This hands-on course equips students with practical skills in electronics prototyping, embedded systems, sensor integration, PCB design, additive and subtractive manufacturing, and wireless communication. By completing guided tasks and a system-level build, students develop end-to-end product development capabilities — from circuit design to testing and deployment fostering readiness for maker spaces, startups, and hardware engineering roles.

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

COs	Statements
CO 1	**Demonstrating** safe handling of lab tools and using core electronics instruments.
CO 2	**Designing,** simulating, and fabricating sensor-integrated circuits and enclosures.
CO 3	**Implementing** actuation and wireless communication in embedded prototypes.
CO 4	**Evaluating** system performance, **optimizing** hardware design, and **documenting** a manufacturable prototype using industry-standard tools and practices.

Course Outline:

Unit Number: 1	Title: Bench Foundations & Safe Circuits	No. of hours: 4

Content Summary:

- PPE, fume-extraction zones, ESD protocol, Li-ion handling.
- Ohm's law review, breadboard hygiene, interpreting datasheets.
- Multimeter, bench PSU & oscilloscope fundamentals.
- Git/GitHub essentials for hardware: repo structure, committing code + CAD.

Guided tasks

1. Safety passport practical – Identify five hazards in the shop; demonstrate correct use of goggles, fume extractor and antistatic mat.

2. Blink-&-Measure

   - Wire LED + buzzer on an Arduino Nano 33 IoT.
   - Program SOS blink in C++; push commit *blink_v1* to GitHub.
   - Measure supply current with Fluke 117; record values in README table.

3. Instrument simulation – Replicate the circuit in TinkerCAD Circuits; capture a screenshot showing virtual ammeter reading.

Deliverables

- Signed safety passport card.
- GitHub repo with blink_v1 code, README table of current values and simulation screenshot.

Unit Number: 2	Title: Sensors, PCB Design & Additive Fabrication	No. of hours: 3

Content Summary:

- Analogue vs digital sensor interfacing; voltage dividers & level shifting.
- KiCad 8 workflow: schematic → footprint → layout → Gerber.
- Power-budget spreadsheet (voltage, peak current, thermal limits).
- FDM vs SLA printing; slicer parameters; laser-cut acrylic techniques.

Guided tasks

1. Sensor characterization – Wire DS18B20 thermometer, log 30 samples to Serial; export CSV.

2. Breakout PCB
    - Draw schematic in KiCad; add decoupling cap & pull-up resistor.
    - Route 2-layer board ≤ 50 mm × 25 mm; run DRC; generate Gerbers.
    - Mill board on Othermill; reflow in IR oven; smoke test at 5 V.
3. Smart-Temp Logger Enclosure
    - Model vented case in Fusion 360 with snap-fits and cable gland.
    - Export STL, slice in PrusaSlicer; print on Prusa MK4.
    - Laser-cut clear acrylic lid; tap holes, add brass inserts; final assembly.
4. Calibration & log – Place logger in fridge and room temp; log for 10 min each; plot comparison in LibreOffice Calc.

Deliverables

- KiCad project folder in GitHub.
- Photos of assembled PCB + printed case.
- CSV + temp-comparison chart embedded in README.

Unit Number: 3	**Title: Actuation, Wireless Comms & Subtractive CAM**	**No. of hours: 4**

Content Summary:

- H-bridge drivers: L298N vs DRV8833.
- PWM & PID: Ziegler–Nichols tuning method.
- BLE vs Wi-Fi, OTA basics on ESP32.
- CAM workflow: Fusion 360 tool-paths, feeds/speeds for Delrin.

Guided tasks

1. Gear-set machining – Design 30-tooth Delrin spur gear in Fusion 360; CAM & cut on Bantam Tools mill; measure backlash < 0.3 mm.
2. BLE Rover chassis
    - 3-D-print chassis body; vinyl-cut panel graphics; apply.
    - Mount motors, gear set, L298N driver; secure battery pack.

3. ESP32 joystick control – Program BLE GATT server; phone app (nRF Connect) as joystick; tune PID to keep rover heading error < 5 cm over 1 m.
4. Telemetry log – Stream speed & battery voltage to serial; save 1-minute log to SD.

Deliverables

- Fusion 360 CAM file and machined gear photo.
- BLE Rover video showing straight-line run & joystick turns.
- PID parameter sheet + telemetry CSV in repo.

Unit Number: 4	Title: System Test, Optimization & Handoff	No. of hours: 4

Content Summary:

- Automated tests with Unity + PlatformIO; GitHub Actions CI.
- Thermal & current profiling; mean-time-between-failure estimate.
- Generative design weight-reduction in Fusion 360.
- CE/UL basics, RoHS checklists; cost-of-goods & sustainability hotspots.

Guided tasks

1. Firmware test-suite – Write three unit tests (sensor, motor direction, BLE packet) and add CI badge to repo.
2. Burn-in & log – Run rover 30 min at 1.2× nominal load; log temp & current every 5 s; generate Python plot.
3. Lightweight chassis – Apply Fusion 360 Generative Design to cut ≥ 10 % mass; print, swap, re-test.
4. Manufacturing pack – Compile Gerbers, STLs, CAM, wiring loom PDF and costed BOM spreadsheet.
5. Poster & pitch deck – One A1 poster and five-slide deck with metrics, cost, and sustainability notes.

Deliverables

- CI-passing GitHub repo with burn-in log plot.
- Generative design comparison (before/after mass, image).

- Manufacturing ZIP and poster/deck uploaded to LMS

Key Software / Hardware Stack

- Arduino IDE 2.x, PlatformIO, KiCad 8, Fusion 360 (Edu CAM), PrusaSlicer / Cura, Git & GitHub Desktop, Python 3.x (Jupyter for data plots), Saleae Logic (Free tier).
- Hardware kits: Arduino Nano 33 IoT, ESP32-DevKitC, motor driver boards, sensor assortment, Li-ion pack + BMS, desktop FDM printer, 3018-class CNC, vinyl cutter.

Assessment & Weighting

Component	Weight
Weekly Shop Check-offs & Git commits	20 %
PCB + Enclosure Sub-assembly (Unit 2 deliverable)	20 %
Robot / Mechanism Functional Demo (Unit 3)	20 %
Final Integrated Product, Engineering Brief & Viva	40 %

Pass Rule – Prototype must operate for ≥ 30 min continuous without fault and all design files must be committed to GitHub with a tagged release

Lab Experiments

Lab Task 1 – "SOS Desk-Beacon" ***(maps to Unit 1: Ideation, Circuits & Visualization)***

Sub-tasks

- Select an appropriate Arduino family board (Uno / Nano / Pro Mini) and justify choice in a log.

- Breadboard an LED-buzzer circuit with a push-button; program it to flash **SOS** in Morse.
- Stream button-press counts to the Serial Monitor and capture a 30-second CSV trace.
- Replicate the circuit in **TinkerCAD Circuits**, verify current draw warnings.
- Create a two-view orthographic sketch of the breadboard layout (top & side) in AutoCAD (or TinkerCAD sketch).

Outcome → students touch every Unit-1 topic: board selection, basic I/O, serial diagnostics, simulation, 2-D visualization.

Lab Task 2 – "Smart-Temp Logger Enclosure" *(maps to Unit 2: Sensor Integration & 3-D Modelling)*

Sub-tasks

- Wire and calibrate a temperature sensor (DS18B20 or LM35) on Arduino; log ten readings.
- Model a snap-fit sensor holder + vented enclosure in **Fusion 360** (≤ 80 × 60 × 40 mm).
- Export STL, slice in **Cura**, print the enclosure; measure tolerance error ≤ 0.3 mm.
- Design a matching acrylic lid in 2-D, laser-cut it, and assemble with threaded inserts.
- Embed the calibrated sensor, rerun logging, and compare ambient vs enclosed temp (plot in LibreOffice).

Outcome → students integrate sensors, CAD, 3-D printing and laser cutting, satisfying all Unit-2 skills.

Lab Task 3 – "BLE Rover Chassis" *(maps to Unit 3: Actuation, Communication & Assembly)*

Sub-tasks

- CAD-adjust Unit-2 enclosure into a two-wheel differential-drive chassis; add servo mount for sensor mast.

- Control two DC motors via **L298N**; tune PWM to keep straight-line error < 5 cm over 1 m.
- Pair an **ESP32** to a smartphone app (nRF Connect) and stream joystick commands over BLE.
- Add an ultrasonic sensor on a servo; sweep 90°, send distance data back over BLE.
- Assemble printed parts, ensure wiring strain-relief and pass a 5-minute bench-safety checklist.

Outcome → full use of actuators, wireless comms, motion-part CAD and mechanical fitment.

Lab Task 4 – "Debug & Optimise Sprint" *(maps to Unit 4: System Integration, Testing & Demo)*

Sub-tasks

- Instrument BLE Rover with a logic-analyser pin and current-shunt; log a 1-minute run.
- Identify one bottleneck (e.g., loop latency, motor stall current) and apply an optimisation (PROGMEM tables, PWM ramp).
- Run a 30-minute burn-in; record temperature and battery voltage, produce a validation graph.
- Apply **basic generative-design** in Fusion 360 to lighten the chassis by ≥ 10 %, re-print, re-test.
- Prepare a poster: block diagram, data charts, costed BOM, and a QR-coded demo video.

Outcome → students practise structured debugging, data validation, performance tuning, generative redesign, and presentation.

Lab Task 5 – Capstone – "Campus Courier Bot" *(integrates Units 1 ↔ 4)*

Sub-tasks

1. **Concept & Requirements** – define payload (< 1 kg), route length, safety criteria; commit a GitHub project board.

2. **Electro-Mechanical Build** – design a custom PCB shield, integrate sensors (IMU, ToF, line-follow array) and motor drivers; 3-D-print a modular chassis with quick-swap battery tray.
3. **Comms & Control** – implement Wi-Fi MQTT for waypoint commands and BLE fallback; OTA update workflow.
4. **Testing & Metrics** – achieve: speed ≥ 0.4 m/s, obstacle-avoid success ≥ 90 %, runtime ≥ 45 min; provide plots and logs.
5. **Manufacturing Pack & Demo** – submit Gerbers, STLs, CAM files, annotated code, BOM with cost & sustainability notes; deliver a live delivery run and a 5-minute pitch.

Learning Experience

Classroom Based Learning

- Short theory sessions on safety, sensor interfacing, PCB workflows, communication protocols, and testing tools.
- Introduce key platforms like Arduino, ESP32, KiCad, Fusion 360, and GitHub.

Lab-Based Learning

- Perform structured tasks such as blinking circuits, logging sensor data, PCB fabrication, and BLE rover control.
- Apply design-for-manufacture thinking through CAD, CAM, and enclosure modelling.
- Use tools like multimeters, 3D printers, laser cutters, CNC mills, and thermal cameras.

Beyond Classroom

- Students maintain GitHub repos for version control and documentation.
- Encourage exploration of TinkerCAD, PrusaSlicer, and Unity test framework outside lab hours.
- Promote peer reviews, demo days, and optional project showcases to simulate real-world maker culture.

Textbooks/References:

1. Simon Monk – Programming Arduino: Getting Started with Sketches, McGraw Hill, 2nd Edition, 2016.ISBN: 9781259641633
2. Charles Platt – Make: Electronics: Learning Through Discovery, Maker Media, 2nd Edition, 2015.ISBN: 9781680450262
3. Matthew Scarpino – KiCad Like a Pro: Learn the World's Favourite Open Source PCB Design Tool, Elektor, 2nd Edition, 2020. ISBN: 9783895764470
4. Jennifer Herron – Fusion 360 for Makers: Design Your Own Digital Models for 3D Printing and CNC Fabrication, Maker Media, 1st Edition, 2020. ISBN: 9781680455236

Open Elective-I

Course 1: AI & Machine Learning for Everyone

Program Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Open Elective-I(AI & Machine Learning for Everyone)**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		3-0-0	3	45
Type of Course:	OEC			

Course Perspective. This foundational course introduces the core concepts of Artificial Intelligence (AI) and Machine Learning (ML) to a non-technical audience. Students explore how AI works, what ML algorithms do, and how these technologies impact society, businesses, and personal lives. The course emphasizes real-world applications, responsible AI practices, and intuitive understanding without programming or mathematical prerequisites.

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

COs	**Statements**
CO1	Understanding the fundamental ideas behind AI and machine learning in everyday terms.
CO2	Identifying key application areas and case studies where AI/ML is being used.
CO3	**Applying** intuitive understanding of how data, algorithms, and learning models interact in real-world AI/ML scenarios
CO4	Evaluating the social, ethical, and economic implications of AI systems
CO5	Critically **assessing** AI tools in media, workplace, and society

Course Outline:

Unit Number: 1	Title: AI Foundations & History	No. of hours: 12
Content Summary: • What is AI? Historical timeline and major milestones • Types of AI: Reactive, limited memory, general, and superintelligence • Machine learning vs. traditional programming • The role of data in AI: examples from healthcare, finance, entertainment		
Unit Number: 2	**Title: Machine Learning Intuition**	**No. of hours: 13**
Content Summary: • Supervised vs. unsupervised learning • Introduction to decision trees, clustering, and neural networks (conceptual only) • What makes a machine "learn"? Importance of training and testing • Use-cases: email spam filters, movie recommendations, face recognition		
Unit Number: 3	**Title: AI in the Real World**	**No. of hours: 10**
Content Summary: • AI in business: customer analytics, predictive maintenance, chatbots • AI in daily life: smart assistants, cameras, GPS, language translation • Government and education applications of AI • Limits of current AI technologies		
Unit Number: 4	**Title: Responsible & Ethical AI**	**No. of hours: 10**
Content Summary: • Bias in AI and how it affects real people • Privacy and surveillance concerns • Explainable AI: Can AI decisions be trusted? • The future of work and AI's societal impact		

Learning Experience

Classroom Learning

- Interactive Lectures: Use storytelling, real-life analogies, and multimedia to explain AI/ML concepts in simple terms.

- Case Study Discussions: Analyze current AI applications (e.g., Netflix recommendations, ChatGPT, facial recognition).
- Ethics Debates: Engage students in structured debates on AI bias, privacy, and job displacement.
- Mini Quizzes & Polls: Use Kahoot/Mentimeter for real-time feedback and conceptual reinforcement.

Hands-On & Beyond Classroom

- No-code Tools Exploration: Allow students to explore AI demos using platforms like Teachable Machine, Lobe.ai, or Google's AI Experiments.
- Media Literacy Tasks: Assign students to analyze news/media reports on AI critically.
- Micro-Projects: Create posters or short presentations on AI in healthcare, education, or governance.
- Group Activity: Simulate how an AI system learns using role-play exercises (e.g., students act as decision trees or neural networks).

Textbook: Mitchell, M. (2019). *Artificial Intelligence: A Guide for Thinking Humans*. Farrar, Straus and Giroux. ISBN: 978-0374257835

Additional References:

- Dignum, V. (2019). *Responsible Artificial Intelligence: How to Develop and Use AI in a Responsible Way*. Springer. ISBN: 978-3030303709
- Domingos, P. (2015). *The Master Algorithm: How the Quest for the Ultimate Learning Machine Will Remake Our World*. Basic Books. ISBN: 978-0465065707

Open Elective-I

Course 2: Digital Transformation & Industry 4.0

Program Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Open Elective-I(Digital Transformation & Industry 4.0)**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		3-0-0	3	45
Type of Course:	OEC			

Course Perspective. This course explores the principles and technologies driving the fourth industrial revolution—Industry 4.0. Students learn how digital transformation is reshaping manufacturing, services, and organizational workflows through automation, cyber-physical systems, and data-driven strategies. Key focus areas include IoT, smart factories, AI integration, and the organizational change required to implement digital transformation effectively.

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

COs	**Statements**
CO1	Understanding the evolution and core principles of Industry 4.0.
CO2	Analyzing the role of digital technologies in modernizing business and industrial processes.
CO3	Identifying key components of smart manufacturing such as IoT, cloud, and AI.
CO4	Evaluating the challenges and strategies for implementing digital transformation.
CO5	Examining real-world applications and future trends in Industry 4.0.

Course Outline:

Unit Number: 1	Title: Cybersecurity Fundamentals	No. of hours: 12

Content Summary:

- What is cybersecurity? Scope and importance
- Key concepts: confidentiality, integrity, availability (CIA Triad)
- Overview of threats: malware, phishing, ransomware, social engineering
- Anatomy of a cyber-attack: how breaches happen

Unit Number: 2	Title: Securing Devices and Networks	No. of hours: 13

Content Summary:

- Secure passwords and multi-factor authentication (MFA)
- Network security basics: firewalls, VPNs, antivirus software
- Device hardening: OS updates, secure settings, mobile security
- Safe browsing and email hygiene

Unit Number: 3	Title: Digital Privacy and Data Protection	No. of hours: 10

Content Summary:

- Understanding personal data: what's collected and why
- Cookies, trackers, and surveillance capitalism
- Data protection laws: GDPR, IT Act (India), and global perspectives
- Managing online footprints and digital identity

Unit Number: 4	Title: Ethics, Policies, and Future Risks	No. of hours: 10

Content Summary:

- Ethical hacking and cybersecurity careers
- Responsible disclosure and bug bounty programs
- Emerging risks: deepfakes, IoT, AI-powered attacks
- Case studies: data breaches and lessons learned

Learning Experience

In-Class Learning

- Conceptual lectures with real-world case studies on Industry 4.0.
- Group discussions on digital disruption and smart technologies.

- Guest talks or video sessions from industry professionals.

Beyond Classroom

- Explore live demos of IoT or digital twin tools.
- Mini-projects to create basic digital transformation plans.
- Student reflections on automation's impact on careers.
- Track and present updates on emerging Industry 4.0 trends.

Textbook: Grimes, R. A. (2019). *Cybersecurity Basics: A Hands-On Approach*. Packt Publishing. ISBN: 978-1838827846

Additional References:

- Schneier, B. (2015). *Data and Goliath: The Hidden Battles to Collect Your Data and Control Your World*. W.W. Norton & Company. ISBN: 978-0393352177
- Andress, J. (2019). *Cybersecurity and Privacy: An Introduction for Non-Technical People*. Syngress. ISBN: 978-0128142847

Open Elective-I

Course 3: Cybersecurity Essentials & Digital Privacy

Program Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Open Elective-I(Cybersecurity Essentials & Digital Privacy)**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		3-0-0	3	45
Type of Course:	OEC			

Course Perspective. This course provides an essential foundation in cybersecurity and personal data privacy for everyone. It explores common threats in the digital world, introduces best practices for protecting information, and examines ethical and legal issues related to digital security. The course empowers students to become vigilant digital citizens and informed decision-makers in a tech-driven society.

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

COs	**Statements**
CO1	Understanding the basics of encryption, authentication, and secure communications.
CO2	Identifying common cyber threats and vulnerabilities in everyday technologies.
CO3	Applying best practices for securing personal devices, accounts, and online identities.
CO4	Evaluating legal frameworks and ethical concerns related to data privacy.
CO5	Critically evaluate and make informed decisions regarding digital safety in both personal and professional settings.

Course Outline:

Unit Number: 1	Title: Cybersecurity Fundamentals	No. of hours: 12
Content Summary: • What is cybersecurity? Scope and importance • Key concepts: confidentiality, integrity, availability (CIA Triad) • Overview of threats: malware, phishing, ransomware, social engineering • Anatomy of a cyber-attack: how breaches happen		
Unit Number: 2	**Title: Securing Devices and Networks**	**No. of hours: 13**
Content Summary: • Secure passwords and multi-factor authentication (MFA) • Network security basics: firewalls, VPNs, antivirus software • Device hardening: OS updates, secure settings, mobile security • Safe browsing and email hygiene		
Unit Number: 3	**Title: Digital Privacy and Data Protection**	**No. of hours: 10**
Content Summary: • Understanding personal data: what's collected and why • Cookies, trackers, and surveillance capitalism • Data protection laws: GDPR, IT Act (India), and global perspectives • Managing online footprints and digital identity		
Unit Number: 4	**Title: Ethics, Policies, and Future Risks**	**No. of hours: 10**
Content Summary: • Ethical hacking and cybersecurity careers • Responsible disclosure and bug bounty programs • Emerging risks: deepfakes, IoT, AI-powered attacks • Case studies: data breaches and lessons learned		

Learning Experience

Inside Classroom Learning

- **Interactive Lectures:**
  Explain core cybersecurity concepts (CIA triad, threats, malware) and digital privacy principles using real-world examples.
- **Live Demos & Simulations:**
  Demonstrate secure password creation, phishing detection, VPN usage, and browser privacy settings.
- **Case-Based Learning:**
  Analyze famous cyberattacks and data breaches to discuss risks and legal/ethical issues.
- **Discussion & Q&A Sessions:**
  Debates on surveillance capitalism, data protection laws (like GDPR), and ethical hacking practices.
- **Mini Activities:**
  Students conduct a security check-up on their devices and evaluate their own digital footprints.

Outside Classroom Learning

- **Hands-On Practice:**
  Practice setting up MFA, using password managers, managing cookies, and simulating email security filters.
- **Collaborative Assignments:**
  Group presentations on recent cybersecurity incidents or privacy policy audits of popular platforms.
- **Self-Paced Learning:**
  Resources: CyberAware campaigns, FreeCodeCamp security tutorials, Mozilla's Privacy Tools.

- **Capstone Task:**

  Audit a personal or simulated digital presence and propose an action plan for improving digital safety

Textbook: Grimes, R. A. (2019). *Cybersecurity Basics: A Hands-On Approach*. Packt Publishing. ISBN: 978-1838827846

Additional References:

- Schneier, B. (2015). *Data and Goliath: The Hidden Battles to Collect Your Data and Control Your World*. W.W. Norton & Company. ISBN: 978-0393352177
- Andress, J. (2019). *Cybersecurity and Privacy: An Introduction for Non-Technical People*. Syngress. ISBN: 978-0128142847

Open Elective-I

Course 4: IoT and Smart Devices in Everyday Life

Program Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Open Elective-I(IoT and Smart Devices in Everyday Life)**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		3-0-0	3	45
Type of Course:	OEC			

Course Perspective. This course introduces the Internet of Things (IoT) with a focus on the everyday technologies that surround us — from smart homes and wearables to connected vehicles and city infrastructure. Students will explore how devices communicate, the role of sensors, and the ecosystem of apps and cloud platforms that power smart environments. No technical background is assumed, making it ideal for anyone curious about the impact of connected devices on modern life.

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

COs	**Statements**
CO1	Explaining the basic concepts of IoT, including sensors, networks, and data flow.
CO2	Identifying common smart devices and understand how they function.
CO3	Understanding communication protocols used in IoT (conceptually).
CO4	Discussing privacy, security, and ethical issues with smart technologies.
CO5	Evaluating the potential and risks of IoT in various sectors like healthcare, homes, and transportation.

Course Outline:

Unit Number: 1	Title: Introduction to IoT and Smart Devices	No. of hours: 12
Content Summary: • What is the Internet of Things? Evolution and use cases • Types of smart devices: wearables, appliances, vehicles, city infrastructure • Components of IoT: sensors, actuators, microcontrollers • How smart devices collect and transmit data		
Unit Number: 2	**Title: IoT Connectivity and Platforms**	**No. of hours: 13**
Content Summary: • Communication technologies: Wi-Fi, Bluetooth, ZigBee, LoRa (conceptual) • Cloud platforms and IoT apps: how data is stored and accessed • Mobile apps, voice assistants, and smart hubs • Examples of IoT ecosystems (e.g. Google Home, Alexa, Apple HomeKit)		
Unit Number: 3	**Title: Smart Applications and Industry Use Cases**	**No. of hours: 10**
Content Summary: • Smart homes: thermostats, lighting, home security • Healthcare: fitness trackers, telemedicine, remote monitoring • Smart cities: traffic systems, waste management, pollution sensors • Transportation: connected cars, autonomous navigation		
Unit Number: 4	**Title: Ethics, Security, and Future Trends**	**No. of hours: 10**
Content Summary: • Privacy and surveillance in IoT • Data ownership and transparency • Security risks: hacking smart devices, botnets (conceptually) • The future of ubiquitous computing and ambient intelligence		

Learning Experience

Inside Classroom Learning

Conceptual Lectures:
Explain IoT components (sensors, data flow, smart hubs) and smart device functions using real-life examples.

Interactive Discussions:
Discuss applications in smart homes, healthcare, transportation, and cities.

Visual Demonstrations:
Use videos or device mockups to show IoT ecosystems (e.g., Alexa, Google Home).

Ethics & Security Dialogues:
Analyze privacy concerns and conceptual security threats (e.g., data leaks, surveillance).

Case Study Analysis:
Evaluate successful IoT applications like smart traffic systems or fitness trackers.

Outside Classroom Learning

- **Hands-On Exploration**:

  Students explore IoT devices via mobile apps, virtual labs, or device simulators.
- **Mini Projects**:

  Design simple use-case diagrams for smart environments (home/healthcare).
- **Research Assignments**:

  Investigate emerging IoT trends or ethical dilemmas in connected technology.
- **Self-Learning Resources**:

  Use YouTube explainers, Coursera/EdX IoT intros, and consumer IoT product reviews.

Texbook: Vermesan, O., & Friess, P. (Eds.). (2022). Internet of Things – From Research and Innovation to Market Deployment. River Publishers. ISBN: 978-8793102946

Open Elective-I

Course 5: Coding Basics with Python for Beginners

Program Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Open Elective-I(Coding Basics with Python for Beginners)**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		3-0-0	3	45
Type of Course:	OEC			

Course Perspective. This beginner-friendly course introduces students to the fundamentals of computer programming using Python, one of the most accessible and widely-used programming languages today. Emphasis is placed on developing problem-solving skills, writing readable code, and applying logic to automate tasks. The course is designed for students with no prior coding experience and uses hands-on examples to make learning engaging and intuitive.

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

COs	**Statements**
CO1	Understanding the basic syntax and structure of Python programs.
CO2	Using variables, data types, and operators to write simple programs.
CO3	Implementing control structures like conditionals and loops.
CO4	Writing and use functions to modularize code.
CO5	Interpreting data by Read, process, and write data using files and lists.

Course Outline:

Unit Number: 1	Python Fundamentals	No. of hours: 12

Content Summary:

- Introduction to programming and Python
- Installing Python and writing your first program
- Basic data types: numbers, strings, booleans
- Variables and type conversion
- Using print(), input(), and simple expressions

Unit Number: 2	Title: Control Structures and Logic	No. of hours: 13

Content Summary:

- Boolean logic and comparison operators
- Conditional statements: if, elif, else
- Loops: for and while
- Loop control: break, continue, pass
- Practical examples: menu systems, calculators

Unit Number: 3	Title: **Functions and Code Organization**	No. of hours: 10

Content Summary:

- Defining and calling functions
- Parameters, return values, and scope
- Default and keyword arguments
- Organizing code with modules and comments
- Debugging basics and error handling

Unit Number: 4	Title: **Working with Data**	No. of hours: 10

Content Summary:

- Lists, tuples, and dictionaries: storing collections
- List comprehensions and simple algorithms
- Reading from and writing to files
- Basic string manipulation and formatting
- Mini project: command-line to-do list or contact manager

Learning Experience

Inside Classroom Learning

- **Hands-on Coding Sessions:**
  Practice writing basic Python programs using variables, loops, and functions.
- **Interactive Problem Solving:**
  Solve real-life logic problems like calculators, menus, and file handling.
- **Live Debugging & Demos:**
  Demonstrate error handling and function behavior with live examples.
- **Peer Code Reviews:**
  Improve readability and structure through code walkthroughs.

.

Outside Classroom Learning

- **Mini Coding Tasks:**
  Practice small challenges using platforms like Replit or Google Colab.
- **Self-paced Assignments:**
  Write scripts to automate simple tasks or process user input/output.
- **Project-based Learning:**
  Build a basic utility (like a to-do list app or contact book) as a capstone.
- **Video Tutorials & Practice Platforms:**
  Use resources like W3Schools, Codecademy, or Python.org to reinforce learning.

Textbook: Downey, A. (2015). *Think Python: How to Think Like a Computer Scientist* (2nd ed.). O'Reilly Media. ISBN: 978-1491939369

Additional References:

- Sweigart, A. (2020). *Automate the Boring Stuff with Python* (2nd ed.). No Starch Press. ISBN: 978-1593279929

  Zelle, J. (2017). *Python Programming: An Introduction to Computer Science* (3rd ed.). Franklin, Beedle & Associates. ISBN: 978-1590282755

SEMESTER: III

Nand to Tetris-I

Program Name	B.Tech (CSE) with specialization in UI/UX			
Course Name: Nand to Tetris-I	Course Code	L-T-P	Credits	Contact Hours
	ETCCNT301	3-0-2	4	45
Type of Course:	Major			

Course Perspective. This course Students build a complete 16-bit computer—starting with a single NAND gate—by successively engineering logic circuits, memory modules, a CPU, and a full assembler. Emphasis is on hands-on simulation, rigorous unit testing, version control, and reflective design journaling. This course uniquely grounds software engineers in the fundamental hardware abstractions, exposing the deep connection between code and physical computation, vital for optimizing performance, understanding operating systems, and developing secure, efficient systems.

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

COs	Statements
CO 1	**Constructing** foundational digital circuits and memory units from basic logic gates.
CO 2	**Integrating** components to build a functional CPU and understanding its architecture.

CO 3	**Developing** an assembler to translate symbolic instructions into machine code.
CO 4	**Applying** low-level debugging techniques and connecting hardware concepts to software execution.

Course Outline:

Unit Number: 1	**Boolean & Arithmetic Foundations**	**No. of hours: 12**

Content:

- Functional completeness of NAND; truth tables & boolean algebra refresh.
- Boolean Logic in Programming: Brief discussion of bitwise operations, logical operators (AND, OR, NOT, XOR) in high-level languages and their hardware mapping.
- HDL syntax (Nand2Tetris style) & automated test scripts.
- Devices: NOT, AND, OR, XOR, Mux, DMux, Half/Full Adder, 16-bit Ripple-Carry Adder.
- Introduction to Logic Families (Conceptual): Briefly touch upon physical implementations like TTL/CMOS gates, gate delays, and power consumption for real-world context.

Unit Number: 2	**Sequential Logic & Memory Hierarchy**	**No. of hours: 12**

Content:

- D-flip-flop from gates; binary clocking model.
- Registers, Register files, 16-bit RAM chips (8K & 16K variants).
- Program Counter (PC) design, load-increment logic.
- Memory-mapped-I/O concept preview.
- Memory Hierarchy (Conceptual Link): Briefly discuss the relationship between the RAM built and higher levels of memory hierarchy (cache, virtual memory) from a structural perspective, acknowledging these are detailed in advanced courses.
- Sequential Logic Applications: How flip-flops form the basis of counters, shift registers, and state machines.

Unit Number: 3	Title: CPU Architecture	No. of hours: 11

Content:

- Control word design for the Hack CPU (ALU flags, jump bits).
- Von Neumann Architecture: Explicit discussion of its principles (stored program concept, single address space for instructions and data) as realized in the Hack CPU.
- Micro-programmed vs hard-wired control (comparative discussion).
- Instruction Cycle: Detailed explanation of the Fetch-Decode-Execute cycle as implemented in the Hack CPU.
- Integration workflow: ALU ↔ Control ↔ Memory ↔ PC.
- Cycle-accurate simulation, waveform inspection, regression test harness.
- Introduction to RISC vs. CISC (Conceptual): Briefly compare the Hack ISA's simplicity (RISC-like) to more complex ISAs (CISC) and the trade-offs.
- Pipelining (Conceptual Preview): Briefly introduce the idea of pipelining instructions for performance, as a logical next step in CPU design.

Unit Number: 4	Machine Language & Assembler Implementation	No. of hours: 10

Content:

- Hack ISA: C-instruction, A-instruction, pseudo-commands.
- Two-pass assembler algorithm (symbol resolution → code emission).
- Error handling, source-map file generation, automated test pack.
- Deploying assembled binaries on the CPU emulator.
- Software Toolchain Context: Discuss the broader roles of compilers, linkers, and loaders in transforming high-level code to executable binaries, and where the assembler fits in this chain.
- System Calls (Conceptual Bridge): How machine instructions interact with the underlying hardware/operating system for I/O (e.g., keyboard input, screen output) and other services.
- Low-level Security Implications (Brief): How understanding machine code can provide insights into vulnerabilities like buffer overflows or reverse engineering.

List of Experiments

Ex. No	Experiment Title	Mapped COs
1	**Digital Logic Prototyping Toolkit** **Objective:** mastery of combinational design & test. • Implement AND/OR/NOT/XOR (1-bit & 16-bit). • Build 4-way & 8-way multiplexers/demultiplexers. • Develop a Python-based truth-table generator to auto-validate HDL chips. Sub-task Extension: Analyze the propagation delay of your combinational circuits (conceptually or using simulation tools if provided) and discuss its impact on clock speed. Deliverable: Git repo + CI badge + brief design journal.	CO 1
2	**Arithmetic-Logic Unit (ALU) Construction** **Objective:** perform arithmetic/logic with control bits & status flags. • Ripple-carry adder → 16-bit ALU including zero/neg flags. • Extend ALU with bitwise AND/OR and unary negation options. • Create an interactive CLI tester that lets classmates feed control bits and observe outputs. • Sub-task Extension: Design a basic 2-bit barrel shifter or logical shifter as an optional ALU extension and discuss its utility in programming. • Performance extension (optional): design a carry-look-ahead variant & compare delay.	CO 2
3	**Memory Subsystem Engineering** **Objective:** build and benchmark memory. • 16-bit register → Register file (8 registers). • RAM8→RAM64→RAM512→RAM8K→RAM16K (hierarchical build script). • Instrument memory with read/write latency counters; graph latency vs size.	CO 3

	• Sub-task Extension: Briefly compare the access patterns and ideal use cases of the RAM built here with conceptual L1/L2 caches (no implementation, just discussion). • Design a parametric memory generator HDL macro.	
4	**CPU Integration & Diagnostic Harness** **Objective:** realize the Hack CPU and validate instruction cycle. • Design finite-state control unit per Hack spec. • Integrate ALU + PC + Memory; run supplied test ROMs. • Build a waveform-viewer workflow (GTKWave/ModelSim) showing ALU/control signals. • Create automated regression tests for 10 canonical programs (Max, Add, Pong ASM stubs). • Sub-task Extension: Using the CPU emulator, identify and demonstrate a simple case where instruction timing or ALU flags are critical for program execution (e.g., a tight loop, conditional jump).	CO 3
5	**Capstone: Fully Bootable Hack Computer** **Objective:** deliver the complete hardware + assembler tool-chain. • Write the two-pass assembler in Python/Go/Rust (student's choice). • Assemble & execute a small OS-style monitor that echoes keyboard input to screen. • Sub-task Extension: Implement a very basic interrupt or exception handling mechanism (e.g., a fixed memory location for a "system call" or error routine), demonstrating how the CPU can respond to external events or program errors. • Produce a screencast demo + README detailing architecture, challenges, lessons. • Assessment: functionality (40 %), code quality (20 %), documentation (20 %), demo (20 %).	CO 4

Classroom Learning Experience

Inside Classroom Learning:

1. **Conceptual Chalk-Talks & Live Demos:** Use board work and simulations to explain key topics like NAND logic, HDL syntax, memory circuits, and CPU architecture.
2. **Hands-on HDL Lab Sessions:** Students build logic gates, adders, and memory components using HDL with real-time simulator feedback.
3. **Interactive Proof-of-Concepts:** Implement and test D flip-flops, Program Counters, and instruction cycles in the classroom with guided templates.
4. **Logic Games & Group Work:** Group-based challenges to design optimized circuits (like a 4-bit ALU or full adder) using only NAND gates.
5. **Concept Bridge Discussions:** Regular conceptual links to real-world systems (TTL/CMOS, memory hierarchy, bitwise operations, RISC/CISC) with visuals and comparisons.

Outside Classroom Learning Experience

1. **Take-Home HDL Assignments:** Design and simulate logic components like Mux, RAM, and CPU parts using HDL and test scripts.
2. **Tool-Based Practice:** Use Nand2Tetris, Logisim, or Python-based emulators to visualize circuits and instruction execution.
3. **Mini Projects:** Tasks like building a mini assembler, simple FSM, or logic-based calculator using HDL and testing tools.
4. **Online Explorations:** Learn via YouTube, OCW videos, or open simulators on CPU design, instruction cycles, and logic families.
5. **Peer Learning Forums:** Collaborate with classmates through LMS discussion boards to debug HDL code, share insights, and post queries.

Textbooks:

- Nisan, Noam, and Schocken, Shimon. *The Elements of Computing Systems: Building a Modern Computer from First Principles*. MIT Press, 2005.
- Patterson, David A., and Hennessy, John L. *Computer Organization and Design RISC-V Edition: The Hardware/Software Interface*. Morgan Kaufmann

Specialization Course-I

Introduction to UI/UX & Design Thinking

Program Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Introduction to UI/UX & Design Thinking**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETUXDT305	3-0-2	4	45
Type of Course:	DSE			
Pre-requisite(s), if any: Basic of programming				

Course Perspective. This course equips students with practical skills and foundational knowledge in UI/UX design and design thinking. It provides a comprehensive introduction to human-centered design, UX research, wireframing, prototyping, and ethical digital product development using industry-standard tools like Figma and Miro. Emphasizing collaboration, iterative development, and real-world case studies, the course prepares students to design intuitive and engaging digital experiences. By the end of the course, students will have a beginner-level portfolio and be job-ready for roles like UI/UX Designer, Product Designer, or UX Research Assistant.

The Course Outcomes (COs)

COs	Statements
CO 1	**Understanding** fundamental differences between User Interface (UI) and User Experience (UX), and their interdependent roles in the product development.
CO 2	Applying design thinking process and UCD methodologies to real-world design problems.

CO 3	Using Figma, Miro, and Trello to ideate, wireframe, prototype, and collaborate on design projects.
CO 4	Analyzing existing digital products using heuristic evaluation and usability principles.
CO 5	Demonstrating awareness of ethical, inclusive, and accessible design practices

Course Outline:

Unit 1:	**Title: Understanding UI, UX, and Design Thinking Foundations**	**No. of hours: 1 0**

Objective: Understand the foundational concepts of UI/UX and why design matters in product success.

Content:

- Introduction to UI and UX: Definitions, key differences, real-world impact
- How UI and UX work together: Journey from interaction to experience
- Business value of UX: Brand loyalty, user engagement, retention, ROI
- Introduction to Design Thinking: Empathize, Define, Ideate, Prototype, Test
- Real-world product examples: Case study analysis (e.g., IRCTC vs Zomato, Paytm vs Google Pay)
- Critiquing Designs: Heuristic evaluation (Nielsen's principles), Usability goals

Hands-On:

- Team-based product critique and UI/UX SWOT analysis
- Design Thinking Icebreaker: Reimagine a student canteen app in teams

Unit 2:	**Title: User-Centered Design, Personas & User Research**	**No. of hours: 1 0**

Objective: Apply empathy-based design techniques and understand the user before designing.

Contents:

- What is UCD? Principles and benefits of user-first design
- Empathy building: User interviews, surveys, contextual inquiry
- Creating personas: Demographics, goals, frustrations, needs
- Mapping journeys and touchpoints: Empathy maps, user journey maps
- Ideation strategies: HMW questions, SCAMPER, Crazy 8s
- Basics of Lean UX and agile integration in UI/UX workflows

Hands-On:

- Field assignment: Conduct 3 peer interviews about a common app
- Create user personas, empathy maps, and journey maps in Miro or FigJam

Unit 3	**Title: Wireframing, Prototyping & Tooling with Figma, Miro, Trello**	**No. of hours: 1 3**

Objective: Use modern tools to create interactive and accessible interfaces.

Contents:

- Figma Fundamentals: Frames, layout grids, text, components, styles
- Wireframing: Low-fidelity and high-fidelity designs
- Prototyping basics: Links, transitions, hover states
- Creating reusable UI components and design systems
- Collaborative Design: Miro and FigJam for team ideation
- Project Tracking: Trello/Asana basics (Boards, lists, Kanban views)
- Conducting basic usability testing on prototypes

Hands-On:

- Lab: Design a 3-screen mobile app (Login, Dashboard, Profile) in Figma
- Team sprint: Manage a design sprint using Trello with tasks and deadlines

Unit 4	**Industry Roles, Ethics, Accessibility & UI/UX Critique**	**No. of hours: 1 2**

Objective: Understand team roles, industry practices, and responsible, inclusive design.

Contents:

- Industry roles: UI Designer, UX Researcher, Product Designer, UX Writer, Interaction Designer
- Team collaboration models in startups vs large firms
- Design ethics: Dark patterns, cognitive overload, persuasive tech
- Inclusive design & accessibility (WCAG principles, alt text, color contrast)
- Peer critiques: Giving/receiving feedback constructively
- Portfolio preparation basics

Hands-On:

- Peer review activity: Heuristic critique of a team's Figma prototype
- Create a checklist-based accessibility audit of a live website (e.g., Flipkart)

LAB EXPERIMENTS

Ex. No	Experiment Title	Mapped COs

1	**Lab Assignment 1 (Unit 1):** **Title:** "Redesigning the Experience of a University Attendance App" **Sub-Objectives:** 1. Identify current UI/UX issues in a real or mock attendance app. 2. Map out user flows for students, faculty, and admin roles. 3. Conduct a heuristic evaluation using Nielsen's principles. 4. Apply design thinking to reimagine user flows. 5. Present design improvement recommendations with visuals.	CO 1
2	**Lab Assignment 2 (Unit 2):** **Title:** "Designing a Mental Health Tracker for College Students" **Sub-Objectives:** 1. Conduct 3 short interviews to understand emotional pain points. 2. Build 2 user personas based on collected data. 3. Draft empathy maps and journey maps of students seeking help. 4. Identify touchpoints and opportunity areas for digital solutions. 5. Ideate app features collaboratively using FigJam.	CO 2
3	**Lab Assignment 3 (Unit 3):** **Title:** "Prototyping a Food Delivery App for Rural India" **Sub-Objectives:** 1. Sketch a 5-screen user flow (Home, Cart, Orders, Payment, Support).	CO 3

	2. Use Figma to build high-contrast, simple UI for low-end phones. 3. Create reusable UI components for forms and cards. 4. Add interactivity with Figma prototyping features. 5. Present your prototype and gather usability feedback from peers.	
4	**Lab Assignment 4 (Unit 4):** **Title:** "Inclusive Redesign of a Movie Ticket Booking App" **Sub-Objectives:** 1. Perform a WCAG audit on an existing app (e.g., BookMyShow). 2. Identify barriers for elderly and visually impaired users. 3. Redesign 3 key screens in Figma using inclusive practices. 4. Document the ethical choices and justifications in the design. 5. Conduct a peer-based design critique and implement feedback.	CO4
5	**Capstone Assignment** **Title:** Designing an Inclusive Campus Life SuperApp for Students Sub-Objectives: 1. Apply the design thinking process (Empathize, Define, Ideate, Prototype, Test) to identify and solve major student life pain points (e.g., cafeteria access, campus navigation, event booking, complaint redressal). 2. Conduct user research using interviews or surveys with students from different years and backgrounds to inform design decisions. 3. Create user personas, empathy maps, and journey maps to synthesize insights and map out user needs.	CO5

	4. Design low- to mid-fidelity wireframes in Figma covering at least 5 app screens with proper layout, navigation, and interaction. 5. Conduct a heuristic evaluation and usability testing session, document feedback, and iteratively refine the design for inclusivity, accessibility, and usability.	

Text Books

- Krug, S. (2014). Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability (3rd ed.). New Riders.

Reference Books

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

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.

Web Development III (Node.js & Express Backend)

Programme Name	B.Tech (CSE) with specialization in UI/UX			
Course Name: Web Development III (Node.js & Express Backend)	Course Code	L-T-P	Credits	Contact Hours
	ETCCWD303	3-0-2	4	45
Type of Course:	Major			

Course Perspective: This course takes you from "I can build a React front-end" to "I own the whole stack." You'll design and implement RESTful & real-time backends with Node.js and Express, wire them to databases, secure them with auth & validation, write automated tests, and ship via CI/CD. By the end, you'll have a production-grade API powering a deployed full-stack app, plus the mindset of an entry-level Backend/Full-Stack Engineer.

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

COs	Statements
CO 1	**Architect and implement RESTful/GraphQL APIs** using Node.js, Express, and modular code organization.
CO 2	**Persist and query data** with both relational and NoSQL databases, applying schema design, indexing, and ORM/ODM best practices.
CO 3	**Secure backends** through input validation, authentication (JWT/OAuth2), authorization (RBAC/ABAC), and OWASP-recommended practices.
CO 4	**Automate quality and delivery** with unit/integration tests (Jest/Supertest), linting, Dockerization, and CI/CD pipelines.

CO 5	**Deploy and monitor** Node services on cloud platforms, implement logging/metrics, and optimize performance & scalability.

Course Outline:

Unit Number: 1	**Title: Node.js Internals & Express Basics**	**No. of hours: 12**

Content:

- **Node runtime model:** Event loop, libuv, threads vs async I/O, streams & buffers.
- **Module systems & project structure:** ES Modules vs CommonJS, dotenv/config patterns, scripts.
- **Express fundamentals:** Middleware pipeline, routing, request/response lifecycle, error handlers.
- **Async patterns in backend:** Promises/async-await pitfalls, centralized error handling.
- **Tooling:** Nodemon, ts-node/TypeScript intro (optional), ESLint/Prettier for server code.
- **Hands-on focus:** Build a tiny REST API (users/todos), custom middleware, centralized logger & error handler.

Unit Number: 2	**Title: Data Layer & Business Logic**	**No. of hours: 12**

- **Database choices:** SQL vs NoSQL; when to pick Postgres/MySQL vs MongoDB.
- **ODM/ORM basics:** Mongoose/Prisma/Sequelize – schema/model definition, relations, migrations.
- **Data validation:** Joi/Zod/Yup schemas on request body & DB layer; sanitization against NoSQL injection.
- **Service & repository pattern:** Separating controllers, services, and data access for clean architecture.
- **Caching layer:** Redis basics, caching strategies (per-request, query-level), cache invalidation workflows.

- **Hands-on focus:** Implement CRUD with validation, indexes, pagination & filtering; add Redis caching for hot endpoints.

Unit Number: 3	Title: Auth, Security & Real-Time Communication	No. of hours: 11

Content

- **Auth flows:** Session vs JWT, refresh tokens, OAuth2 basics (Google/GitHub login), password hashing (bcrypt/argon2).
- **Authorization:** Role-Based (RBAC) & Attribute-Based (ABAC) guards, route-level & resource-level checks.
- **Security best practices:** OWASP Top 10 for APIs, rate limiting, helmet/cors, CSRF for SSR, secrets management.
- **Real-time backends:** WebSockets/Socket.IO basics, event architecture, presence & typing indicators.
- **File uploads & media handling:** Multer/Busboy, S3/object storage integration.
- **Hands-on focus:** Add full auth stack (login/refresh/logout), protected routes, role checks; live notifications via Socket.IO.

Unit Number: 4	Title: Testing, Performance & Deployment	No. of hours: 10

Content:

- **Testing pyramid:** Unit vs integration vs E2E; Jest + Supertest for HTTP, Testcontainers/Docker for DB tests.
- **Performance & scalability:** Clustering, PM2, load testing (k6/Artillery), profiling, async bottleneck fixes.
- **Logging & observability:** Winston/Pino loggers, morgan, structured logs, basic metrics (Prometheus/Grafana intro).
- **DevOps & CI/CD:** Dockerfile & docker-compose, GitHub Actions for test/build/deploy, env separation.

- **Deploy targets:** Railway/Render/Heroku, AWS EC2/Lambda basics, reverse proxies (NGINX), HTTPS/Let's Encrypt.

Hands-on focus: Containerize app, set up CI to run tests & push image, deploy staging build; run load tests & document results

List of Experiments

Each lab ties to a unit and builds toward a capstone. Submit code, README with setup steps, and short demo video/gif.

Ex. No	Experiment Title	Mapped COs
1	**Lab 1 (Unit 1) – "MicroBlog API Starter"** • Initialize Node + Express project with ES Modules and dotenv. • Create CRUD endpoints for posts and users with in-memory store. • Write a custom logger middleware and global error handler. • Add ESLint/Prettier, run unit tests for utility functions (slugify, sanitizer).	CO 1
2	**Lab 2 (Unit 2) – "Inventory Manager with DB & Cache"** • Connect to MongoDB or Postgres (student's choice) using Mongoose/Prisma. • Define schemas/tables with validations, relations (products ↔ categories).	CO 2

	• Implement pagination, filtering, and sorting; cache GET /products in Redis. • Add Postman/Insomnia collection; write integration tests with Supertest	
3	**Lab 3 (Unit 3) – "Auth'd Chat API + WebSocket Events"** • Implement user signup/login with JWT access + refresh tokens. • Protect chat endpoints with middleware (role: user/admin). • Use Socket.IO to broadcast messages and typing indicators. • Add rate limiting & helmet; test protected routes and WebSocket events.	CO 3
4	**Lab 4 (Unit 4) – "CI/CD & Observability Pipeline"** • Dockerize the API; write docker-compose for app + DB + Redis. • Configure GitHub Actions: lint, test, build Docker image, deploy to Render/Netlify Functions/AWS. • Add Pino/Winston logging with request IDs; expose a /health and /metrics endpoint. • Run a basic load test (Artillery/k6); document RPS, latency, CPU/mem graphs.	CO4
5	**Lab 5 – Capstone (All Units) – "CampusConnect API"** • Feature set: announcements, events, clubs, chat, user roles (student/faculty/admin). • Tech stack: Express API + Mongo/Postgres + Redis cache + Socket.IO notifications. • Quality gates: 80%+ test coverage, Dockerized, CI pipeline, monitored endpoints.	CO5

	• Deploy to cloud; provide OpenAPI/Swagger docs and a Postman collection. • Final deliverable: live URL, repo, docs with architecture diagram & performance report.	

Learning Experiences

Inside Classroom Learning:

- **Code-Along Sessions**
  Guided coding of RESTful APIs, middleware, and real-time features using Node.js and Express.
- **Architecture Walkthroughs**
  Analyze real backend architectures, applying clean code and service-layer patterns.
- **Live Debugging & Security Drills**
  Practice tracing bugs, fixing async issues, and applying security patches (JWT, RBAC, OWASP fixes).
- **Project Discussions & Peer Reviews**
  Present API design choices, receive feedback, and improve modularity and test coverage.

Outside Classroom Learning Experience

- **Hands-on Labs & Mini Projects**
  Build modular backends, integrate Mongo/Postgres, add caching/auth, and deploy using Docker and CI/CD.
- **Tool Familiarization**
  Practice with tools like Postman, Docker, GitHub Actions, Redis, Socket.IO, and Prometheus.
- **Capstone Development**
  Collaboratively develop a full-stack API (CampusConnect) with test coverage, CI/CD, and deployment.

- **Testing & Monitoring Exercises**
  Run load tests, analyze logs/metrics, and document performance optimizations and bottlenecks

Practical Textbook (Primary)

Node.js Design Patterns (3rd Edition, 2024) – Mario Casciaro & Luciano Mammino, Packt Publishing, ISBN: 978-1805817393

Additional References (Optional but Recommended)

- **API Design Patterns (2022)** – JJ Geewax, O'Reilly.
- **Express in Action (2nd Ed., 2022)** – Evan Hahn, Manning.
- **Testing JavaScript Applications (2021)** – Lucas da Costa, Manning.

Assessment Strategy (Suggested)

- **Midterm (30%)** – Theory + short coding task on paper/online.
- **Labs & Assignments (30%)** – Code quality, feature completion, documentation.
- **Capstone Project (25%)** – Functionality, testing, deployment, presentation.
- **Viva/Quizzes (15%)** – Concept checks on Node internals, security, DB design.

Tools & Platforms

- **Core:** Node.js LTS, Express, MongoDB/Postgres, Redis, Socket.IO
- **Testing:** Jest, Supertest, Testcontainers (optional)
- **DevOps:** Docker, GitHub Actions, Railway/Render/Heroku/AWS
- **Monitoring:** Pino/Winston, Postman, k6/Artillery for load testing

Database Management Systems

Programme Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Database Management System**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCMS304	**3-0-2**	**4**	**45**
Type of Course:	**Major**			

Course Perspective: This course provides a comprehensive, hands-on introduction to database concepts, relational database design, SQL programming, query optimization, transactions, database security, and modern industry tools. It also covers NoSQL databases, cloud databases, and real-world applications to equip students with industry-relevant skills.

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

COs	Statements
CO 1	**Designing** and normalizing relational databases using ER modeling and functional dependencies.
CO 2	**Implementing** complex SQL queries, indexing, and views for data retrieval and optimization.
CO 3	**Developing** database transactions, stored procedures, and triggers while ensuring data integrity and security.
CO 4	**Working** with NoSQL databases, cloud databases, and integrate databases into real-world applications.

Course Outline:

Unit Number: 1	**Title: Database Design & Relational Model**	**No. of hours: 12**

Content:

- Introduction to DBMS: Database vs. File Systems, Characteristics, Applications.
- DBMS vs RDBMS with E. F. Codd's Rules & Their Industry Significance
- ER Model: Entities, Relationships, Generalization, Specialization, Aggregation.
- Relational Model: Schema, Keys (Primary, Foreign, Candidate), Integrity Constraints.
- Normalization Techniques: Functional Dependencies, Normal Forms (1NF to BCNF).
- Star & snowflake schemas for analytics

Tool demo: Lucidchart/dbdiagram → MySQL

Industry Use Cases & Practical Focus:

✓ Industry Tool: Use Lucidchart or dbdiagram.io to create ER models.

✓ Practical Exercise: Design a Library Management System using ER diagrams and convert them into relational schemas using MySQL/PostgreSQL.

✓ Real-World Application: How relational databases are structured for banking and healthcare applications.

Unit Number: 2	**Title: SQL Queries & Advanced Operations**	**No. of hours: 12**

Content:

- SQL Fundamentals: DDL, DML, DCL, TCL Commands.
- Complex Queries: Joins (Inner, Outer, Self), Subqueries, Set Operations.
- Views and Indexing: Creating, Modifying, and Optimizing Queries with Indexes.
- Advanced SQL: Window Functions, Common Table Expressions (CTEs).

Industry Use Cases & Practical Focus:

✓ Industry Tool: Use MySQL Workbench or pgAdmin for query execution and indexing.

✓ Practical Exercise: Write advanced SQL queries to analyze customer behavior trends from an e-commerce database.

✓ Real-World Application: How SQL is used in business intelligence (BI) for data analytics and reporting in retail and finance.

Unit Number: 3	**Title: Transactions, PL/SQL & Database Security**	**No. of hours: 11**

Content:

- Transactions & Concurrency Control: ACID Properties, Commit, Rollback, Savepoints.
- Locking & Deadlocks: Isolation Levels, Optimistic vs. Pessimistic Locking.
- PL/SQL Programming: Stored Procedures, Functions, Cursors, Triggers.
- Database Security: User Roles, Privileges, Role-Based Access Control (RBAC), SQL Injection Prevention.

Industry Use Cases & Practical Focus:

✓ Industry Tool: Implement stored procedures & triggers using MySQL or PostgreSQL.
✓ Practical Exercise: Develop a Banking System to handle secure transactions, fund transfers, and balance updates with triggers.
✓ Real-World Application: How transaction management is used in stock trading platforms and e-commerce payment gateways.

Unit Number: 4	**Title: Database Performance, NoSQL & Real-World Applications**	**No. of hours: 10**

Content:

- Query Optimization Techniques: Execution Plans, Indexing (B-Trees, Hash Indexing).
- NoSQL Database Concepts: Key-Value Stores, Document Stores (MongoDB, Firebase).
- Cloud Databases & Distributed Storage: AWS RDS, Google BigQuery, Azure SQL.
- Big Data & Data Warehousing: ETL Processes, Data Lake vs. Data Warehouse.

Industry Use Cases & Practical Focus:

✓ Industry Tool: Use MongoDB Atlas to create NoSQL databases.
✓ Practical Exercise: Design a Social Media Database using MongoDB for storing user posts, comments, and interactions.

✓ Real-World Application: How NoSQL databases are used in real-time recommendation systems (Netflix, YouTube).

List of Experiments

Ex. No	Experiment Title	Mapped COs
1	**Database Design & Basic SQL – College Record Management** **Objective:** To design a relational database from an ER model, normalize it, and perform basic CRUD operations using SQL. Real-World Scenario: University admission and internal marks management system. **Sub Objectives** • Draw ER diagram and normalize data up to 3NF for entities: Student, Faculty, Course, Exam. • Implement schema using DDL commands in MySQL/PostgreSQL. • Perform DML operations: INSERT, UPDATE, DELETE, and SELECT with filters and joins. • Apply integrity constraints (Primary Key, Foreign Key, Unique, Not Null). Learning Focus: ER modeling, normalization, schema creation, basic queries.	CO 1
2	**Transactions, Triggers, and Procedural SQL – Hospital Appointment System** **Objective:** To implement real-world logic using triggers and procedures and handle time-sensitive operations securely. Real-World Scenario: Hospital patient records, appointments, and medical logs. **Sub Objectives** • Design tables for Doctor, Patient, Appointment, Treatment, and Prescription. • Write SQL triggers to avoid overlapping appointments and log changes. • Use transactions to ensure atomicity while booking and modifying appointments. • Create stored procedures for patient registration and treatment entry.	CO 2

	Learning Focus: Triggers, atomic transactions, stored procedures, real-time validations.	
3	**Data Integration and Web Connectivity – E-Commerce Backend** **Objective:** To connect the database with a simple frontend using backend scripting to simulate real-time order processing. Real-World Scenario: Shopping cart, inventory updates, and order placement in an online store. **Sub Objectives** • Create normalized schema for Users, Products, Orders, and Payments. • Implement SQL queries and triggers for inventory management. • Connect database to frontend using Python Flask / PHP / Node.js. • Test data insertion, deletion, and querying through a web form or API call. Learning Focus: Backend connectivity, data manipulation via UI/API, frontend integration.	CO 3
4	**Advanced Queries & Access Control – Movie Ticket Booking System** **Objective:** To work with complex queries, user input handling, and enforce access rules using SQL constraints and roles. Real-World Scenario: Online platform for cinema seat booking and user registration. **Sub Objectives** Create schema for Movies, Theatres, ShowTimings, Users, and Bookings. • Write nested queries and joins to fetch available seats by date and time. • Apply constraints to prevent double booking and overbooking. • Implement basic role-based access (e.g., Admin vs. Customer roles).	CO4

	Learning Focus: Nested queries, referential integrity, joins, constraints, user role management.	
5	**CAPSTONE PROJECT: Employee Payroll & Attendance Management System** **Objective:** To integrate all learned concepts into a full-stack DBMS application simulating a secure payroll and HR management system. Real-World Scenario: Payroll generation and attendance tracking system for a mid-sized IT firm. Project Tasks: • Design a relational schema including Employees, Attendance, Leave, Salary, and Payroll. • Write stored functions and procedures for salary calculation, tax deductions, and monthly slip generation. • Implement complex queries to generate reports: monthly attendance, salary slips, late marks. • Develop a basic role-based web or CLI interface where HR/admin can manage employees and employees can view their records. Learning Focus: Schema design, functions, report generation, UI integration, full lifecycle DB operations, access control.	CO3,CO 4

Learning Experiences

Inside Classroom Learning:

1. **Interactive Concept Lectures**: Use whiteboard and slide presentations to explain DBMS fundamentals, ER modeling, normalization, SQL queries, and transaction management with real-world analogies.
2. **ER Modeling Workshops**: In-class group activities using tools like Lucidchart or dbdiagram.io to create ER diagrams for systems like Library or Hospital Management.
3. **SQL Hands-On Labs**: Execute DDL, DML, joins, subqueries, and PL/SQL procedures using MySQL Workbench or pgAdmin in lab sessions.

4. **Normalization & Schema Design Practice**: Classroom problem-solving sessions on converting unnormalized tables into 1NF–BCNF with peer discussion and instructor walkthroughs.
5. **Case-Based Discussions**: Explore use cases of database design and performance in banking, e-commerce, and healthcare through guided case studies.

Outside Classroom Learning Experience

1. **Tool-Based Assignments**: Use Lucidchart, dbdiagram.io, and MySQL/PostgreSQL at home to design and convert ER models into normalized schemas.
2. **Self-Guided Projects**: Build mini-databases for systems like e-commerce, social media, or library management with full schema and SQL query files.
3. **Online Platforms Practice**: Practice SQL and PL/SQL on platforms like LeetCode, HackerRank, and W3Schools to reinforce joins, subqueries, and window functions.
4. **Industry Simulation Tasks**: Analyze customer or transaction data using advanced SQL on sample datasets mimicking retail or finance environments.
5. **Video-Based Explorations**: Watch curated YouTube/OCW videos on query optimization, NoSQL (MongoDB/Firebase), and cloud databases like AWS RDS and Google BigQuery.

Text and Reference Book

1. Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). *Database system concepts* (7th ed.). McGraw-Hill.
2. Pratt, P. J., & Last, M. Z. (2020). *Concepts of database management* (10th ed.). Cengage Learning.

Design and Analysis of Algorithms

Programme Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Design and Analysis of Algorithms**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCDA302	**3-0-2**	**4**	**45**
Type of Course:	**Major**			
Pre-requisite(s), if any: Basic knowledge of understanding the concepts of Data structure				

Course Perspective: This course provides an in-depth theoretical and practical understanding of algorithmic complexity analysis and advanced data structures. The focus is on analyzing algorithm efficiency and implementing real-world problem-solving techniques.

The course emphasizes complexity analysis, divide & conquer, greedy methods, dynamic programming, graph algorithms, NP-completeness, and advanced tree-based data structures.

Course Outcomes (CO)

COs	**Statements**
CO1	**Analyzing** and compare algorithm efficiency using asymptotic notation and mathematical proofs.
CO2	**Implementing** divide and conquer, greedy, and dynamic programming techniques for problem-solving.
CO3	**Utilizing** advanced data structures such as tries, Fibonacci heaps, B+ trees, and binomial heaps in algorithm design.
CO4	**Understanding** NP-completeness, approximation algorithms, and parallel processing techniques for large-scale computations.

CO5	**Solving** real-world computational problems using advanced algorithmic techniques and coding.

Course Outline:

Unit Number: 1	**Title: Complexity Analysis & Fundamental Algorithms**	**No. of hours: 12**

Content Summary:

- Mathematical Foundations of Algorithm Analysis – Growth of functions, Recurrence Relations, Master Theorem.
- Asymptotic Notations – Big-O, Omega, Theta, Complexity Classes.
- Sorting Algorithms & Complexity – Merge Sort, Quick Sort, Heap Sort, Counting Sort, Radix Sort.
- Searching Algorithms – Binary Search, Interpolation Search, Hashing Techniques.
- Amortized Analysis & Advanced Complexity Considerations.
- Randomised algorithms: QuickSort/QuickSelect, min-cut (Karger), Monte-Carlo vs Las-Vegas, universal hashing.
- Cache-aware thinking: memory hierarchy, locality, introductions to cache-oblivious algorithms.

Real-World Use Cases:

✓ Algorithmic Trading – Optimizing financial transactions with fast sorting and searching techniques.
✓ Big Data Processing – Efficient sorting and searching in large-scale databases.
✓ Performance Engineering – Improving website load times using efficient algorithms.

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

Content Summary:

- Divide & Conquer Techniques – Binary Search, Closest Pair of Points, Convex Hull.
- Greedy Algorithms – Huffman Encoding, Activity Selection, Kruskal's & Prim's Algorithm.
- Dynamic Programming – Tabulation and memorization, 0/1 Knapsack, Matrix Chain Multiplication, Longest Common Subsequence, Floyd-Warshall Algorithm.

- Complexity Analysis of Recursive Algorithms.

Real-World Use Cases:

✓ AI & Machine Learning – Optimization techniques for training deep learning models.
✓ Data Compression Algorithms – Huffman encoding in file compression (ZIP, JPEG, MP3).
✓ Cloud Network Routing – Shortest path optimization for real-time traffic management (Google Maps, Uber, Waze).

Unit Number: 3	**Title: Graph Algorithms & Advanced Data Structures**	**No. of hours: 11**

Content Summary:

- Graph Traversal Algorithms – BFS, DFS, Strongly Connected Components.
- Minimum Spanning Trees (MSTs) – Kruskal's Algorithm, Prim's Algorithm.
- Shortest Path Algorithms – Dijkstra's, Bellman-Ford, Floyd-Warshall.
- Advanced Data Structures – Trie, B-Trees, B+ Trees, Skip Lists, Splay Trees.
- Heap-Based Structures – Binomial Heaps, Fibonacci Heaps, Complexity Analysis.

Real-World Use Cases:

✓ Cybersecurity & Network Forensics – Graph-based intrusion detection & anomaly detection.
✓ Database Indexing (MySQL, MongoDB) – Trie, B+ Trees used in indexing large-scale datasets.
✓ Blockchain & Cryptography – Data structures used in ledger verification & encryption.

Unit Number: 4	**Title:NP-Completeness, Approximation Algorithms, & Parallel Processing**	**No. of hours: 12**

Content Summary:

- P, NP, NP-Hard & NP-Complete Problems – Traveling Salesman Problem (TSP), Graph Coloring.
- Backtracking & Branch and Bound – N-Queens, Hamiltonian Cycle.
- Approximation Algorithms – Vertex Cover, Set Cover, TSP Approximation.
- Parallel Processing & MapReduce – Introduction to parallel computing models.
- **Algorithm engineering**: profiling (perf, gprof, py-spy), technical-debt-driven refactoring, competitive-programming heuristics.

Real-World Use Cases:

✓ Genomic Data Analysis (Bioinformatics) – DNA sequence alignment using approximation algorithms.
✓ Optimizing Cloud Computing Costs – NP-hard resource allocation problems in AWS, Google Cloud.
✓ High-Speed Internet & Network Routing – Graph algorithms in routing protocols (BGP, OSPF).

Inside Classroom Learning:

1. **Whiteboard Problem Solving**: In-depth walkthroughs of complexity analysis, recurrence relations, and asymptotic notation using real-world-inspired problems.
2. **Algorithm Design Workshops**: Group coding sessions in class to implement divide & conquer, greedy, and dynamic programming approaches (e.g., Knapsack, Convex Hull).
3. **Graph Algorithms Walkthroughs**: Step-by-step board and projector demos of graph traversal (BFS/DFS), MSTs, and shortest paths using real-world network mapping scenarios.
4. **Data Structure Demonstrations**: Classroom implementation and comparison of advanced structures like Trie, B+ Trees, and Fibonacci Heaps with performance analysis.
5. **Concept Integration Discussions**: Instructor-led talks on NP-completeness, approximation algorithms, and parallel processing, tied to real-world problems (e.g., TSP, cloud cost optimization).

Outside Classroom Learning Experience

1. **Take-Home Coding Challenges**: Implement and analyze complex algorithms (e.g., randomized min-cut, matrix chain multiplication, heap operations) using Python/C++.
2. **Tool-Based Practice**: Use profiling tools (e.g., gprof, py-spy) to measure performance of implemented algorithms and understand time/space trade-offs.
3. **Mini-Projects**: Build small applications such as a Huffman compressor, shortest path navigator, or DNA sequence matcher using algorithmic techniques learned in class.

4. **Online Problem Solving Platforms**: Practice advanced algorithmic problems on platforms like LeetCode, Codeforces, and HackerRank to reinforce lecture content.
5. **Research & Industry Use Case Exploration**: Study and report on how algorithms and data structures are used in domains like machine learning, cybersecurity, or bioinformatics.

Text Books:

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (3rd ed.). MIT Press.
2. Skiena, S. (2020). *The Algorithm Design Manual* (3rd ed.). Springer.

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Algo Toolbox & Profiler** **Problem statement:** Build a command-line tool that chooses the fastest sort and search strategy for a given array size and data distribution. 1. Implement & time o Merge Sort, Counting Sort, and Randomized QuickSort (just the standard pivot-randomization). 2. Search module o Binary Search vs. Hash-Table lookup on the same dataset. 3. Mini-report o Plot running times for n = 10^4, 10^5, 10^6; state the observed Θ notation and one sentence on cache behavior you noticed. *Data provided:* three CSV files (random, nearly-sorted, reverse).	CO 1
2	**Greedy vs DP Optimizer** **Problem statement:** For a small warehouse, decide what items can go in a delivery box while also compressing its packing slip. 1. 0/1 Knapsack	CO 2

	○ Implement *Greedy* by value/weight and the exact *DP* solution; compare total value achieved on ≤ 200 items. 2. Huffman Coding ○ Generate the binary codes for the packing-slip text and output compressed size. 3. Reflection ○ ½-page note: when did greedy fail and why did DP succeed?	
3	**Graph Navigator Lite** **Problem statement:** Create a simple bike-route finder for a small city graph (≤ 5 000 nodes). 1. Graph core ○ Read edge list, store it, and implement BFS to list connected components. 2. Shortest path ○ Write Dijkstra using a binary heap; output distance & path between two input intersections. 3. Search-assist ○ Build a Trie for street-name auto-completion (max 10 suggestions).	CO 3
4	**Approximate & Parallel Solver** **Problem statement:** Speed-plan a salesperson's daily tour of 15 customers on a multi-core laptop. 1. TSP Approximation ○ Code the simple *Nearest-Neighbour* heuristic and report tour length vs. optimal (use brute force for n ≤ 10 to estimate optimum). 2. Backtracking Demo ○ N-Queens for n = 8 with pruning; print one solution and node-count explored. 3. Parallel Bonus ○ Implement *parallel* QuickSort on 1 million ints using language threads; show speed-up vs. serial.	CO2,C03
5	**CAPSTONE PROJECT: Mini-Courier Planner** **Problem statement:** Combine what you have learned to build a prototype that, given a list of parcels and delivery points, outputs an *ordered* drop-off route and a packing plan.	CO3,CO 4

	Sub-tasks (tie earlier work together): 1. Package Prioritiser o Sort parcels by deadline using the fastest algorithm discovered in Project 1. 2. Box Filler o Use your DP knapsack to decide which parcels fit in one ride. 3. Route Maker o Build customer graph, run Dijkstra for individual legs, then apply your TSP approximation for overall route. 4. Performance snapshot o One chart comparing runtime of each major step; one paragraph on possible future optimisations	

VERBAL ABILITY

Program Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **Verbal Ability**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		2-0-0	2	**30**
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		

Learning Experiences

Inside Classroom Learning*:*

1. **Vocabulary Games & Quizzes**: Use crosswords, word ladders, and timed quizzes in class to teach root words, synonyms, antonyms, and idioms.

2. **Grammar Drill Sessions**: Interactive blackboard activities and group exercises on tenses, parts of speech, subject-verb agreement, and sentence types.
3. **Sentence Correction Workshops**: Practice sessions where students identify and correct grammatical errors and improve sentence construction.
4. **Cloze Tests & Spot-the-Error Practice**: In-class completion and discussion of cloze passages and sentence correction tasks with peer evaluation.
5. **Reading Comprehension with Discussion**: Analyze short passages followed by Q&A focusing on context clues, sentence arrangement, and analogy-based questions.

Outside Classroom Learning*:*

1. **Vocabulary Builder Journal**: Maintain a daily vocabulary diary with root words, commonly confused words, idioms, and one-word substitutions with examples.
2. **Online Grammar Practice**: Use tools like Grammarly, British Council, or Cambridge Grammar for self-paced exercises on tenses, clauses, and pronouns.
3. **Sentence Accuracy Worksheets**: Take-home assignments focusing on spotting errors, sentence improvement, and rearrangement tasks.
4. **Cloze & RC Apps**: Practice comprehension and cloze tests on platforms like Textbook, Magoosh, or Read Theory for independent learning.
5. **Peer Vocabulary Sharing Groups**: Create small WhatsApp/Google Classroom groups where students share 2 new words daily with meaning and usage.

Textbooks

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

COMPETITIVE CODING -I

Program Name	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: COMPETITIVE CODING**-I**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		2-0-0	2	30
Type of Course:	SEC			
Pre-requisite(s), if any: Fundamentals of programming				

Course Perspective: This course enhance students' problem-solving abilities in competitive coding by providing in-depth knowledge of core data structures, algorithms, and efficient coding techniques. This course aims to prepare students for technical assessments and coding interviews, building a strong foundation for tackling real-world coding challenges.

Course Outcomes

CO1	**Applying** fundamental and advanced coding techniques to solve problems involving arrays, strings, recursion, matrices, and linked lists.
CO2	**Analyzing** and implementing efficient data structure operations, including stacks, queues, and their real-world applications in competitive programming.
CO3	**Evaluating** and optimize problem-solving approaches through comprehensive understanding and revision of key concepts from previous sessions.

SESSION WISE DETAILS		
Session: 1	Introduction to competitive programming	No. of hours: 2
Content Summary:		

Introduction to LeetCode and Codechef coding platforms, Overview of competitive programming, setting up environment, approach to problem solving		
Session: 2	Array I	No. of hours: 2
Content Summary Reversing the array, finding maximum and minimum elements, Running sum of 1d Array, count elements with maximum frequency , left/right rotate an array by k positions.		
Session: 3	Array II	No. of hours: 2
Content Summary: find element in an array, Remove duplicate elements from an sorted array, find repeating element an array, find equilibrium element in an array.		
Session: 4	Array's Sorting and Time and space complexity Analysis	No. of hours: 2
Content Summary: Bubble sort, selection sort, Insertion Sort and complexity Analysis		
Session: 5	Array III	No. of hours: 2
Content Summary: union and intersection of sorted arrays, maximum subarray sum (Kadane's Algorithm), maximum product subarray(based on Kandane's) , majority Element (moore's voting algorithm)		
Session: 6	Strings I	No. of hours: 2
Content Summary: check given string is palindrome or not, count number of vowel and consonant, remove character except alphabet.		
Session: 7	String II	No. of hours: 2
Content Summary: Calculate frequency of a character, print maximum occurring character in a string, Remove duplicate character from a string, count number of word in a string		
Session: 8	Recursion I	No. of hours: 2

Content Summary: find factorial, find power of a number, (printing increasing, decreasing and Decreasing Increasing), count digit, sum of array using recursion		
Session: 9	Recursion II	No. of hours: 2
Content Summary: find pivot index, remove duplicates, fibonacci number, tower of hanoi with recursion tree presentation,		
Session: 11	Matrix Problems I	No. of hours: 2
Content Summary: Spiral traversal, searching elements in a matrix, Printing elements in sorted order.		
Session: 12	Matrix Problems II	No. of hours: 2
Content Summary: Finding median in row-wise sorted matrix, identifying rows with maximum 1s , rotating matrices by 90 degrees.		
Session: 13	LinkedList Introduction.	No. of hours: 2
Content Summary: add Node on any position, delete Node from given position, search Node in a linked List, Count Node in linked List		
Session: 14	LinkedList I	No. of hours: 2
Content Summary: reverse LinkedList, find mid of the linkedList, Merge Two sorted LinkedList.		
Session: 15	LinkedList II	No. of hours: 2
Content Summary: add two number, rotate list, remove duplicates from sorted list		
Session: 16	Stack Implementation	No. of hours: 2
Content Summary: Stack Implementation using Array, Next Greater Element		
Session: 17	Stack I	No. of hours: 2

Content Summary: Smaller element on left, valid parentheses, Evaluate postfix expression		
Session: 18	Stack II	No. of hours: 2
Content Summary: min stack, asteroid collision, stock span problem		
Session : 19	Queue Introduction.	No. of hours: 2
Content Summary: Queue implementation using array, Implement circular queue, queue using stack		
Session :20	Summary	
Content Summary: Revising the completed topics and company specific problems on given topics.		

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.

Reference Books:

- ***Programming Challenges*** – Steven Skiena & Miguel Revilla

  A gentle introduction to algorithmic problem solving with problems and detailed solutions.
- **Competitive Programming (3rd Edition)** – Steven Halim & Felix Halim

Widely recommended for ICPC preparation. Covers data structures, algorithms, and contest strategies.

Online Resources:

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

Evaluation Criteria

Criteria	**Internal (50 marks)**
After 2 weeks Coding Test	**No. of tests: 6** **Marks per test: 5** **Total marks: 30**
Mid Term Coding Test	**20 Marks**

External (50 marks)
End Term Paper

Summer Internship-I

Program Name	B.Tech (CSE) with specialization in UI/UX		
Course Name: Summer Internship-I	Course Code	L-T-P	Credits
	ETCCIN305	0-0-4	2
Type of Course:	INT		
Pre-requisite(s), if any: NA			

The Summer Internship Program (1st June – 31st July) is designed to integrate academic learning with real-world professional experiences, enabling students to apply theoretical knowledge to practical situations. It forms a mandatory part of the Semester III for students currently in Semester II, carrying a weightage of **2 academic credits**.

The key objectives of the Summer Internship Program are:

- To enhance professional skills and industry readiness.
- To expose students to real-world technical, managerial, and research practices.
- To promote self-learning, professional responsibility, and critical thinking.
- To foster connections between academic knowledge and industry practices.

Duration

The duration of the internship will be 6-8 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 (Online/Offline):**
   Students must produce a joining letter at the start and a relieving letter upon completion.

2. **Global Certifications:**
   Students can opt for globally recognized certification programs relevant to their field of study.
3. **Government/Research Institution Internship:**
   Students can engage in a research internship with premier government or research organizations such as IITs, IISc, ISRO, DRDO, CSIR, NPL, etc.
4. **On-Campus Industry Internship Programs:**
   The university will offer on-campus internships in collaboration with industry partners.

Deliverables and Documentation:

Each student must submit the following after completing their internship/certification:

Deliverable	Description	Marks
Summer Internship File	A detailed report/file based on the provided format including objectives, methodology, learnings, and reflections.	10 Marks
Video Presentation	A 7–10-minute recorded video presentation showcasing work done during the internship/certification. The template of slides will be shared.	20 Marks
Certificate of Completion	A color-printed certificate on bond paper from the host organization/certification body, mentioning duration, role/project.	70 Marks

Evaluation Metrics

The Summer Internship will be evaluated based on the following comprehensive criteria:

Evaluation Component	Weightage	Description
Internship Report/File	10%	Completeness, professional formatting, relevance to internship tasks.
Video Presentation	20%	Content quality, clarity, communication skills, professional presentation.

Certificate of Completion	70%	Authenticity, completion of internship/certification within stipulated time, relevance to program objectives.

Internship Evaluation Rubric:

S. No.	Component	Sub-Component / Criteria	Marks
1	**Internship Certificate**	**Relevance to Core Subjects**	**20 Marks**
		- Directly relates to core subjects	20
		- Partially relates to core subjects	15
		- Minimally relates to core subjects	10
		- Not relevant	0
2	**Report Submission**	**Structure and Organization**	**10 Marks**
		- Well-structured and organized report	10
		- Moderately structured report	7
		- Poorly structured report	3
		- No structure	0
3	**Solo Video-Based Evaluation**	**a. Technical / Professional / Soft Skills Acquired**	**10 Marks**
		- Highly relevant and advanced technical skills	10
		- Moderately relevant technical skills	8
		- Basic technical skills	5
		- No new skills acquired	0
		b. Content Delivery	**10 Marks**
		- Clear, engaging, and thorough delivery	10
		- Clear but less engaging delivery	7
		- Somewhat clear and engaging delivery	3
		- Unclear and disengaging delivery	0
		c. Visual Aids & Communication Skills	**10 Marks**
		- Effective visual aids + excellent communication skills	10
		- Moderate visual aids + good communication skills	7
		- Basic visual aids + fair communication skills	3
		- No visual aids + poor communication skills	0
4		**Weeks Completed**	**10 Marks**

	Internship Duration	- 6–8 weeks completed	10
		- 4–6 weeks completed	8
		- Less than 1 month	5
5	**Outcome of the Internship**	**Application / Project / Key Learnings & Findings**	**30 Marks**
		- Clear, outcome-based project with applied learnings and key findings	25–30
		- Moderate outcome with partial application and findings	15–24
		- Minimal outcome, unclear learning/application	0–14

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 intern- ship 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, team- work, time management, and adaptability in a professional setting.
- **Present Findings Effectively:**

- Deliver clear and engaging presentations to effectively communicate project outcomes, research findings, and acquire 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.

COMMUNITY SERVICE

Program Name	B.Tech (CSE) with specialization in UI/UX			
Course Name: Community Service	Course Code	L-T-P	Credits	Contact Hours
		1-0-0	1	15
Type of Course:	CS			

Course Perspective. The Community Engagement 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.

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

COs	Statements
CO 1	To **engaging** students in meaningful social service activities.
CO 2	To **developing** socially responsible engineers
CO 3	To **applying** technical and non-technical skills for the benefit of society.
CO 4	To **fostering** community engagement and support.

Course Outline:

1. Introduction

Overview of the Course: The Community Engagement 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.
1. **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.

2. **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.

Conclusion:

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

Student Report Template

Title Page:

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

1. Introduction:

- Overview of the Course: Provide a brief overview of the Community Engagement Service (VAC II) course, highlighting its purpose and importance.
- Importance of Social Service in Engineering Education: Discuss why incorporating social service into engineering education is crucial for developing well-rounded professionals.

- Expectations and Requirements: Outline the course expectations, including participation, documentation, and reporting requirements.

2. Chosen Activity:
   - Activity Name: State the name of the chosen social service activity.
   - Description of the Activity: Provide a detailed description of the activity.
   - Objectives and Goals: List the objectives and goals of the activity.
3. Methodology:
   - Steps Taken: Describe the steps taken to complete the activity.
   - Tools and Techniques Used: Mention any tools or techniques used, such as mobile apps, web-based platforms, etc.
   - Duration of Engagement: Specify the duration of the engagement (at least 30 hours).
4. Implementation:
   - Detailed Description of Engagement Activities: Provide a detailed log of the engagement activities, including day-wise descriptions.
   - Proof of Engagement: Include video clips, photographs, and other relevant proofs of engagement.
5. Impact Analysis:
   - Impact on Society: Analyze the impact of the activity on society.
   - Benefits to the Community: Discuss the benefits provided to the community.
   - Testimonials from Beneficiaries: Include testimonials from beneficiaries showcasing the outcomes and benefits.
6. Learning Experiences:
   - Skills and Knowledge Gained: Detail the skills and knowledge gained through the activity.
   - Reflections on the Experience: Reflect on the overall experience.
   - Challenges Faced and Overcome: Describe any challenges faced and how they were overcome.
7. Ethical Considerations:
   - Ethical Issues Encountered: Discuss any ethical issues encountered during the activity.
   - Solutions and Best Practices: Provide solutions and best practices for addressing these ethical issues.

- Reflections on Social Responsibility: Reflect on the importance of social responsibility.

8. Conclusions:

- Summary of the Experience: Summarize the overall experience.
- Personal Growth and Development: Discuss personal growth and development resulting from the activity.
- Future Recommendations: Provide recommendations for future engagements.

9. Appendices:

- Additional Documents and Proofs: Include any additional supporting documents, such as logbook entries and extra photographs.
- Video Presentation Link: Provide a link to the video presentation.

SEMESTER: IV

NAND to Tetris-II

Program Name	B.Tech (CSE) with specialization in UI/UX			
Course Name: NAND to Tetris-II	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCNT401	3-0-2	4	45
Type of Course:	Major			
Pre-requisite(s):				

Course Perspective. Extending the hardware platform, students implement a full software stack: a Virtual Machine (VM), a compiler for the Jack language, and core OS libraries. The course culminates in a cross-compiled game or application executing on the self-built Hack computer.

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

COs	Statements
CO 1	Translating stack-based VM commands to optimized Hack assembly.
CO 2	Constructing a two-phase compiler (Jack → VM) with symbol management and static typing.
CO 3	Implementing memory, I/O, and math services forming a minimalist operating system.
CO 4	Integrating, debuging, and profile high-level Jack applications on the Hack platform.

CO 5	Practicing collaborative software engineering (issues, pull requests, code review).

Course Outline:

Unit Number: 1	**Virtual Machine Architecture & Translator**	**No. of hours: 10**
Content: ▪ VM command taxonomy (arithmetic, memory access, branching, function call) ▪ Call stack layout; frame pointer maintenance; recursion support ▪ Code-generation patterns for VM → ASM (template method) ▪ **Industrial parallel:** JVM byte-code & Web Assembly comparison		
Unit Number: 2	**Compiler Front-End (Jack Language)**	**No. of hours: 12**
Content: ▪ Lexical analysis (token regex, finite-state scanners) ▪ LL(1) Recursive-descent parsing; parse tree vs AST ▪ Static vs field vs subroutine symbol tables; scope handling ▪ Semantic checks and error-reporting architecture		
Unit Number: 3	**Compiler Back-End & Optimization**	**No. of hours: 12**
Content: ▪ VM-code emission rules for expressions, control flow, methods ▪ Object model: 'this', dynamic dispatch, constructor code ▪ Peephole optimizations: constant folding, dead push/pop elimination ▪ Pipelining compilation with Make/CMake & continuous tests		
Unit Number: 4	**Operating System & High-Level Applications**	**No. of hours: 11**

Content:

- Implementing Jack-level OS classes: Memory, Array, String, Math, Screen, Keyboard, Sys
- Event-loop pattern, basic graphics primitives, sprite animation
- Profiling & performance tuning using CPU emulator statistics
- Packaging applications: ROM image, documentation, demo showcase

Classroom Learning Experience

Inside Classroom Learning:

1. **Concept-Driven Lectures:** Use slide decks and board work to explain core ideas of stack-based virtual machines, compiler stages (front-end & back-end), and operating system design.
2. **Code Translation Sessions:** Translate Virtual Machine (VM) commands to Hack Assembly step-by-step in class to demonstrate template-based code generation and stack manipulation techniques.
3. **Compiler Construction Labs:** Guide students through implementing lexical analyzers, parsers, and code generators using Jack language, with live coding demonstrations and recursive-descent parsing examples.

Outside Classroom Learning Experience

1. **Stage-Wise Project Implementation:** Students complete milestones of the full-stack project independently — including building the compiler, implementing OS libraries, and developing final applications.
2. **Use of Software Tools & Automation:** Practice automated compilation and testing using tools like Make, CMake, and Git for version control and continuous integration of compiler and OS components.
3. **Peer Collaboration via Version Control:** Collaborate through platforms like GitHub for issue tracking, pull requests, and code reviews, simulating real-world software engineering workflows.

4. **Test-Driven Development:** Write unit and integration tests for compiler modules and OS functions, using emulator tools to verify output correctness and program behavior.

Textbooks:

Nisan, Noam, and Shimon Schocken. The Elements of Computing Systems: Building a Modern Computer from First Principles. MIT Press, 2005.

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab 1 – Stack-VM Translator v1.0** • Implement arithmetic & logical VM commands → ASM. • Handle push/pop for constant, local, argument, temp segments. • Develop automated testing harness comparing emulator traces. • Provide complexity report: instruction count per command class. • Sub-task Extension: Implement a simple profiler within the VM translator that counts the execution frequency of each VM instruction in a given program, to identify performance hotspots.	CO 1
2	**Lab 2 – VM Translator v2.0 (Control & Functions)** • Add branching (label, goto, if-goto) with unique label generator. • Implement call/return mechanism; support nested & recursive calls.	CO 2

	• Integrate with CI: every push triggers emulator regression suite. • Sub-task Extension: Analyze the generated assembly code for recursive functions and discuss the stack frames' growth and unwinding, relating it to potential stack overflow issues. • Stretch: inline simple functions to reduce call overhead.	
3	**Lab 3 – Jack Compiler – Front End** • Tokenizer: spec-compliant XML & JSON token streams for debugging. • Parser + AST dump; unit tests covering all Jack grammar rules. • Symbol table: class vs subroutine vs static/field/var handling. • Semantic validator: type-checking, undeclared identifiers, call-arity mismatch. • Sub-task Extension: Implement a basic pretty-printer that takes the AST and prints a syntactically correct, formatted Jack code, demonstrating the AST's utility.	CO 3
4	**Lab 4 – Jack Compiler – Back End & OS Libraries** • VM-code generator producing human-readable commentary. • Optimization pass: remove redundant push-pop pairs. • Implement OS class subset: Memory.alloc/deAlloc, String.new, Screen.drawPixel. • Integration test: compile and run TetrisCore.jack logic on your tool-chain. • Sub-task Extension: Investigate memory usage of compiled Jack programs. Analyze if memory fragmentation could be an issue with your Memory.alloc/deAlloc implementation and propose basic solutions.	CO 3
5	**Lab 5 – Capstone: Full-Stack Jack Game / App**	CO 4

	• Design phase: propose & storyboard a game or GUI utility (e.g., "Hack-Paint", "Breakout", "2048"). • Build phase: author in Jack; compile with your compiler; run on your VM/OS. • Performance phase: gather FPS / CPU-cycle stats; iterate for speed. • Sub-task Extension: Implement a basic debugger feature in the VM emulator or compiler, allowing breakpoints or step-by-step execution to understand program flow at the VM level. • Sub-task Extension: Discuss potential security vulnerabilities that could arise at the VM or OS level (e.g., direct memory access, buffer overflows, if applicable to the Hack architecture) and conceptual mitigation strategies. • Release phase: deliver GitHub repo + 5-minute video + technical report. • Evaluation rubric: correctness 35 % · performance 15 % · code quality 15 % · innovation 15 % · documentation/demo 20 %.	

(Specialization Course-II)

User Research & Information

Architecture

Programme Name:	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: **User Research & Information Architecture**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETUXIA402	3-0-2	4	45
Type of Course:	DSE			

Course Perspective: This course provides students with hands-on skills to plan, conduct, analyze, and apply user research insights to design intuitive information architectures for digital products. Through practical engagement with real users and data, students will learn how to synthesize research into actionable insights, create user personas, build navigation systems, conduct card-sorting activities, and apply usability testing principles. Emphasis is placed on industry-standard practices and tools like Miro, Figma, Optimal Workshop, and Notion. By the end of the course, students will be equipped to work in roles such as UX Researcher, Information Architect, or Usability Analyst.

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

COs	Statements
CO 1	Conducting user research design using qualitative and quantitative methods.
CO 2	Translating research findings into personas, user journeys, and mental models.
CO 3	Creating effective information architecture using sitemaps, labeling, and taxonomy.
CO 4	Applying IA techniques like card sorting, tree testing, and content auditing.

CO 5	Testing and iterate on IA and navigation systems based on user feedback

Course Outline:

Unit Number: 1	**Title: Introduction to User Research in UX**	**No. of hours: 13**
Objective: Understand user research types, ethics, and tools used in real-world settings. **Contents:** • Importance of user research in the UX process • Types of research: generative vs evaluative • Qualitative methods: interviews, observation, contextual inquiry • Quantitative methods: surveys, analytics basics • Research planning: framing questions, selecting users, ethics • Data collection and synthesis overview **Hands-On:** • Write a basic research plan for a chosen digital product • Conduct and record 2 peer interviews • Create a research documentation template in Notion		
Unit Number: 2	**Title: Personas, Scenarios & Journey Mapping**	No. of hours: 12
Objective: Synthesize data into user models and design artifacts. **Contents:** • Affinity diagramming for theme extraction • Creating detailed personas (background, needs, frustrations) • Writing user scenarios and storyboards • Creating empathy maps and user journey maps • Jobs-To-Be-Done framework • Representing user mental models **Hands-On:** • Create 2 personas based on interview insights • Build journey maps for each persona • Represent user mental models using Miro or FigJam		
Unit Number: 3	**Title: Information Architecture & Navigation Design**	No. of hours: 10

Objective: Design user-centered content structures and test them.

Contents:

- Definition and goals of Information Architecture (IA)
- Content inventory and audit techniques
- Card sorting methods: open, closed, hybrid (Miro/Optimal Workshop)
- Creating taxonomy, labeling systems, and categories
- Sitemap and navigation design (flat vs hierarchical structures)
- Basics of testing

Hands-On:

- Conduct a content audit for a well-known website (e.g., college portal)
- Conduct a card-sorting exercise using Optimal Workshop
- Build a sitemap and labeling system based on findings

Unit Number: 4	**Title: Cloud Deployment, Automation, and Monitoring**	No. of hours: 10

Objective: Translate IA into wireframes and evaluate usability.

Contents:

- Translating IA into low/high fidelity wireframes
- Wireframing tools: Figma/Balsamiq
- Introduction to usability testing methods
- Designing usability test scenarios and tasks
- Conducting tests: task-based, think-aloud method
- Analyzing results and documenting insights

Hands-On:

- Create wireframes for a 5-page site structure
- Design a usability test plan and task sheet
- Conduct 2 peer usability tests and log observations

List of Experiments

Ex. No	Experiment Title	Mapped

		CO/COs
1	**Lab Assignment 1 (Unit 1):** Title: "Investigating Student Pain Points in the Hostel Complaint System" Sub-Objectives: 1. Plan and conduct a short user research study (interviews + survey). 2. Frame key research questions and collect meaningful data. 3. Apply qualitative and quantitative analysis to extract insights. 4. Document themes and initial findings using affinity mapping. Present findings using a simple research deck	CO 1
2	**Lab Assignment 2 (Unit 2):** Title: "Designing for First-Time Users of a Public Transport App" Sub-Objectives: 1. Build 2 user personas from real or mock research data. 2. Write realistic user scenarios based on public transport usage. 3. Develop empathy maps and journey maps for each persona. 4. Identify problem touchpoints and opportunities in the journey. 5. Present all artifacts visually using Miro or Notion.	CO 2
3	**Lab Assignment 3 (Unit 3):** Title: "Restructuring the Information Architecture of a Department Website" Sub-Objectives: 1. Audit an existing college website's IA (structure + content). 2. Conduct a card sorting session with 3–5 users. 3. Analyze data and extract grouping patterns. 4. Design a new sitemap and taxonomy based on user mental models.	CO 3

4	Lab Assignment 4 (Unit 4): Title: "Wireframing and Testing the Navigation of a Local Job Portal" Sub-Objectives: 1. Use the IA structure to design 4–5 wireframes in Figma. 2. Create navigation menus and page content layouts. 3. Prepare a usability test plan with clear scenarios and tasks. 4. Conduct 2 usability tests and log user behavior. 5. Refine wireframes based on usability insights and justify changes.	CO3,CO 4
5	**Capstone Project: Redesigning the Digital Experience of a University Student Services Portal** **Objectives:** 1. Plan and conduct user research using interviews and surveys to uncover usability issues and student pain points. 2. Synthesize research insights into personas, empathy maps, and journey maps to represent diverse student needs. 3. Create a logical and user-centered information architecture using content audits, card sorting, and sitemap design. 4. Design low- to mid-fidelity wireframes of key portal screens based on the finalized IA structure. 5. Conduct usability testing, gather feedback, and iterate designs to enhance navigation and content clarity.	CO 4

Learning Experience

In-Class Learning

- Interactive lectures with case-based discussions on user-centered design.
- Step-by-step guidance on research planning, persona creation, IA structuring, and usability testing.

- Tool demos using Miro, Notion, Figma, and Optimal Workshop.
- Regular peer critiques and guided synthesis activities to refine research artifacts.

Hands-On Labs & Practice

- Real-time application of methods such as interviews, card sorting, journey mapping, and wireframing.
- Iterative design tasks grounded in real or simulated user data.
- Weekly assignments that progressively build toward a complete IA and UX prototype.

Beyond the Classroom

- Field/user engagement for authentic data collection (e.g., hostel system, transport app).
- Observational research and usability tests with real users or peers.
- Capstone project promoting end-to-end thinking—from research to usability validation.

Text and Reference Book

Russ Unger & Carolyn Chandler (2019). A Project Guide to UX Design: For User Experience Designers in the Field or in the Making (3rd ed.). New Riders.

Cloud Computing & DevOps

Programme Name:	**B.Tech (CSE) with specialization in UI/UX**			
Course Name: Cloud Computing & DevOps	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCCD403	3-0-2	4	45
Type of Course:	Major			
Pre-requisite(s), if any: Programming concepts, and familiarity with command-line interfaces to effectively engage with cloud platforms and DevOps tools				

Course Perspective: This course introduces the foundational concepts of cloud computing and the key principles and practices of DevOps. Students will explore various service and deployment models, virtualization, containerization (with Docker), container orchestration (Kubernetes basics), and automation through CI/CD pipelines. The course emphasizes hands-on skills using cloud platforms (AWS, Azure, GCP), command-line interfaces, infrastructure as code, and DevOps tools to enable efficient deployment, scaling, and monitoring of cloud-native applications.

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

COs	Statements
CO 1	**Understanding** the fundamental concepts of cloud computing, service models, and deployment models, utilize virtualization and containerization technologies, including Docker.
CO 2	**Explaining and applying** DevOps principles, including CI/CD pipeline practices.
CO 3	**Deploying and managing** applications on major cloud platforms using automation and infrastructure-as-code tools.

Course Outline:

Unit Number: 1	Title: Foundations of Cloud	No. of hours: 13

Content:

- Cloud computing definition, characteristics, benefits, challenges
- Cloud service models: IaaS, PaaS, SaaS
- Deployment models: Public, Private, Hybrid, Community
- Cloud infrastructure components: data centers, servers, storage, networking
- Introduction to major providers: AWS, Azure, GCP
- Basics of cloud security and compliance

Real-World Use Case:

- Choosing the appropriate cloud model for various business applications, Exploring the infrastructure powering common cloud services

Unit Number: 2	**Title: Virtualization, Containers, and Orchestration**	No. of hours: 12

Content:

- Virtualization: Hypervisors (Type 1 & 2), Virtual Machines
- Containerization: Containers vs. VMs, Docker overview
- Docker: Docker Engine, Dockerfile, Images, Containers, Docker Hub
- Container orchestration need and introduction
- Kubernetes basics: Pods, Deployments, Services
- **Real-World Use Case:**
- Running isolated applications via VMs or containers, Packaging an application and its dependencies into a Docker image, Deploying and scaling containerized apps in Kubernetes

Unit Number: 3	**Title: DevOps Principles and Practices**	No. of hours: 10

Content:

- Introduction to DevOps: Culture, CALMS principles

- DevOps lifecycle: Plan, Code, Build, Test, Release, Deploy, Operate, Monitor
- CI/CD: Concepts, CI tools (Jenkins, GitLab CI basics)
- Version control: Git workflows
- Infrastructure as Code (IaC): Introduction to Terraform / CloudFormation

Real-World Use Case:

- Automating build and test processes in software pipelines, Collaboratively managing source code using Git, Defining and deploying infrastructure using code

Unit Number: 4	**Title: Cloud Deployment, Automation, and Monitoring**	No. of hours: 10

Content:

- Deploying applications on cloud using CLI and web consoles
- Automating infrastructure using IaC tools
- Configuration management basics: Ansible / Chef
- Building basic CI/CD pipelines
- Cloud monitoring and logging: CloudWatch, Azure Monitor
- Basics of serverless computing

Real-World Use Case:

- Deploying a web app on a VM or container in the cloud, Automating infrastructure for staging and testing environments, Setting up alerts and logs for monitoring services

Classroom Learning Experience

Inside Classroom Learning:

Concept-Driven Lectures: Use slide decks, whiteboard illustrations, and real-time cloud dashboards to explain:

- Cloud computing fundamentals (IaaS, PaaS, SaaS; deployment models)
- Virtualization and containers (VMs vs. Docker)

Guided Hands-on Labs : Lab activities in supervised sessions:

- Spin up a VM using cloud provider CLI
- Build and push Docker images to Docker Hub
- Use Kubernetes (Minikube) to deploy and scale apps

Case-Based Discussions: Explore real-world scenarios like:

- Choosing the right cloud model for a startup or enterprise
- Automating multi-environment deployments

Outside Classroom Learning Experience

1. **Stage-Wise Project Development –** Implement a complete cloud-native app in phases: containerization, deployment, automation, and monitoring.

2. **Tool-Based Practice –** Use tools like Docker, Jenkins, Terraform, and GitHub for real-world DevOps automation and IaC scripting.

3. **Version Control Collaboration –** Collaborate on GitHub using branches, pull requests, and issue tracking to simulate industry workflows.

4. **Self-Led Learning –** Practice independently using cloud provider free tiers, tutorials, and documentation to reinforce classroom concepts.

5. **Pipeline Testing & Validation –** Apply test-driven practices by integrating unit tests, infrastructure validation, and monitoring into CI/CD pipelines.

Text and Reference Book

1. Hurwitz, J., Bloor, R., Kaufman, M., & Halper, F. – Cloud Computing for Dummies, Wiley
2. Kim, G., Humble, J., Debois, P., & Willis, J. – The DevOps Handbook, IT Revolution Press
3. Hightower, K., Burns, B., & Beda, J. – *Kubernetes: Up and Running*, O'Reilly Media

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Unit 1: Understanding Cloud Foundations & Deployment Models** **Objectives:** • Compare cloud service models (IaaS, PaaS, SaaS) by selecting appropriate services from AWS, Azure, and GCP for given use cases. • Map deployment models (public, private, hybrid) to industry scenarios and explain their implications. • Explore the backend infrastructure (datacenter, storage, networking) of a major cloud provider using service dashboards. • Identify key cloud security standards and compliance frameworks (e.g., GDPR, HIPAA, ISO 27001) and explore where they are applied in the cloud console. **Real-World Scenario:** You are part of an IT consulting team tasked with choosing and justifying a cloud model and provider for a healthcare startup with privacy-sensitive workloads. **Tools:** AWS/Azure/GCP free tier, Whiteboarding, Provider documentation	CO 1
2	**Lab Task 2: Dockerizing and Running Applications in Isolated Environments (Unit 2)** **Objectives:** • Install Docker and create Dockerfiles to containerize a sample web app (e.g., Node.js/Flask). • Push and pull container images from Docker Hub. • Run the app using docker run with environment variables and port mappings.	CO 2

	• Set up a basic multi-container application using Docker Compose. **Real-World Scenario:** You are assigned to onboard a legacy application into containers for better portability and deployment consistency across dev and staging. Tools: Docker, Docker Compose, Docker Hub	
3	**Lab Task 3: Implementing DevOps Pipelines and IaC (Unit 3)** **Objectives:** • Create a Git repository, configure branches, and apply branching strategy (e.g., GitFlow or Feature Branch). • Automate build and test using GitHub Actions or Jenkins pipeline for a sample app. • Write a basic Terraform or CloudFormation script to provision a virtual machine (e.g., EC2) with tags. • Use CI/CD tools to automatically deploy the app after each push. **Real-World Scenario:** As part of a DevOps team, you're responsible for automating the build/test/deploy cycle for a web service while provisioning infrastructure as code. Tools: Git, GitHub Actions, Jenkins, Terraform or AWS CloudFormation	CO 3
4	**Lab Task 4: Application Deployment and Monitoring on the Cloud (Unit 4)** **Objectives:** • Deploy a simple application on AWS EC2 or GCP Compute Engine using CLI. • Implement infrastructure automation using Terraform or Ansible to provision a server and install dependencies.	CO 3

	• Enable logging and monitoring with AWS CloudWatch or Azure Monitor. • Simulate and respond to alerts by modifying thresholds or scaling policies. **Real-World Scenario:** You are part of an SRE team setting up monitoring for a production API. The service must log critical errors and auto-scale if CPU usage exceeds limits. Tools: AWS/GCP CLI, Terraform, Ansible, CloudWatch/Azure Monitor	
5	**Capstone Project: Full Lifecycle Cloud-Based Web Application Delivery** **Objectives:** • Design and document the architecture of a multi-tier web application (frontend, backend, database). • Containerize and deploy the application using Docker and push to a container registry. • Automate the infrastructure provisioning using Terraform or CloudFormation. • Build a CI/CD pipeline using GitHub Actions or Jenkins to deploy the app to cloud (AWS/GCP/Azure). • Set up monitoring and alerting with CloudWatch or Azure Monitor. • Document compliance, backup strategies, and basic cloud security configurations. **Real-World Scenario:** You work for a SaaS startup launching a new service and must design, deploy, automate, and secure the entire app lifecycle on the cloud from scratch. Expected Output: A functional end-to-end web application	CO 4

	deployed with infrastructure as code and automated CI/CD, monitored via dashboards, and running on a cloud provider's infrastructure.	

Object-Oriented Programming with Java

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: Object-Oriented Programming with Java	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCJP404	3-0-2	4	45
Type of Course:	Major			
Pre-requisite(s), if any: Basic Programming concepts				

Course Perspective: This course provides a comprehensive and hands-on introduction to Core Java programming with a focus on real-world applications and industry use cases. The course covers fundamental Java concepts, OOP principles, exception handling, multithreading, file handling, collections framework, and JDBC for database connectivity. The practical focus of the course ensures that students develop industry-relevant Java skills applicable to software development, enterprise applications, cloud-based systems, and microservices. Each unit integrates real-world use cases to help students apply their learning effectively.

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

COs	Statements
CO 1	Developing Java programs using fundamental programming constructs and object-oriented principles.
CO 2	Handling exceptions, multithreading, and concurrency to build efficient applications.
CO 3	Working with file handling and the Java Collections Framework to manage and manipulate data.

CO 4	Implementing database connectivity using JDBC and build simple Java-based data-driven applications.
CO 5	Developing real-world Java applications using best practices and industry standards.

Course Outline:

Unit Number: 1	**Title: Introduction to Java Programming & OOP Principles**	**No. of hours: 10**

Content:

- Java Overview – Features & Architecture
- Java Development Kit (JDK), JVM, JRE
- Data Types, Variables, Operators, Control Flow Statements
- Functions (Methods) – Pass by Value, Recursion
- Object-Oriented Programming (OOP) in Java
- Classes & Objects, Constructors
- Encapsulation, Inheritance, Polymorphism, Abstraction
- Method Overloading & Overriding
- **Real-World Use Cases:**

  ✓ **Bank Account System** – Implement a **class-based banking model** for deposits, withdrawals, and balance inquiries.

  ✓ **E-Commerce Product Catalog** – Define **product classes** with methods for pricing and stock updates.

Unit Number: 2	**Title: Exception Handling, Multithreading, & Concurrency**	No. of hours: 10

Content:

- Exception Handling in Java
- try, catch, finally, throw, throws
- Custom (User-Defined) Exceptions
- Multithreading & Concurrency

- Thread Lifecycle & States
- Creating Threads (Extending Thread class, Implementing Runnable)
- Synchronization, wait() & notify(), Deadlocks
- Executors Framework

Real-World Use Cases:

- ✓ **Stock Market Price Updater** – Use **multithreading** to fetch and display real-time stock prices.
- ✓ **ATM System** – Simulate **concurrent transactions** with synchronization to prevent race conditions.

UnitNumber: 3	**Title: File Handling & Java Collections Framework**	No. of hours: 12

Content:

- File Handling in Java
- Reading & Writing Files (FileReader, FileWriter, BufferedReader)
- Serialization & Deserialization
- Java Collections Framework
- List, Set, Map Interfaces (ArrayList, LinkedList, HashMap, TreeSet)
- Sorting & Searching with Collections
- Iterator & Streams API
- **Real-World Use Cases:**
- ✓ **Log File Analyzer** – Read and analyze a **server log file** to find failed requests.

  ✓ **Contact Management System** – Implement a **contacts database** using **HashMap** with file persistence.

Unit Number: 4	**Title: JDBC & Real-World Java Applications**	No. of hours: 13

Content:

- Java Database Connectivity (JDBC)

- JDBC API Overview, JDBC Drivers
- Connecting Java with MySQL/PostgreSQL
- Executing Queries (Statement, PreparedStatement, CallableStatement)
- Transactions & Batch Processing

Best Practices in Java Development

- Code Optimization Techniques
- Industry-Level Java Coding Standards
- Debugging & Performance Tuning

Real-World Use Cases:

- ✓ Employee Payroll System – Implement a payroll management system with a database backend.
- ✓ Online Library Management – Allow users to borrow and return books with database tracking.

Classroom Learning Experience

Inside Classroom Learning:

1. **Live Coding Sessions:** Demonstrate real-world examples such as:
    a. Creating class-based models (e.g., bank account, e-commerce product)
    b. Implementing custom exceptions
    c. Writing multithreaded programs
2. **Interactive Code Walkthroughs:** Analyze and debug sample programs line-by-line.
3. **Guided Labs & Hands-On Exercises:** Step-by-step lab exercises during contact hours to build:
    a. Thread-safe applications
    b. File readers and writers
    c. JDBC-based data CRUD operations

4. **Case-Based Learning & Group Discussions:** Discuss industry use cases (e.g., payroll systems, stock updaters).

Outside Classroom Learning Experience

1. **Mini Projects & Use Case Implementation –** Build apps like contact managers, payroll systems, or online libraries based on real-world problems.
2. **Tool Practice –** Practice using IDEs (Eclipse, IntelliJ), version control (Git), and MySQL/PostgreSQL for database integration.
3. **Peer Code Reviews –** Use GitHub to review classmates' code and give feedback based on Java coding standards.
4. **Self-Led Debugging & Testing** – Write test cases and debug Java apps using logging and IDE-based breakpoints.
5. **Applied Learning Tasks –** Complete assignments like analyzing logs, creating multi-threaded apps, or optimizing file I/O performance.

Text and Reference Book

1. **Schildt, H. (2018).** *Java: The Complete Reference* (11th ed.). McGraw Hill.
2. **Deitel, P. J., & Deitel, H. M. (2017).** *Java: How to Program* (10th ed.). Pearson.
3. Kim, G., Humble, J., Debois, P., & Willis, J. – The DevOps Handbook, IT Revolution Press

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab Task 1: Java OOP Fundamentals through Banking & Product Catalog Systems** **Objectives:**	CO 1

	• Implement a BankAccount class with methods for deposit(), withdraw(), and checkBalance(). • Use constructors, method overloading, and encapsulation for a clean OOP design. • Create a Product class with pricing logic and stock update features using inheritance and polymorphism. • Demonstrate pass-by-value and recursion with auxiliary methods (e.g., factorial, interest calculation). **Real-World Scenario:** Model a basic digital banking system and e-commerce inventory logic using foundational OOP principles. Tools: Java (JDK 17+), IntelliJ IDEA / Eclipse	
2	**Lab Task 2: Exception Handling & Concurrent ATM Simulation** **Objectives:** • Build an ATM simulation with threads handling multiple users accessing a shared BankAccount object. • Apply synchronization to avoid race conditions in withdraw() and deposit(). • Create and use custom exceptions for invalid PIN, insufficient balance, etc. • Demonstrate the use of Executors and Thread.sleep() to simulate realistic delays. Real-World Scenario: Simulate multiple ATMs operating concurrently on shared accounts, while managing exception safety and concurrency control. **Tools:** Java, IntelliJ IDEA / Eclipse	CO 2
3	**Lab Task 3: Contact Manager with File I/O and Collections** **Objectives:**	CO 3

	• Create a ContactManager class using HashMap to store and manage contacts. • Implement serialization to persist contact data to a file (e.g., .ser or .txt format). • Enable search, delete, and update operations on contacts using Map and List interfaces. • Use Iterator and Stream API to sort/filter contacts (e.g., by name or number prefix). **Real-World Scenario:** Manage a persistent contact directory system with advanced filtering and file-based storage.	
4	**Lab Task 4: JDBC Integration with Payroll Management System** **Objectives:** • Design a relational schema for employees and payroll records in MySQL/PostgreSQL. • Use JDBC to connect Java application with database and perform CRUD operations. • Use PreparedStatement for secure input and batch processing for bulk payroll entry. • Manage transactions (commit, rollback) and apply performance tuning techniques. Real-World Scenario: Automate payroll processing for an organization, allowing admins to add, update, and retrieve payment records securely. **Tools:** Java, MySQL/PostgreSQL, JDBC Driver, DBeaver	CO 3
5	• **Capstone Project: Online Library Management System with Full Java Stack** **Objectives:**	CO 4

	• Design and develop a fully functional library management system supporting multiple users. • Implement classes for Book, User, and Transaction with OOP best practices. • Persist book and user data in MySQL using JDBC; include borrowing, return, and fine calculation features. • Use multithreading to simulate multiple concurrent users borrowing books. • Implement exception handling, file logging (e.g., for overdue fines), and a simple admin CLI interface. **Real-World Scenario:** Deploy a complete library system where multiple users can interact with a database-backed catalog in a concurrent, persistent, and user-friendly environment.	

Open Elective-II

Course 1: Creative Coding & Generative Art

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Creative Coding & Generative Art**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	OEC	3-0-2	3	45
Type of Course:	OEC			

Course Perspective: This course introduces students to the expressive and artistic side of computing through creative coding. Students will explore algorithms and code as mediums for generative visuals, sound, and interactive art using beginner-friendly tools such as p5.js. Emphasis is placed on creativity, design, iteration, and experimentation.

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

COs	Statements
CO 1	Understanding the fundamentals of creative coding and generative art.
CO 2	Using JavaScript and p5.js to create interactive and visually dynamic programs.
CO 3	Applying design and computational thinking to produce creative works.
CO 4	Exploring randomness, noise, and algorithms in visual composition.
CO 5	Developing digital art projects that express personal or cultural themes

Course Outline:

Unit Number: 1	**Title: Introduction to Creative Coding**	**No. of hours: 10**
Content: • What is creative coding? History and philosophy • Generative art and algorithmic aesthetics • Introduction to p5.js and JavaScript basics • Drawing shapes, colors, and animation in code		
Unit Number: 2	**Title: Motion and Interaction**	**No. of hours: 10**
Content: • Coordinates, motion, and object behaviors • Responding to user input (mouse, keyboard) • Building interactive sketches • Real-time graphics and performance		
Unit Number: 3	**Title: Generative Design Techniques**	No. of hours: 13
• Randomness, Perlin noise, and pattern generation • Recursion, symmetry, and fractals • Text and typography in generative art • Sound, visuals, and synesthesia		
Unit Number: 4	**Title: Final Project and Art for Social Good**	No. of hours: 12
Content: • Designing and iterating generative art projects • Documenting code and artistic intent		

- Ethics and accessibility in creative technology
- Final exhibition or online portfolio

Textbook: McCarthy, L., & Reas, C. (2015). Getting Started with p5.js: Making Interactive Graphics in JavaScript and Processing. Maker Media.

Additional References: Shiffman, D. (2012). The Nature of Code. http://natureofcode.com

Course 2: Human-Computer Interaction (HCI) & UX Principles

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Human-Computer Interaction (HCI) & UX Principles**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	OEC	3-0-2	3	45
Type of Course:	OEC			

Course Perspective: This course explores the design and evaluation of user-centered digital systems. Topics include HCI principles, usability heuristics, interaction design, and user experience (UX) methods. Students will learn prototyping, wireframing, and user research skills using industry tools like Figma.

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

COs	Statements
CO 1	Understanding HCI concepts and the human-centered design process.
CO 2	Conducting user research and usability evaluation.
CO 3	Creating low- and high-fidelity prototypes using UX tools.
CO 4	Applying design principles for usability, accessibility, and aesthetics.
CO 5	Communicating design decisions through documentation and user feedback.

Course Outline:

<table>
<tr><td>Unit Number: 1</td><td>Title: Introduction to HCI and UX</td><td>No. of hours: 10</td></tr>
<tr><td colspan="3">Content:

• HCI history, human factors, and cognitive psychology
• UX vs UI, usability heuristics (Nielsen's principles)
• Affordances, feedback, and user flows</td></tr>
<tr><td>Unit Number: 2</td><td>Title: User Research and Interaction Design</td><td>No. of hours: 10</td></tr>
<tr><td colspan="3">Content:

• Conducting interviews, observations, surveys
• Personas, scenarios, and storyboards
• Paper prototyping and design sketching</td></tr>
<tr><td>Unit Number: 3</td><td>Title: Digital Prototyping and Evaluation</td><td>No. of hours: 13</td></tr>
<tr><td colspan="3">• Wireframing and clickable prototypes (e.g. Figma)
• Usability testing and heuristic evaluation
• A/B testing and analytics</td></tr>
<tr><td>Unit Number: 4</td><td>Title: Accessibility, Ethics, and Final Project</td><td>No. of hours: 12</td></tr>
<tr><td colspan="3">Content:

• Designing and iterating generative art projects
• Documenting code and artistic intent
• Ethics and accessibility in creative technology
• Final exhibition or online portfolio</td></tr>
</table>

Textbook: Rogers, Y., Sharp, H., & Preece, J. (2011). *Interaction Design: Beyond Human-Computer Interaction*. Wiley.

Additional References: Norman, D. (2013). *The Design of Everyday Things*. Basic Books.

Course 3: Digital Storytelling & Communication Tools

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Digital Storytelling & Communication Tools**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	OEC	3-0-2	3	45
Type of Course:	OEC			

Course Perspective: This course teaches students how to craft and share powerful stories using digital media. Students explore narrative techniques, visual communication, and multimedia production with tools such as Canva, Adobe Spark, and podcasting platforms.

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

COs	Statements
CO 1	Understanding elements of narrative structure and storytelling frameworks.
CO 2	Using digital tools to design visual and multimedia stories.
CO 3	Creating impactful presentations, videos, or podcasts.
CO 4	Developing storytelling strategies for personal branding or advocacy.
CO 5	Critically analyzing digital media narratives and design for engagement.

Course Outline:

Unit Number: 1	Title: Storytelling Foundations	No. of hours: 10
Content: • Elements of story: plot, character, setting • Story arcs and digital adaptation • Branding through narrative		
Unit Number: 2	**Title: Visual and Multimedia Communication**	**No. of hours: 11**
Content: • Design principles for visual storytelling • Creating infographics and social posts (e.g. Canva) • Video editing basics and animation (e.g. Adobe Spark)		
Unit Number: 3	**Title: Audio and Podcasting**	No. of hours: 12
• Podcast planning, scripting, and recording • Tools for editing and publishing • Storyboarding for multimedia stories		
Unit Number: 4	**Title: Publishing and Impact**	No. of hours: 12
Content: • Digital storytelling for social impact • Content strategy for platforms (YouTube, Instagram, etc.) • Final project: publish a story-driven digital piece		

Textbook: Alexander, B. (2011). *The New Digital Storytelling: Creating Narratives with New Media*. Praeger.

Additional References: Lambert, J. (2013). *Digital Storytelling: Capturing Lives, Creating Community*. Routledge.

Course 4: Sustainable Innovation & Tech for Good

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Sustainable Innovation & Tech for Good**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	OEC	3-0-2	3	45
Type of Course:	OEC			

Course Perspective: This course explores how technology can be used as a force for sustainable development and social impact. Students learn about the principles of sustainable innovation, analyze real-world case studies, and work on designing tech-enabled solutions for societal challenges. The course emphasizes systems thinking, ethical technology, and global goals such as the UN SDGs.

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

COs	Statements
CO 1	Understanding the principles of sustainable development and responsible innovation.
CO 2	Analyzing technological solutions in the context of environmental and social impact.
CO 3	Applying design thinking to frame and address sustainability challenges.
CO 4	Exploring the intersection of emerging technologies and social innovation.
CO 5	Proposing a tech-for-good solution aligned with real-world problems.

Course Outline:

Unit Number: 1	**Title: Introduction to Sustainable Innovation**	**No. of hours: 10**
Content: • Sustainability principles and UN Sustainable Development Goals (SDGs) • Systems thinking and life cycle perspective • Tech for social impact: definitions and frameworks • Case studies from global innovation hubs		
Unit Number: 2	**Title: Ethical and Responsible Technology**	**No. of hours: 10**
Content: • Environmental footprint of digital technologies • Equity and access in innovation • Bias, surveillance, and data privacy • Inclusive innovation and co-design methods		
Unit Number: 3	**Title: Emerging Technologies for Good**	No. of hours: 13
Content: • Green computing and IoT for environment monitoring • AI for climate resilience, health equity, education access • Blockchain for transparency in social services • Drones, AR/VR, and low-cost robotics for humanitarian work		
Unit Number: 4	**Title: Design Thinking for Social Impact**	No. of hours: 12
Content: • Problem framing and stakeholder empathy		

- Ideation and prototyping sustainable solutions
- Theory of change and social business models
- Storytelling and pitching tech-for-good projects

Textbook:

Bocken, N., Short, S., Rana, P., & Evans, S. (2014). A Literature and Practice Review to Develop Sustainable Business Model Archetypes. Journal of Cleaner Production, 65, 42–56.

Additional References:

Smith, A., Fressoli, M., & Thomas, H. (2014). Grassroots Innovation Movements: Challenges and Contributions. Journal of Cleaner Production, 63, 114–124.

Course 5: Entrepreneurship & MVP Prototyping with No-Code Tools

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Entrepreneurship & MVP Prototyping with No-Code Tools**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	OEC	3-0-2	3	45
Type of Course:	OEC			

Course Perspective: This hands-on course equips students with the ability to take ideas from concept to prototype using no-code development tools. It covers entrepreneurial fundamentals, MVP design, and rapid prototyping to empower aspiring founders to build and test digital products without deep coding expertise.

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

COs	Statements
CO 1	Applying the principles of startup ideation and lean methodology.
CO 2	Using no-code tools to build working prototypes for digital products.
CO 3	Designing and iterating MVPs based on user feedback.
CO 4	Communicating and pitching startup concepts with clarity and confidence.
CO 5	Evaluating business model viability and product-market fit using real-world tools.

Course Outline:

Unit Number: 1	**Title: Startup Ideation & Validation**	**No. of hours: 10**
Content: • Problem identification and value proposition canvas • Target audience, early adopters, and customer interviews • Market analysis and competitor benchmarking • Lean startup and build-measure-learn feedback loop		
Unit Number: 2	**Title: MVP Design and Prototyping Tools**	**No. of hours: 10**
Content: • Overview of no-code platforms (Bubble, Glide, Adalo, Webflow) • Designing user flows and wireframes • Building MVPs with drag-and-drop tools • Testing MVPs with real users and collecting feedback		
Unit Number: 3	**Title: Product-Market Fit and Business Modeling**	No. of hours: 13
• Minimum viable metrics and KPIs • Revenue models and monetization strategies • Business model canvas and lean canvas • Legal, payment, and integration basics for no-code founders		
Unit Number: 4	**Title: Pitching and Scaling**	No. of hours: 12
Content: • Storytelling and elevator pitch techniques • Creating compelling pitch decks and demos		

- • Growth strategies for MVPs
- • Final demo day with peer and mentor feedback

Textbook:

Ries, E. (2011). The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Business.

Additional References:

Koenig, N., & Beaver, J. (2020). The No-Code Startup: Launch a Startup Without Writing Code. Indie Publishing.

COMPETITIVE CODING -II

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Competitive Coding -II**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		2-0-0	2	30
Type of Course:	SEC			
Pre-requisite(s), if any: Competitive Coding-I, Fundamentals of programming & data structure				

Course Perspective: This course enhance students' problem-solving abilities in competitive coding by providing in-depth knowledge of core data structures, algorithms, and efficient coding techniques. This course aims to prepare students for technical assessments and coding interviews, building a strong foundation for tackling real-world coding challenges.

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

COs	Statements
CO 1	Apply advanced string algorithms to solve complex problems.
CO 2	Analyze and implement efficient linked list operations and complex problem solutions.
CO 3	Evaluate and apply various tree traversal techniques to solve traversal and view-related problems.

Course Outline:

Session:1	Advance Array-I	No. of hours: 2
Content summary: Two sum, Best time to buy and sell stocks, Sort 0, 1 and 2(Dutch flag algorithm)		
Session:2	Advance Array-II	No. of hours: 2
Content Summary: container with most water, merge sorted array, trapping rain water		
Session:3	Binary Search-I	No. of hours: 2
Content Summary: lower bound , upper bound, koko eating bananas, first bad version		
Session: 4	Binary Search-II	No. of hours: 2
Content Summary: Search in rotated sorted array, Search in rotated sorted array II, aggressive cows		
Session: 5	Binary Tree Introduction	No. of hours: 2
Content Summary: Introduction of Tree, type of tree, implementation of tree.		
Session: 6	Binary Tree Traversal	No. of hours: 2
Content Summary: Tree Traversal, preorder traversal, inorder traversal, postorder traversal, level order traversal(Morris traversal).		
Session: 7	Binary Tree-III.	No. of hours: 2
Content Summary: Height of the tree, same tree, symmetric tree,		
Session: 8	Binary Tree-IV.	No. of hours: 2
Content Summary: diameter of tree, path sum, print left/right view of Binary tree.		

Session : 9	Binary Search Tree.	No. of hours: 2
Content Summary: Implementation of BST, check valid BST		
Session : 10	Binary Search-II	No. of hours: 2
Content Summary: convert sorted array to BST, Delete node in BST, lowest common ancestor		
Session : 11	Hashmap Introduction.	No. of hours: 2
Content Summary: HashMap Implementation (operations put, get, containsKey, KeySet)		
Session: 12	HashMap-II.	No. of hours: 2
Content Summary: Two Sum, highest frequency character, missing number		
Session:13	HashMap-III.	
Content Summary: intersection of two arrays, set matrix zeros, valid anagram		
Session: 14	hashmap/Sliding window-technique Algorithm	No. of hours:2
Content Summary: longest consecutive sequence, longest substring without repeating character, bulls and cows		
Session: 15	hashmap/Sliding window-technique Algorithm	No. of hours: 2
Content Summary: largest subarray with 0 sum, count of zero sum subarray, length of largest subarray with contiguous element		
Session: 16	Priority Queue	No. of hours: 2
Content Summary: Implementation of Priority queue, min and max Heap		
Session: 17	priority Queue-II	No. of hours: 2
Content Summary: Inplace heap sort, kth largest element, kth smallest element		
Session:	priority Queue-III	No. of hours: 2

18		
Content Summary: check max heap, top k frequent element, sliding window maximum		
Session: 19	Sum up Binary tree and Binary search Tree	No. of hours: 2
Content Summary: sum of leaves, top view, bottom view,		
Session: 20	Sum up Hashmap / Sliding window technique.	No. of hours: 2
Content Summary: find all anagram in string, isomorphic string		

Reference Books:

- "Introduction to Algorithms" by Cormen, Leiserson, Rivest, and Stein
- "Cracking the Coding Interview" by Gayle Laakmann McDowell
- "Elements of Programming Interviews" by Adnan Aziz, Tsung-Hsien Lee, and Amit Prakash

Communication & Personality Development

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: Communication & Personality Development	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		2-0-0	**2**	**30**
Type of Course:	**AEC**			
Pre-requisite(s):				

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	Improve public speaking and presentation abilities to confidently convey ideas and information.
CO 2	Understand the framework of Communication to augment oratory skills and written English
CO 3	Cultivate essential soft skills required at the different workplaces.

Course Outline:

Unit Number: 1	Title: Developing self and others	No. of hours: 10
Content:		

<table>
<tr><td colspan="3">Self Awareness, Personality Concepts (Personality Assessments -Johari Window, Myers & Brigg), Self-Management, Self Esteem, Self-Efficacy, Interpersonal skills, mindset, grit and working in teams.</td></tr>
<tr><td>Unit Number: 2</td><td>Title: Enhancing Reading and Writing Skills</td><td>No. of hours: 8</td></tr>
<tr><td colspan="3">Content:
Speed reading and its importance in competitive examinations, techniques for speed reading, note-taking, and critical analysis. Paragraph Writing, Essay and Summary writing, Business Letter, Email writing</td></tr>
<tr><td>Unit Number: 3</td><td>Title: Effective Communication and Public Speaking</td><td>No. of hours: 12</td></tr>
<tr><td colspan="3">Content:
Communication Framework, barriers & overcoming these barriers, Group Discussions, Extempore & Public Speaking drills, to manage stage fright and anxiety. Structuring and organizing a presentation (Oral & PPT), Etiquettes, Grooming, Body Language and Conversation starters, TMAY.</td></tr>
<tr><td>Unit Number: 4</td><td>Title: Career Guide and readiness</td><td>No. of hours: 6</td></tr>
<tr><td colspan="3">Content:
Cover Letter, ATS friendly resume, Elevator Pitch, Video Resume (Visume), Networking, Group Discussion, Mock Interviews. Capstone Project</td></tr>
</table>

Learning Experiences

1. Impromptu Speaking & Pitch Practice : Real-time speaking drills such as elevator pitches, TMAY ("Tell Me About Yourself") sessions, and ideation for tech projects.
2. Mock Interviews & Group Discussions (GD): Simulated tech-based GDs and interviews with peer feedback and rubrics to build confidence and spontaneity.
3. Personality Assessments: MBTI and Johari Window-based self-analysis followed by reflective writing on strengths and areas of improvement for tech careers.

4. Presentation & Demo Etiquette: Preparing project pitch decks, oral walkthroughs of ML/AI workflows, and technical presentations with body language cues and audience engagement.

Outside Classroom Learning

1. **Tech Blog Writing & LinkedIn Posts**
    - Regular writing assignments on AI trends, personal projects, or hackathon experiences to develop thought leadership and online presence.
2. **Video Resume (Visume) Practice**
    - Record and critique short videos pitching AI/ML skills, final-year projects, or explaining complex tech topics in layman's terms.
3. **Professional Networking Activities**
    - Connect with tech professionals, attend virtual AI/ML events, and maintain a log of key takeaways and connections made.
4. **Portfolio Development**
    - Maintain a GitHub repository with project documentation, README files, and contribution **logs; practice explaining work to non-technical audiences.**

Textbooks:

1. Textbooks/Web resources/MOOCs/Magazines/Journals/Videos/Podcast etc.
2. https://www.indiabix.com/online-test/aptitude-test/
3. https://www.geeksforgeeks.org/aptitude-questions-and-answers/
4. https://www.hitbullseye.com/

MINOR PROJECT-II

Program Name:	**B.Tech (CSE) with specialization in UI/UX**		
Course Name: Minor Project-II	**Course Code**	**L-T-P**	**Credits**
	ETCCPR405	0-0-4	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.

Semester: 5

Program Name	B.Tech CSE (Specialization in UI & UX)			
Course Name: **Operating Systems**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCOS502	**3-0-2**	**4**	**45**
Type of Course:	**Major**			
Pre-requisite(s): To study Operating Systems, one should know computer architecture, data structures, and programming (preferably in C/C++). Basic understanding of memory, processes, algorithms, and digital logic is also essential.				

Course Perspective: This course treats the operating system as the practical toolbox every software engineer must master. It bridges hardware and code, reveals where performance is won or lost, and underpins many interview questions. By blending concise theory with hands-on command-line work, students gain the insight and habits needed to build, tune, and troubleshoot real systems—skills that translate directly to success in internships, campus placements, and day-to-day engineering tasks.

The Course Outcomes (COs). Upon successful completion of this course, students will be able to:

COs	Statements
CO 1	Understanding core OS abstractions—processes, threads, memory, files.
CO 2	Applying scheduling and synchronisation for efficient, deadlock-free execution
CO 3	Analysing and optimising memory and virtualisation performance.
CO 4	Building fault-tolerant storage and file-system components
CO 5	Troubleshoot and defend OS design choices in interview scenarios.

Course Outline:

Unit Number: 1	**Foundations, Process Model, & Command-Line Essentials**	**No. of hours: 11**
• Why operating systems exist: resource abstraction, protection, concurrency. • Kernel architectures (monolithic, modular, micro-kernel, exo-kernel) and the boot sequence from firmware to first user process. • Privilege rings, traps, system-call flow, context switch anatomy. • Process lifecycle: fork–exec, signals, zombie reaping, POSIX vs native threads, introduction to containers and namespaces. • Command-line mastery: ps, top/htop, nice, kill, strace, lsof, grep, awk, redirection & pipelines; shell job-control and basic scripting. • Essential administration: user & group management (useradd, passwd, sudoers), file permissions and ACLs, systemd service inspection, log discovery under /var/log.		
Unit Number: 2	**Title: Concurrency, Scheduling & Synchronization**	**No. of hours: 10**
• Pre-emptive scheduling vs cooperative yielding; context-switch overhead and CPU affinity. • Classic and modern schedulers: multi-level feedback queue, Completely Fair Scheduler, real-time classes. • Synchronization primitives: mutex, spinlock, semaphore, barrier, RCU; atomic instructions and memory-ordering gotchas. • Deadlock detection, avoidance and recovery; canonical problems (dining philosophers, readers–writers). • System-level profiling: perf sched, sar, vmstat, flame graphs.		
Unit Number: 3	**Title: Memory Management, Virtualization & Advanced Administration**	**No. of hours: 12**

• Virtual memory mechanics: paging, TLBs, copy-on-write, page faults; page-replacement algorithms (LRU, CLOCK, working set). • Kernel allocators (buddy, slab, SLUB), mmap versus brk, NUMA awareness. • Hardware-assisted virtualisation (VT-x, AMD-V), paravirtual machines, containers (cgroups, namespaces), sandboxing with seccomp-bpf. • Memory-related admin: monitoring with free, smem, /proc/meminfo; tuning swappiness, huge pages; diagnosing leaks with valgrind and massif.		
Unit Number: 4	**Storage, File Systems, Reliability, Networking & Troubleshooting**	**No. of hours: 12**
• Device basics: block vs character, NVMe queueing, DMA, interrupt-driven I/O. • Disk scheduling (C-LOOK, CFQ, deadline), SSD firmware concepts (TRIM, wear-levelling). • File-system internals: inodes, journaling, copy-on-write, snapshots; RAID levels and checksum-based integrity. • Consistency and recovery: write-ahead logging, fsck, distributed storage hints (eventual vs strong consistency). • Security hooks: DAC, MAC (SELinux/AppArmor), Linux capabilities, container breakout mitigation.		

Learning Experience:

Classroom Learning

- Interactive Lectures: Use visual diagrams, animations, and analogies to explain core concepts:
- Concept Clarity: Step-by-step breakdowns of complex topics
- Live Demos & Simulations
- Group Work & Case Studies
- Problem-Solving Practice:

Outside Classroom Learning

1. **Hands-On Projects**

   Build:

   - A basic shell interpreter
   - Simulated CPU scheduler or memory manager

- File system navigator in C/C++

2. **Coding Assignments**
   - Implement:
     - Multithreaded programs using POSIX threads.
     - Paging, segmentation, or LRU algorithms in C/C++.
     - Monitor system performance using tools like top, vmstat, htop.

3. **Collaborative Assignments**

   Team activities such as:

   - OS-level debugging for user applications.
   - Writing a proposal for a lightweight kernel module.
   - Designing a custom command interpreter or file system explorer.

Textbooks:

1. Arpaci-Dusseau, R. H., & Arpaci-Dusseau, A. C. (2018). *Operating Systems: Three Easy Pieces (OSTEP)*.
2. Silberschatz, A., Galvin, P. B., & Gagne, G. (2020). *Operating System Concepts* (10th ed.). Wiley.

Reference Books:

• William Stallings, Operating Systems: Internals and Design Principles, 8th Edition, Pearson, 2014.

• D.M. Dhamdhere, Operating Systems: A Concept-Based Approach, 2nd Edition, McGraw Hill, 2006.

• Gary Nutt, Operating Systems: A Modern Perspective, 2nd Edition, Addison-Wesley, 2004.

Additional Readings:

- NPTEL Course Link:

  Operating System by Prof. P.K. Biswas (IIT Kharagpur)
- Linux Journey (Basics of Linux OS): https://linuxjourney.com
- xv6 Operating System (MIT Educational OS): https://pdos.csail.mit.edu/6.828/2020/xv6.html
- GeeksforGeeks OS Tutorials: https://www.geeksforgeeks.org/operating-systems/

Lab Experiments

Ex. No	Experiment Title Each mini project is designed with 3–4 sub-objectives and focuses on real-world use cases:	Mapped CO/COs
1	Experiment 1 — "Hello-Shell & Syscall Peek" (Unit 1: Kernel basics, processes, command-line) Main problem Write a tiny interactive shell, then watch it cross the user-kernel boundary. Sub-problems 1. Parse simple commands, I/O redirection, cd, exit, and background "&". 2. Add an internal pid command that lists child PIDs created so far. 3. Run strace -c ./hello-sh while executing one external command; copy the top five system calls and write a one-line purpose for each.	CO 1
2	**Experiment 2 — "CPU-Scheduling Explorer" *(Unit 2: Concurrency, scheduling, synchronisation)*** Main problem Understand how different schedulers share the CPU, first in a simulator, then on a live Linux box. Sub-problems 1. Write a command-line simulator that reads a file of ⟨arrival, burst⟩ pairs and computes average turnaround and waiting time for FCFS, SJF (non-pre-emptive), Round-Robin (2 ms slice) and static-priority. 2. Generate a simple ASCII Gantt chart for each policy. 3. Compile two CPU-bound loops (spinA, spinB). Launch them at default priority, measure per-process CPU time with pidstat or top. 4. Re-run with nice -n 10 spinB; note the change. 5. Set spinB to real-time SCHED_RR with chrt -r -p 80 <pid> and record context-switch counts using perf stat -e context-switches. Explain each difference in a short note.	CO 2

3	**Experiment 3 — "Page-Replacement Toy"** ***(Unit 3: Memory management & virtualisation)*** Main problem Simulate virtual-memory pressure and compare replacement algorithms. Sub-problems 1.Read a trace of page numbers and simulate LRU and CLOCK for a user-defined frame count. 2.Print total page faults for each algorithm. 3. Repeat for frame counts 4, 8, 16 and 32; plot the fault curve or create a small table and explain the trend.	CO 3
4	**Experiment 4 — "Mini Log-Safe File Store"** ***(Unit 4: Storage, file systems, reliability)*** Main problem Build a simple key-value store that survives crashes. Sub-problems 1.Implement PUT key value by appending to a text log; GET key returns the latest value. 2. Kill the program mid-run and restart; verify data is intact. 3. Benchmark 1 000 puts with immediate fsync() and again batching fsync() every 100 writes; record the speed-up and discuss durability trade-off.	CO 3
5	**Experiment 5 — "Capstone: Tiny Guestbook Service"** ***(Units 1-4 combined)*** Main problem Combine your earlier work into a small but complete network service and tune it end-to-end. Sub-problems 1. Reuse the thread-pool server pattern; each POST (name,msg) stores an entry via the log-safe file store, then returns the last 10 messages. 2. Replace default malloc with a small fixed-size pool (adapt ideas from Experiment 3). 3. Create a systemd service file, add a dedicated user, and set restrictive file permissions.	CO 4

	4. Run a 60-second load test with two clients; capture CPU, memory, context-switch and disk-I/O stats (pidstat, free, iostat). 5. Produce a one-page report showing before/after metrics and explaining how process design, scheduling choices, memory management, and logging strategy interact to meet the workload.	

Arithmetic and Reasoning

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Arithmetic and Reasoning**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		2-0-2	**2**	**30**
Type of Course:	**AEC**			
Pre-requisite(s): Basic understanding of numbers, operations, logic patterns, and problem-solving skills.				

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 academic and various practical fields.

The Course Outcomes (COs). Upon successful completion of this course, students will be able to:

COs	**Statements**
CO 1	Understanding arithmetic algorithms required for solving mathematical problems.
CO 2	Applying arithmetic algorithms to improve proficiency in calculations**.**
CO 3	Analyzing cases, scenarios, contexts and variables, and understanding their inter-connections in each problem.
CO 4	Evaluating & deciding approaches and algorithms to solve mathematical & reasoning problems.

Course Outline:

Unit Number: 1	**Title: Mathematical Essentials**	**No. of hours: 10**

Vedic Maths, Classification of Numbers and Divisibility Rule, Percentage, Ratio and Proportion		
Unit Number: 2	**Title: Fundamentals of Logical Reasoning**	**No. of hours: 05**
Blood Relations, Direction Sense, Coding Decoding		
Unit Number: 3	**Title: Elementary Quantitative Skills**	**No. of hours: 10**
Simple and Compound Interest, Average, Partnership, Time and Work, Time Speed & Distance		
Unit Number: 4	**Advanced Quantitative Skills**	**No. of hours: 05**
Permutation & Combination, Probability		

Learning Experience:

Inside Classroom Learning

- **Interactive Lectures:** Visual tools, real-life examples (bills, clocks, puzzles) to explain core topics like percentages, averages, ratios, coding-decoding, and syllogisms.
- **Concept Clarity:** Step-by-step methods, mental math tricks, logical tables/flowcharts, and elimination techniques for MCQs.
- **Simulations & Games:** Use of Kahoot, Quizizz, and puzzles like Sudoku for engaging practice.
- **Collaborative Learning:** Group activities and case-based tasks for word problems and logical reasoning.
- **Focused Practice:** Regular drills on quantitative and analytical topics with previous exam questions.

Outside Classroom Learning

- **Hands-On Projects:** Build creative prototypes like a Math Maze game, budget tracker, and logic-based quiz app to apply math concepts.
- **Optional Coding Tasks:** For advanced learners—write programs in C/Python for solving number patterns and math logic problems.

- **Collaborative Assignments:** Team activities include designing aptitude tests, creating explainer videos, and developing logic-based board games.
- **Self-Learning Support:** Access curated books, educational apps, websites, and YouTube channels for independent practice and concept reinforcement.

Textbooks/Web resources / MOOCs / Magazines/ Journals/ Videos /Podcast etc.

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

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

https://www.hitbullseye.com/

Specialization Course-III

Visual & Interface Design Principles

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Visual & Interface Design Principles**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		3-0-2	4	45
Type of Course:	**ETUXDP503**			
Pre-requisite(s), if any: Understanding of Design Basics, Graphic Design Tools such as Adobe Photoshop, Illustrator, Figma, Sketch, or Adobe XD.				

Course Perspective: This course introduces students to the foundational principles of visual and interface design as applied in digital product development. Through a hands-on approach, students will learn to design aesthetically pleasing, usable, and consistent user interfaces using color theory, typography, layout grids, hierarchy, and iconography. Students will practice creating responsive and accessible design components using modern tools like Figma and Adobe XD. The course places strong emphasis on consistency, feedback, affordances, and the psychology behind good design. By the end of the course, students will be capable of designing high-quality UI screens aligned with user experience principles and business goals.

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

COs	Statements
CO 1	Explaining the fundamental visual design principles including layout, hierarchy, balance, and contrast.
CO 2	Applying principles of typography, color, iconography, and branding elements to design aesthetically pleasing interfaces.
CO 3	Creating responsive layouts and screen designs using grids, spacing systems, and component-based design.
CO 4	Evaluating visual designs based on usability, accessibility, and cognitive load.

CO 5	Designing complete interface screens using modern tools and industry guidelines.

Course Outline:

Unit Number: 1	**Title: Foundations of Visual Design**	**No. of hours: 12**

Objective: Understand basic principles of visual communication and aesthetics.
Contents:

- Introduction to visual design: Role in UX and UI
- Gestalt principles: Proximity, similarity, closure, figure-ground
- Visual hierarchy and layout balance
- Concepts of contrast, alignment, white space, and visual weight
- Moodboards, style tiles, and inspiration boards

Hands-On:

- Create a visual inspiration board for a brand using Figma
- Break down UI designs using Gestalt principles

Unit Number: 2	**Title: Typography & Color Theory in Interface Design**	**No. of hours: 12**

Objective: Apply the use of type and color for communication and accessibility.
Contents:

- Typography basics: Typeface vs font, readability vs legibility
- Choosing fonts for screens: Hierarchy, scale, alignment
- Google Fonts and licensing practices
- Color theory: Primary, secondary, tertiary, complementary schemes
- Creating a color palette and UI color tokens
- Contrast & accessibility (WCAG 2.1 AA)

Hands-On:

- Build a type scale for a mobile/web app

- Create a UI color palette and accessibility test in Stark/Figma plugins

Unit Number: 3	**Title: Layout Grids, Spacing, Icons & Component Design**	**No. of hours: 10**

Objective: Design usable, consistent and responsive interfaces.
Contents:

- Grid systems: 8pt grid, column layout, margin and gutter design
- Spacing systems and modular scales
- Components: Buttons, input fields, form patterns, cards
- Icons: Visual weight, sizing, grid alignment
- Design consistency and design systems (Material, Carbon, Fluent UI)
- Responsive layout principles (Mobile-first, breakpoints)

Hands-On:

- Create a card component using an 8pt grid
- Design a mobile & desktop version of a simple landing page using grid layout

Unit Number: 4	**Title: Feedback, Affordance, Accessibility & Evaluation**	**No. of hours: 11**

Objective: Make interfaces more usable and inclusive through feedback and design heuristics.
Contents:

- Types of UI feedback: Microinteractions, animations, loading states, confirmations
- Affordances and signifiers: Making functions obvious
- UI states: Default, hover, disabled, error, active
- Evaluating interfaces using heuristics (Nielsen's principles)
- UI accessibility considerations: Focus order, keyboard navigation
- Common visual design anti-patterns

Hands-On:

- Create state variants for buttons or forms
- Conduct heuristic evaluation of a public website and suggest UI design improvements

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	Lab Assignment 1 (Unit 1) Title: Applying Gestalt Principles in a UI Redesign Sub-Objectives: 1. Analyze a poorly designed screen (real or provided) for violations of visual hierarchy 2. Re-apply Gestalt principles like proximity and similarity for better grouping 3. Design a corrected layout using white space and contrast 4. Present rationale for visual choices using design vocabulary 5. Submit before/after screen comparison using Figma	CO 1
2	Lab Assignment 2 (Unit 2) Title: Building a Visual Identity Using Typography and Color Sub-Objectives: 1. Choose or define a brand type personality (e.g., youth-centric, formal) 2. Select appropriate fonts and build a scalable typography system 3. Create a color palette with primary, secondary, and accent colors	CO 2

	4. Use Stark plugin to test color accessibility for buttons/text 5. Apply type and color system to a sample login/signup page	
3	Lab Assignment 3 (Unit 3) Title: Designing a Responsive Pricing Section Using Layout Grids Sub-Objectives: 1. Define a 12-column grid for desktop and 4-column for mobile 2. Create 3 pricing cards using modular spacing and consistent visual patterns 3. Ensure visual alignment, spacing consistency, and proper icon sizing 4. Apply responsive layout principles using Figma constraints Share prototype and screen annotations	CO 3
4	Lab Assignment 4 (Unit 4) Title: Enhancing Feedback and Accessibility in a Search Interface Sub-Objectives: 1. Design a search input component with default, hover, error, and loading states 2. Use micro interaction animations or icon feedback 3. Provide visual affordances for clickable areas 4. Evaluate the screen using Nielsen's heuristics 5. Document feedback and accessibility improvements	CO 4
5	Capstone Lab Assignment (All Units)	CO 5

	Title: Designing an End-to-End Event Registration UI with Visual Consistency Sub-Objectives: 1. Apply visual design principles, typography, and color to build a 5-screen responsive interface (Home, Events, Register, Success, Profile) 2. Use component-based design for buttons, cards, and nav elements 3. Ensure accessible color contrasts and UI states 4. Apply feedback elements and micro interactions for inputs, alerts, and loaders Present a final prototype with annotated design decisions and rationale	

Textbook:

Lidwell, W., Holden, K., & Butler, J. (2010). Universal Principles of Design (2nd ed.). Rockport Publishers.

Learning Experiences

Studio-Based Learning

- Students engage in hands-on, design-first learning where theoretical concepts (e.g., Gestalt laws, grids, typography) are applied immediately to digital interface challenges using tools like Figma and Adobe XD.
- Weekly lab assignments reinforce each design principle through iterative practice and peer critique, fostering design thinking and attention to detail.

Project-Based Approach

- Students work on incremental UI assignments—from a single card component to full responsive pages—culminating in a capstone project simulating a real-world event registration flow.
- Emphasis is placed on creating accessible, consistent, and aesthetically aligned user interfaces, grounded in business and user goals.

Critical Design Reflection

- Design critique sessions and heuristic evaluations develop students' ability to evaluate designs for usability, accessibility, and clarity.
- Exposure to anti-patterns and real UI flaws cultivates analytical skills and design sensibility.

Tool Proficiency & Industry Relevance

- Students gain proficiency in Figma plugins (e.g., Stark), layout systems (8pt grid), and design system standards (Material, Carbon).
- By the end of the course, students are equipped to design high-fidelity, responsive, and inclusive UI screens that meet current industry expectations.

Specialization Course-IV

Interaction Design & Prototyping

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Interaction Design & Prototyping**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		3-0-2	4	45
Type of Course:	DSE			
Pre-requisite(s), if any: Basic knowledge of UI design principles and visual hierarchy.				

Course Perspective: This course equips students with the theoretical foundation and practical skills necessary to design and prototype interactive digital interfaces. Students explore key principles of interaction design such as feedback, affordances, constraints, usability, and cognitive load. They build interactive flows and prototypes across fidelity levels using tools like Figma, ProtoPie, and Adobe XD. Emphasis is placed on component-based design, responsive behavior, usability, and validation through iterative testing. The course blends individual and collaborative learning experiences aligned with industry-standard UI/UX workflows. By course completion, students will be equipped to take up roles such as Interaction Designer, UI Prototyper, or UX Designer.

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

COs	Statements
CO 1	Understanding and apply core principles of interaction design and human-computer interaction (HCI).
CO 2	Applying microinteractions and animations to enhance feedback and usability.
CO 3	Designing effective user flows and screen-level interactions for web and mobile products.

CO 4	Creating low-, mid-, and high-fidelity prototypes using Figma, Adobe XD, or ProtoPie.
CO 5	Testing prototypes, analyzing feedback, and iteratively improve designs

Course Outline:

Unit Number: 1	**Title: Foundations of Interaction Design**	**No. of hours: 12**

Objective: Understand how interaction design supports human behavior, usability, and task flow in digital systems.

Contents:

- Definition and scope of Interaction Design (IxD); role in UX
- Core HCI principles: affordances, signifiers, feedback, constraints, mapping
- Usability heuristics (Nielsen's 10 principles) in the context of IxD
- Common interaction problems and design failures
- User flow logic and system response timing
- Touch, click, gesture, and voice interactions across platforms

Hands-On:

- Analyze a poorly designed app interface and identify 5+ interaction flaws
- Redesign a simple screen to improve feedback and reduce friction
- Document proposed changes referencing design principles

Unit Number: 2	**Title: Designing User Flows, Navigation & Wireframes**	**No. of hours: 12**

Objective: Define, visualize, and wireframe user paths and interactive navigation structures.

Contents:

- Distinction between user flows, task flows, and system flows
- Flowcharting with swimlanes and annotations (tools: FigJam, Miro)
- Screen state transitions and branching logic

- Low vs. mid vs. high fidelity wireframes: when to use what
- Best practices for layout, spacing, hierarchy, and screen zoning
- Annotating wireframes with interaction notes and behavior assumptions

Hands-On:

- Choose a real-world process (e.g., hotel booking) and create full user flow
- Design wireframes for 5–6 screens that support flow stages
- Annotate zones of interaction and page hierarchy using Figma

Unit Number: 3	**Title: Interactive Prototyping Techniques & Tools**	**No. of hours: 10**

Objective: Create interactive digital prototypes and understand design systems.

Contents:

- Overview of prototyping tools: Figma (interactive), ProtoPie (high fidelity)
- Building reusable components: buttons, input fields, tabs, navigation menus
- Interactive links: hover, tap, press, delay, auto-animate transitions
- Creating and managing variants, component states (e.g., toggle on/off)
- Design systems and consistency: typography, spacing, color styles
- Naming conventions, documentation, and file handoff best practices

Hands-On:

- Create a design system with at least 10 reusable components
- Design an interactive prototype for a 6-screen mobile/web app
- Use Figma's prototype panel to simulate real user interactions

Unit Number: 4	**Title: Microinteractions, Animation & Usability Testing**	**No. of hours: 11**

Objective: Improve interactivity through animation and evaluate prototypes for usability.

Contents:

- Understanding microinteractions: triggers, rules, feedback, loops

- Types: validation, hover/click response, swipe gestures, toggles, loaders
- Animation basics: easing, duration, delay, curve matching
- Using ProtoPie/Figma Smart Animate for motion design
- Creating usability test plans: goals, scripts, metrics
- Conducting tests: think-aloud method, remote testing, screen recording
- Analyzing observations and iterating designs accordingly

Hands-On:

- Apply 3 types of microinteractions (e.g., error prompt, tooltip, loading)
- Design a usability test plan with 3 tasks for a peer prototype
- Conduct 2–3 peer tests, log observations, iterate on design

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	Lab Assignment 1 (Unit 1) Title: "Evaluating and Enhancing Interaction in a Social Media App" Sub-Objectives: 1. Audit 2–3 screens from an existing social media app 2. Identify at least 5 usability issues or interaction design flaws 3. Apply interaction principles (affordance, feedback) to propose changes 4. Redesign the screen in Figma and annotate revised interactions	CO 1

	5. Submit heuristic evaluation report + redesign justification	
2	Lab Assignment 2 (Unit 2) Title: "User Flow and Wireframe Design for an Online Event Registration System" Sub-Objectives: 1. Define end-to-end user flow from browsing to ticket confirmation 2. Draw flow diagram using FigJam with alternate/exception paths 3. Wireframe at least 6 key screens (homepage, event detail, cart, checkout) 4. Annotate UI regions for interactivity and state changes 5. Review flow logic with peer feedback	CO 2
3	Lab Assignment 3 (Unit 3) Title: "Interactive Prototype of a Personal Finance Tracker" Sub-Objectives: 1. Define user journey: log expenses, set goals, view analytics 2. Create consistent layout and style guide (components, colors, fonts) 3. Design mid-fidelity screens for dashboard, entry form, settings, trends 4. Add interactive transitions (e.g., tab switch, modal open) 5. Present live prototype and document navigation behavior	CO 3
4	Lab Assignment 4 (Unit 4)	CO 4

	Title: "Microinteractions & Animation for a Job Application Form" Sub-Objectives: 1. Design a job application form with fields, error messages, confirmation 2. Add microinteractions (hover, click, success animation) 3. Use Smart Animate or ProtoPie to add motion effects 4. Test the form with 3 users, gather feedback 5. Refine animations and usability based on user input	
5	Capstone Lab Assignment (All Units) Title: "Designing an End-to-End Interactive Mobile App for Local Transport" Sub-Objectives: 1. Map full user flow for route planning, fare calculation, and payment 2. Create mid-to-high fidelity wireframes and prototypes in Figma 3. Use components and microinteractions for feedback and transitions 4. Conduct a usability test with 3–5 users, analyze behavior 5. Iterate and document final design improvements with rationale	CO 5

Textbook:

Saffer, D. (2010). Designing for Interaction: Creating Smart Applications and Clever Devices (2nd ed.). New Riders.

Learning Experiences

This course emphasizes **studio-based and experiential learning**, where students apply interaction design principles through:

- **Hands-on labs and tool-based practice** using Figma, ProtoPie, and Adobe XD.
- **Iterative prototyping tasks** that mirror real-world UX workflows, from wireframing to testing.
- **Critical peer reviews** and usability testing sessions that simulate industry collaboration.
- **Capstone project** where students design and test an end-to-end interactive app with real user feedback

Principles and Practices of System Design

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: Principles and Practices of System Design	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCMD501	4-0-0	4	60
Type of Course:	Major			
Pre-requisite(s), if any: Basic understanding of data structures, algorithms, computer organization, and programming concepts in C/C++ or Java.				

Course Perspective: This course introduces core principles for designing efficient and reliable computing systems. It focuses on abstraction, modularity, scalability, and performance, helping students solve real-world problems through structured design thinking. Through case studies and projects, learners gain practical skills in analyzing and building robust system architectures.

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

COs	Statements
CO 1	**Analyzing** fundamental principles and trade-offs involved in designing efficient, reliable, and scalable systems.
CO 2	**Applying** system design methodologies to solve real-world problems using abstraction, modularity, and interface design.
CO 3	**Evaluating** performance, reliability, and fault tolerance aspects of system components and architectures.
CO 4	**Designing** system-level solutions by integrating hardware, software, and network components to meet specified requirements.

Course Outline:

Unit Number: 1	**Title: Fundamentals & Interview Mind-set**	**No. of hours: 16**
Content:		

Key Concepts

- Translating vague product briefs into functional / non-functional requirements.
- Back-of-envelope sizing: QPS, storage, bandwidth, p50/p95 latency budgets.
- Request/response lifecycle (DNS → TLS → LB → App → DB).
- Caching layers, CDNs, browser connection limits, TLS 1.3 vs 1.2 savings.
- Authentication & sessions (JWT vs opaque IDs, OAuth 2.0 + PKCE).
- Cost awareness: egress, cache hit vs miss economics.

Hands-on / Mock Tasks

- Pastebin v2 (private pastes, syntax highlighting, 50 k QPS) — one-page HLD + latency cost breakdown.
- Secure login flow for a React SPA (OAuth 2.0 + PKCE) — sequence diagram & prototype.
- Latency Budget Drill — allocate 200 ms SLO across DNS, TLS, network, datastore; defend on whiteboard.
- TLS Handshake Walk-through — pinpoint round trips saved by TLS 1.3; live packet capture in Wireshark.

Unit Number: 2	**Title: High-Level Design for Scale**	**No. of hours: 15**

Content:

Key Concepts

- Data partitioning: range vs hash sharding, consistent hashing, re-shard workflows.
- Replication & failover (Redis + Sentinel, leader election basics).
- Read-heavy vs write-heavy paths; global rate-limiters; fan-out vs fan-in.
- Queues & pipelines (Kafka, RabbitMQ) for media processing and dead-letter handling.
- Caching strategies: write-through, write-around, cache-invalidation pitfalls.
- Trade-offs: SQL vs NoSQL, CAP & PACELC in interview language.

Hands-on / Mock Tasks

- Pinterest-style image pinning write-path — design & justify sharding choice under 20:1 user skew.
- Global rate limiter (2 000 req/min/user) across 5 data-centres — code a Redis+Lua prototype and load-test.
- Thumbnail pipeline — message-queue–driven, 500 uploads/s → 7 derived sizes; produce throughput metrics.
- Debate Session — Cassandra vs DynamoDB vs CockroachDB for a log store; 5-min lightning talks.

Unit Number: 3	Title: Low-Level Design & Patterns	No. of hours: 14

Key Concepts

- SOLID principles, design patterns (Builder, Strategy, Observer, Command, State, Composite).
- Concurrency basics: locks, thread-safety, double-checked locking hazards, immutability.
- Testing for correctness: unit tests, mocks/stubs (Mockito/FakeIt), race-condition simulators.
- Clean interfaces & dependency inversion for pluggable components.
- OOP vs data-oriented design trade-offs in interview scenarios.

Hands-on / Mock Tasks

- Parking-Lot / Elevator — full class diagram + thread-safe slot allocation code.
- RateLimiter library — Strategy pattern switch between token-bucket and leaky-bucket at runtime; JMH benchmark.
- Undo/Redo stack for a text editor (Command + Memento pattern); unit-test for edge cases.
- Movie-Ticket LLD — seat lock, payment, release flow; handle concurrency & design integration tests

Unit Number: 4	Title: Distributed Systems, Optimisation & Capstone Mock Panels	No. of hours: 15

Content:

Key Concepts

- Consensus & leader election (Raft overview, Kubernetes Lease API).
- Fault tolerance: circuit breakers, retries, idempotency keys, chaos engineering basics.
- SLO / SLA design: RED metrics (rate, errors, duration), failure budgets.
- Real-time streams & stateful services (adaptive bitrate, live leaderboards, group chat).
- Cost/performance optimisation: GC tuning, cache eviction maths, index strategies.

Hands-on / Mock Tasks

- WhatsApp-style group chat — HLD emphasising group fan-out, offline message store, key rotation.

- Flash-sale inventory — build a Redis-based atomic counter rejecting oversells at 20 000 checkouts/s; benchmark.
- Chaos Monkey lab — kill 25 % of pods in a K8s test cluster; measure auto-healing and alerting response.
- Capstone Mock Panel: "Design Zoom-like video conferencing for 10 k concurrent viewers" — 40-min board-style defence with peer & faculty rubric (requirements, sizing, architecture, trade-offs, failure modes, cost).

Learning Experiences

Inside Classroom

- **Conceptual Understanding**

  Interactive lectures and discussions on system design trade-offs, architecture patterns, scalability, and performance metrics.
- **Mock Interviews & Case Studies**

  Real-world tasks such as Pastebin redesign, OAuth login flows, rate limiters, and distributed systems problems.
- **Visual & Hands-on Learning**

  Use of diagrams, sequence flows, whiteboard drills, packet capture demos (Wireshark), and design justifications.
- **Collaborative Debates & Walkthroughs**

  Comparative evaluations (e.g., SQL vs NoSQL, DynamoDB vs Cassandra) to understand system design choices.
- **Low-Level Design Practices**

  Design patterns, concurrency, and object-oriented programming demonstrated through live coding and unit testing.

Outside Classroom

- **Hands-on Projects**

  Capstone challenges like designing scalable chat systems or flash-sale inventory counters using Redis/Kubernetes.
- **Team Activities**

Group design boards, architecture reviews, mock defense panels for complex systems (e.g., Zoom clone, group chat apps).

- **Performance & Fault Testing**
  Load testing, chaos engineering labs, benchmarking using tools like JMH, Redis+Lua scripting.
- **Self-Learning Resources**
  Encouraged to explore open-source design patterns, case studies, and contribute to real-world scalable system design challenges on GitHub.

Text and Reference Book

System Design Interview – An Insider's Guide by Alex Xu (Vol I & II) [ISBN: 979-8664653403]

Other Recommended Resources:

- Designing Data-Intensive Applications – Martin Kleppmann
- GitHub: System Design Primer
- Educative.io – Grokking the System Design Interview
- YouTube Channels: Gaurav Sen, Tech Dummies Narendra

COMPETITIVE CODING -III

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: COMPETITIVE CODING -III	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		2-0-0	2	30
Type of Course:	SEC			
Pre-requisite(s), if any: Competitive Coding-II, Fundamentals of Data Structures and Algorithms				

Course Perspective: This course enhances advanced problem-solving skills through intensive practice in data structures, algorithms, and coding challenges, preparing students for high-level programming contests and technical interviews.

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

COs	Statements
CO 1	Applying bit manipulation, number theory, and string algorithms to solve computational problems.
CO 2	Analyzing and implement advanced backtracking and recursion techniques for combinatorial problems.
CO 3	Evaluating sliding window techniques and two-pointer algorithms for efficient solutions.
CO 4	Solving graph problems using foundational and advanced concepts in competitive programming.

Course Outline:

SESSION WISE DETAILS		
Session 1	Bit Manipulation Introduction.	No. of hours: 2
Content Summary: Introduction to AND, OR, XOR operations, Count Set/unset Bits, Toggle a given kth bit, check if nth bit is set or unset, Check Power of Two/Four.		

Session: 2	Bit Manipulation-II.	No. of hours: 2
Content Summary: Counting Single Number 1, Single number 2, Subsets using Bits (power set problem) , Find Missing number, Duplicate Numbers.		
Session: 3	Number theory basics.	No. of hours: 2
Content Summary: Sieve of Eratosthenes, Modular Arithmetic, Modular Exponentiation, Chinese Remainder Theorem		
Session: 4	Mathematical Algorithms.	No. of hours: 2
Content Summary: Euler's Totient Function, Permutations and Combinations, Inclusion-Exclusion Principle, Catalan Numbers.		
Session 5	Advance Recursion.	No. of hours: 2
Content Summary: print all subset, permutation of a string, find all unique subset		
Session: 6	Backtracking I	No. of hours: 2
Content Summary: rat in maze, rat in a maze all path, N Queens		
Session: 7	Backtracking-2	No. of hours: 2
Content Summary: combination, combination sum, combination sum-2		
Session: 8	Backtracking-3	No. of hours: 2
Content Summary: generate parentheses, subset-2, sudoku solver		
Session : 9	Greedy I	No. of hours: 2
Content Summary: assign cookies, array partition, can place flower, lemonade change		
Session: 10	Greedy-II.	No. of hours: 2
Content Summary: Activity selection, minimum platform, coin change		
Session : 11	Greedy-III.	No. of hours: 2
Content Summary: max chunk to make sorted, max chunk to make sorted-2, 0/1 knapsack.		
Session: 12	Graph Introduction and representation.	No. of hours: 2
Content Summary: Introduction, Representation using adjacency matrix and adjacency list		
Session: 13	Graph-Traversal Algorithm.	No. of hours: 2

Content Summary: Graph Traversal BFS(Breadth first search) and DFS(Depth first search)		
Session: 14	Graph-III	No. of hours: 2
Content Summary : Connected Components, Detecting Cycles in Graphs		
Session: 15	Graph Problems-IV.	No. of hours: 2
Content summary: find if path exist(has path), print all path from source to destination, Number of Island		
Session: 16	Advanced Graph.	No. of hours: 2
Content summary: Number of Provinces, Flood Fill, Number of closed islands.		
Session: 17	Minimum Spanning Tree algorithms.	No. of hours: 2
Content summary: Prim's Algorithm, Kruskal's algorithm.		
Session: 18	Shortest Path Algorithm.	No. of hours: 2
Content summary: Dijkstra algorithm, Bellman ford algorithm.		
Session: 19	Summarizing the Semester 5.	No. of hours: 2
Content summary: Company specific problems on Graphs, sliding window and recursion.		
Session: 20	Summarizing the Semester 5.	No. of hours: 2
Content summary: Company specific problems on Graphs, sliding window and recursion.		

Reference Books:

- *Elements of Programming Interviews (EPI)* – Adnan Aziz, Tsung-Hsien Lee, Amit Prakash Great for interview-style problems with solutions in C++, Java, and Python editions.
- *Cracking the Coding Interview* – Gayle Laakmann McDowell Best for coding interview prep with 189 questions, concepts, and system design.

Evaluation Criteria

Criteria	Internal (50 marks)
After 2 weeks Coding Test	No. of tests: 6 Marks per test: 5 Total marks: 30
Mid Term Coding Test	20 Marks

External (50 marks)
End Term Paper

Summer Internship-II

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Summer Internship-II**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCIN504	0-0-4	2	30
Type of Course: INT	INT			
Pre-requisite(s), if any: Competitive Coding-II, Fundamentals of Data Structures and Algorithms				

Course Perspective: The Summer Internship Program (1st June – 25th July 2025) is designed to integrate

academic learning with real-world professional experiences, enabling students to

apply theoretical knowledge to practical situations. It forms a mandatory part of the

Semester III for students currently in Semester II, carrying a weightage of **2 academic**

credits.

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

COs	Statements
CO 1	Applying theoretical knowledge to practical work environments and real-world problems.
CO 2	Demonstrating professional skills, workplace ethics, and teamwork in an organizational setting.
CO 3	Developing problem-solving, critical thinking, and communication skills through experiential learning.
CO 4	Preparing technical reports and presentations reflecting the understanding and outcomes of the internship experience.

The key objectives of the Summer Internship Program are:

To enhance professional skills and industry readiness.

To expose students to real-world technical, managerial, and research practices.

To promote self-learning, professional responsibility, and critical thinking.

To foster connections between academic knowledge and industry practices.

Duration

The duration of the internship will be 6-8 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:

- **Industry Internship (Offline):**
  Students must produce a joining letter at the start and a relieving letter upon completion.
- **Global Certifications:**
  Students can opt for globally recognized certification programs relevant to their field of study.
- **Government/Research Institution Internship:**
  Students can engage in a research internship with premier government or research organizations such as IITs, IISc, ISRO, DRDO, CSIR, NPL, etc.
- **On-Campus Industry Internship Programs:**
  The university will offer on-campus internships in collaboration with industry partners.

Deliverables and Documentation:

Each student must submit the following after completing their internship/certification:

Deliverable	Description	Marks
Summer Internship File	A detailed report/file based on the provided format including objectives, methodology, learnings, and reflections.	10 Marks
Video Presentation	A 7–10-minute recorded video presentation showcasing work done during the internship/certification. The template of slides will be shared.	20 Marks
Certificate of Completion	A color-printed certificate on bond paper from the host organization/certification body, mentioning duration, role/project.	70 Marks

Evaluation Metrics

The Summer Internship will be evaluated based on the following comprehensive criteria:

Evaluation Component	Weightage	Description
Internship Report/File	10%	Completeness, professional formatting, relevance to internship tasks.
Video Presentation	20%	Content quality, clarity, communication skills, professional presentation.
Certificate of Completion	70%	Authenticity, completion of internship/certification within stipulated time, relevance to program objectives.

Internship Evaluation Rubric:

S. N.	Component	Sub-Component / Criteria	Marks
1	**Internship Certificate**	**Relevance to Core Subjects**	**20 Marks**
		- Directly relates to core subjects	20
		- Partially relates to core subjects	15
		- Minimally relates to core subjects	10
		- Not relevant	0
2	**Report Submission**	**Structure and Organization**	**10 Marks**
		- Well-structured and organized report	10
		- Moderately structured report	7
		- Poorly structured report	3
		- No structure	0
3	**Solo Video-Based Evaluation**	**a. Technical / Professional / Soft Skills Acquired**	**10 Marks**
		- Highly relevant and advanced technical skills	10
		- Moderately relevant technical skills	8
		- Basic technical skills	5
		- No new skills acquired	0
		b. Content Delivery	**10 Marks**
		- Clear, engaging, and thorough delivery	10
		- Clear but less engaging delivery	7
		- Somewhat clear and engaging delivery	3
		- Unclear and disengaging delivery	0
		c. Visual Aids & Communication Skills	**10 Marks**

		- Effective visual aids + excellent communication skills	10
		- Moderate visual aids + good communication skills	7
		- Basic visual aids + fair communication skills	3
		- No visual aids + poor communication skills	0
4	**Internship Duration**	**Weeks Completed**	**10 Marks**
		- 6–8 weeks completed	10
		- 4–6 weeks completed	8
		- Less than 1 month	5
5	**Outcome of the Internship**	**Application / Project / Key Learnings & Findings**	**30 Marks**
		- Clear, outcome-based project with applied learnings and key findings	25–30
		- Moderate outcome with partial application and findings	15–24
		- Minimal outcome, unclear learning/application	0–14

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.

VAC-II

Startup Ideation & Validation

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name : **Startup Ideation & Validation**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- II			
Pre-requisite(s):				

Course Perspective. This practical course guides students through the entrepreneurial journey from identifying a real-world problem to building and validating a functional MVP. Students will engage in hands-on market research, rapid prototyping using no-code tools, and learn the fundamentals of startup formation, pitching, and pre-seed preparation through real-world simulations and case studies.

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

COs	**Statements**
CO1	Identifying and analyze real-world problems through user interviews and persona development
CO2	Designing and test functional MVPs using no-code tools and usability feedback
CO3	Applying basic financial, legal, and equity concepts to startup planning
CO4	Preparing and deliver investor-ready pitch decks and videos
CO5	Evaluating startup viability based on market response and key metrics

Course Outline:

Unit Number: 1	**Problem Discovery & Market Research**	**No.of hours: 8**
Content: • Conduct 15+ user interviews using The Mom Test framework		

- Create pain-point heat maps and empathy maps
- Estimate TAM, SAM, SOM with real-world assumptions
- Draft user personas and ICPs (Ideal Customer Profiles)
- Analyze competitors using SWOT and differentiation matrix
- Prepare a Lean Canvas and 2-min problem narrative video

Unit Number: 2	**MVP Development & Validation**	**No. of hours: 8**

Content:

- Choose and build MVP using no-code tools like Glide, Carrd, Webflow
- Integrate form submission or payment links (Google Form/Stripe)
- Conduct 5-user hallway usability testing sessions
- Instrument MVP using GA-4 or Hotjar to track user behavior
- Iterate copy, design, CTA based on real feedback
- Define MVP success criteria and measure user engagement

Unit Number: 3	**Legal, Equity & Financial Readiness**	**No. of hours: 8**

Content:

- Calculate CAC, LTV, break-even, and gross margin
- Apply the "Slicing Pie" formula for equity distribution
- Prepare draft founder agreement, NDA, and employment terms
- Navigate MCA-21 and Startup India portals for incorporation
- Prepare a basic IP strategy and understand trademark/copyright basics

Unit Number: 4	**Pitch Preparation & Pre-Seed Readiness**	**No. of hours: 6**

Content:

- Design a 10-slide pitch deck (problem, solution, market, business model, team, etc.)
- Record a 90-second pitch video
- Assemble a startup data-room (Notion, Google Drive)
- Present to mock investor panel and receive feedback
- Explore alternative funding sources: grants, crowdfunding, bootstrapping

Learning Experiences – Startup Ideation & Validation

Inside Classroom Learning

- Brainstorm and shortlist startup ideas using tools like SCAMPER, Mind Mapping, and the Golden Circle.
- Analyze problem-solution fit and customer pain points through case studies and persona mapping.
- Develop a basic Lean Canvas for a proposed idea and present it in class.
- Study examples of validated startups and identify key metrics used during the validation stage.
- Conduct mock interviews or role plays simulating customer discovery sessions.

Outside Classroom Learning

- Conduct a field survey or online research to validate the real-world relevance of your startup idea.
- Perform competitor analysis using tools like SWOT or Porter's Five Forces on similar market players.
- Interview potential users/customers to understand their needs, preferences, and feedback.
- Create a Minimum Viable Product (MVP) concept and test it with a small group of users.
- Document validation insights and pivot/refine the idea based on user responses.

Textbooks & Resources:

Fitzpatrick, R. (2013). *The Mom Test: How to Talk to Customers and Learn If Your Business is a Good Idea When Everyone is Lying to You*. Robfitz Ltd.

Case Studies to Demonstrate:

Zappos – Customer discovery via concierge MVP

Facebook – Ultra-lean MVP for initial traction

Uber – Founder equity & investor negotiation

WhatsApp – Pitch strategy and bootstrap journey

AirBnB – MVP testing via real-world hosts & guests

Digital Wellbeing and Tech-Life Balance

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course 2: VAC-II **(Digital Wellbeing and Tech-Life Balance)**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- II			
Pre-requisite(s):				

Course Perspective. This course empowers students to build a conscious, healthy relationship with digital technologies. It offers hands-on strategies and reflective exercises to improve attention, reduce digital fatigue, avoid tech burnout, and design a lifestyle that integrates productivity, purpose, and emotional resilience. Using science-backed frameworks, mindfulness tools, and tech tracking apps, students will craft their own personalized digital wellbeing blueprint.

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

COs	**Statements**
CO1	Identifying and evaluate patterns of digital overuse and distraction in daily life.
CO2	Applying scientific tools and mindfulness techniques to build tech-life balance.
CO3	Designin personalized interventions to manage screen time and improve mental health.
CO4	Reflecting on lifestyle habits using self-tracking apps, journaling, and productivity frameworks.

Course Outline:

Unit Number: 1	**Understanding Digital Overload and Attention Economy**	**No.of hours: 8**
Content:		

- Impact of screen addiction, doom scrolling, and attention fragmentation.
- Cognitive science of attention and multitasking myths.
- How social media and apps hijack dopamine pathways (The Hooked Model).
- Reflective Activity: Screen time audit (using Digital Wellbeing, RescueTime, or ScreenZen).
- Case Study: Tristan Harris & the Center for Humane Technology.

Unit Number: 2	**Building Awareness and Mindful Tech Usage**	**No. of hours: 8**

Content:

- Introduction to mindfulness in a digital context.
- Breathwork and grounding techniques (practical).
- Deep Work vs. Shallow Work (Cal Newport framework).
- Curating digital environments: declutter apps, notifications, feeds.
- Activity: Implement a 2-day digital detox and write a reflection journal.

Unit Number: 3	**Designing Tech-Life Routines & Energy Management**	**No. of hours: 8**

Content:

- Pomodoro, time-blocking, and digital sabbath practices. Understanding circadian rhythms and tech-induced sleep disruption.
- Setting tech boundaries in relationships and workspace.
- Habit loops, nudges, and digital minimalism (James Clear & Nir Eyal).
- Assignment: Create a 7-day "Digital Wellbeing Plan" with routines and tools.

Unit Number: 4	**Tools, Technologies & Capstone Project (7 hours)**	**No. of hours: 7**

Content:

- Tools for digital wellness: Forest App, Notion for habit tracking, Freedom, Mindful Browsing.
- Understanding algorithmic bias and digital echo chambers.
- Mental health tech (e.g., Headspace, Moodpath, Insight Timer).

- Capstone: Submit a personal "Tech-Life Balance Blueprint" – combining tracked data, insights, goals, and productivity plan

Learning Experiences

Inside Classroom Learning

- Analyze screen time data and identify digital overuse patterns using digital wellbeing tools.
- Participate in guided reflections and group discussions on the psychological impact of social media.
- Study models like the **Attention Economy**, **Dopamine Loop**, and **Digital Minimalism**.
- Practice mindfulness techniques (e.g., deep breathing, silent reflection) as in-class digital detox sessions.
- Evaluate real-world case studies on digital burnout and discuss preventive strategies.

Outside Classroom Learning

- Track and reflect on your own digital habits using apps like Digital Wellbeing, Forest, or RescueTime.
- Observe and document tech usage behavior in a family or peer group setting.
- Design and implement a 3-day personal **Digital Detox Challenge**, then journal the experience.
- Conduct a mini-campaign (online or offline) promoting healthy tech habits in your community or campus.
- Create and share digital wellbeing content (poster, reel, infographic, blog) to spread awareness.

Textbooks & Resources:

Knapp, J., & Zeratsky, J. (2018). *Make Time: How to Focus on What Matters Every Day*. Currency.

Digital Communication, Personal Branding & Influence

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: VAC-II **(Digital Communication, Personal Branding & Influence)**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- II			
Pre-requisite(s):				

Course Perspective. This course equips students with practical tools and strategies to build an authentic digital identity, master online communication, and develop influence across platforms. Through experiential learning using tools like LinkedIn, Canva, personal blogs, and video content creation, students will craft and manage their personal brand with confidence and clarity. They will also learn digital etiquette, storytelling, content planning, and the science of online influence to stand out in the modern professional landscape. **The Course Outcomes (COs).** On completion of the course the participants will be:

COs	**Statements**
CO1	Applying principles of effective digital communication across platforms.
CO2	Creating and managing a personal brand that aligns with their professional goals.
CO3	Using storytelling and content strategy to build digital visibility and engagement.
CO4	Leveraging digital tools to grow influence, network, and credibility.

Course Outline:

<table>
<tr><td>Unit Number: 1</td><td>Foundations of Digital Communication & Online Presence</td><td>No.of hours: 8</td></tr>
<tr><td colspan="3">Content:
Digital identity: What recruiters and collaborators see online Principles of online communication: clarity, tone, audience analysis
Digital etiquette and reputation management
Self-audit activity: Google yourself and reflect
Hands-on: Create or refine a professional email signature, personal bio, and LinkedIn headline</td></tr>
<tr><td>Unit Number: 2</td><td>Personal Branding: Discover, Design, Deliver</td><td>No. of hours: 8</td></tr>
<tr><td colspan="3">Content:
What is a personal brand? Brand archetypes and authenticity
Defining your niche, values, and mission statement
Building your personal branding kit (bio, profile, visual identity)
Visual storytelling using Canva, Notion, or Figma
Hands-on: Design a personal brand mood board and a basic personal logo or template</td></tr>
<tr><td>Unit Number: 3</td><td>Designing Tech-Life Routines & Energy Management (7 hours)</td><td>No. of hours: 8</td></tr>
<tr><td colspan="3">Content:
Content Strategy, Storytelling & Influence (7 hours)
Principles of storytelling for digital content (Hero’s Journey, Hook-Story-Offer)
Writing impactful posts, blogs, and video scripts
Types of content: thought leadership, tutorials, behind-the-scenes, portfolio, etc.
Building content calendar (30-day plan) using Notion or Trello</td></tr>
</table>

Hands-on: Write a blog/article or record a 2-min personal pitch video		
Unit Number: 4	**Growing Digital Influence & Engagement (7 hours)**	**No. of hours: 7**
Content: • Social platforms deep dive: **LinkedIn, YouTube, Medium, Twitter** • Engagement tactics: comments, shares, collaborations, hashtags, tagging • Building a digital portfolio or personal website (using **Carrd**, **Notion**, or **WordPress**) • Case studies: Students analyze 2 successful personal brands in their domain Capstone: Launch a live LinkedIn campaign (e.g., 5-day #showyourwork challenge)		

Learning Experiences

Inside Classroom Learning

- Analyze different digital communication styles (email, social media, blogs) and identify key etiquette rules.
- Participate in mock scenarios to practice persuasive communication and personal pitch delivery.
- Study successful personal brands and identify the elements of their digital identity and influence strategy.
- Create a personal branding framework using tools like SWOT analysis and Ikigai.
- Develop a content plan (text/image/video) tailored for a specific digital platform (LinkedIn, Instagram, etc.).

Outside Classroom Learning

- Audit and update your own digital presence (social media profiles, bio, posts) for professional alignment.
- Publish one blog post, article, or video reflecting your niche, values, or expertise.

- Network with professionals in your domain via platforms like LinkedIn or Twitter and document engagement.
- Run a short digital campaign (e.g., Instagram series or LinkedIn poll) to gauge your influence.
- Collect feedback from peers/mentors on your digital persona and refine it accordingly.

Textbooks & Resources:

Clark, D. (2015). *Stand Out: How to Find Your Breakthrough Idea and Build a Following Around It*. Portfolio.

Purposeful Living and Ikigai: Designing a Meaningful Life

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Purposeful Living and Ikigai: Designing a Meaningful Life**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	1-0-2	2	30
Type of Course:	VAC- II			
Pre-requisite(s):				

Course Perspective. This course helps students discover a deeper purpose by aligning their passions, values, and strengths with real-world needs — inspired by the Japanese philosophy of **Ikigai**. Through reflective exercises, storytelling, journaling, and life design tools, students will explore what brings them joy, what they are good at, and how they can contribute meaningfully to the world. Rooted in **positive psychology, life design from Stanford d.school**, and Eastern philosophies, the course nurtures self-awareness, life clarity, and intrinsic motivation.

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

COs	**Statements**
CO1	Understanding and apply the Ikigai framework to explore personal meaning and life purpose.
CO2	Reflecting on personal strengths, values, passions, and world needs through structured exercises.
CO3	Designing and prototype life pathways using tools like Life Compass, Purpose Canvas, and Vision Board.
CO4	Cultivating purpose-driven decision-making and self-motivation using practical life design strategies.

Course Outline:

Unit Number: 1	Introduction to Purpose and the Ikigai Philosophy (8 hours)	No.of hours: 8
Content: • Understanding Ikigai: The intersection of Passion, Mission, Vocation, and Profession • Case studies from Okinawa and purpose-driven communities • Common myths of success vs. meaning • Self-assessment activities: Purpose journaling, guided reflection • Hands-on: Draw your first Ikigai Venn Diagram + Purpose Journal Entry #1		
Unit Number: 2	**Self-Discovery through Reflection and Strength Mapping (8 hours)**	**No. of hours: 8**
Content: • Discovering personal values, beliefs, and strengths • VIA Strengths Survey, MBTI/16-Personality, or Gallup CliftonStrengths (Free versions) • Identifying peak experiences and flow states • Activity: Life Timeline Mapping – chart highs, lows, and lessons Assignment: Prepare a "Strengths & Values Map" + Purpose Journal Entry #2		
Unit Number: 3	**Life Design and Prototyping Your Future (7 hours)**	**No. of hours: 8**
Content: • Introduction to Design Thinking applied to life (Stanford Life Design Lab) • Odyssey Plans: Designing 3 alternative life paths (5-year plans) • Purpose Canvas: From intention to small experiments • Tools: Notion/Lucidchart for visual life mapping • Hands-on: Build a Digital Vision Board and 3 Odyssey Pathways		
Unit Number: 4	**Purposeful Action, Mindfulness & Legacy Thinking (7 hours)**	**No. of hours: 7**

Content:

- Daily routines, rituals, and habit design for meaningful living
- The role of mindfulness, stillness, and journaling
- Balancing ambition with compassion; service as purpose
- Capstone: Final "Ikigai Presentation" – A 5-minute talk or digital story about your meaningful life experiment
- Purpose Journal Entry #3 (Reflection on growth)

Learning Experiences

Inside Classroom Learning

- Reflect on personal values, strengths, and passions through guided journaling and group sharing.
- Understand the concept of **Ikigai** and map your own life dimensions using the Ikigai Venn diagram.
- Explore case studies of individuals who have aligned work with purpose across diverse fields.
- Participate in workshops on goal setting, visualization, and value-based decision making.
- Engage in class discussions on meaning, happiness, and fulfillment from philosophical and psychological perspectives.

Outside Classroom Learning

- Interview 2–3 people from different walks of life to explore how they found or are pursuing their Ikigai.
- Maintain a **"Purpose Journal"** for 7–10 days documenting meaningful moments, thoughts, or realizations.
- Try a new activity (volunteering, creative hobby, skill-building) and reflect on how it contributes to purpose.
- Design a **Life Vision Board** and present it to peers or mentors for feedback and refinement.

- Share a personal story, blog, or video on what purpose and meaning mean to you—and how you plan to pursue them.

Textbooks & Resources:

García, H., & Miralles, F. (2017). *Ikigai: The Japanese Secret to a Long and Happy Life*. Penguin Books.

Yogic Science and Inner Engineering for Personal Mastery

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Yogic Science and Inner Engineering for Personal Mastery**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- II			
Pre-requisite(s):				

Course Perspective. This course introduces students to the science of inner transformation based on classical yogic practices and the modern framework of Inner Engineering. It is designed to help students gain mastery over their body, mind, and energy through self-discipline, awareness, breathwork, and meditative practices. Blending the ancient yogic principles of Patanjali, Hatha Yoga, and Isha Foundation's Inner Engineering philosophy with neuroscience and emotional intelligence models, the course enhances well-being, focus, and inner clarity.

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

COs	**Statements**
CO1	Applying yogic principles to cultivate physical, emotional, and mental balance.
CO2	Practicing foundational asanas, pranayama, and dhyana (meditation) techniques.
CO3	Understanding the connection between breath, awareness, energy, and inner transformation.
CO4	Designing a personalized routine for holistic well-being and stress mastery.

Course Outline:

Unit Number: 1	Foundations of Yogic Science & Inner Discipline (8 hours)	No.of hours: 8

Content:

- Overview of Yogic Science: Origin, key schools (Patanjali Yoga Sutras, Hatha Yoga, Raja Yoga)
- Pancha Koshas (Five layers of existence)
- Yamas and Niyamas (Ethical disciplines)
- Concept of Karma Yoga and inner mastery
- Activity: Daily introspection using "Self-Awareness Log"

Unit Number: 2	Body-Mind Harmony through Asanas and Breathwork (8 hours)	No. of hours: 8

Content:

- Surya Namaskar (Sun Salutation) – Step-by-step learning
- Foundational Asanas for posture, energy, and vitality
- Breath and emotion regulation: Anulom Vilom, Bhramari, Nadi Shodhana
- Neuroscience of breath and parasympathetic activation
- Hands-on: Practice journal + breath awareness diary

Unit Number: 3	Inner Engineering & Emotional Mastery (7 hours)	No. of hours: 8

Content:

- Introduction to Inner Engineering (based on Sadhguru's work)
- Managing compulsiveness: response vs. reaction
- Emotional intelligence from yogic and psychological perspectives
- Tools for emotional mastery: guided meditation, visualization, self-inquiry
- Group Activity: "Emotion Lab" – identify triggers, patterns, and transformation map

Unit Number: 4	Meditative Practices and Designing a Life of Clarity (7 hours)	No. of hours: 7

Content:

- Introduction to Dhyana (Meditation): mindfulness vs. yogic meditation

- Trataka, Yoga Nidra, Isha Kriya (guided experience)
- Designing a personal sadhana (daily practice routine)
- Final Capstone: "My Inner Engineering Blueprint" – submission of lifestyle integration plan and short oral reflection

Learning Experiences

Inside Classroom Learning

- Learn and practice foundational yogic techniques—asana, pranayama, and dhyana—under guided instruction.
- Study ancient yogic texts (like Patanjali's Yoga Sutras) to understand the philosophy behind personal mastery.
- Explore concepts such as Pancha Kosha, Chakras, and Gunās through interactive discussions.
- Participate in self-awareness exercises like body-scan meditations and journaling for emotional clarity.
- Reflect on case studies of transformation through yogic living and inner engineering practices.

Outside Classroom Learning

- Maintain a daily yogic routine (asana, breathing, and mindfulness) for 10–14 days and journal your progress.
- Observe and document changes in behavior, concentration, or emotional stability post-practice.
- Attend a local yoga/meditation session or satsang and reflect on the collective experience.
- Create a personal Inner Engineering Plan based on your lifestyle goals, weaknesses, and aspirations.
- Share your journey of inner transformation through a creative medium—poem, blog, video, or artwork.

Textbooks & Resources:

Satchidananda, S. (2012). The Yoga Sutras of Patanjali: Commentary by Sri Swami Satchidananda. Integral Yoga Publications.

Mindfulness and Emotional Intelligence

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Mindfulness and Emotional Intelligence**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- II			
Pre-requisite(s):				

Course Perspective. This course explores the science and practice of mindfulness and emotional intelligence (EI) as key tools for personal and professional success. Students will engage in reflective journaling, guided meditation, empathy training, and emotional regulation practices. Based on frameworks from **Daniel Goleman, Jon Kabat-Zinn**, and **Tara Brach**, this course helps learners enhance their awareness, manage stress, and navigate relationships more effectively through conscious attention and compassion.

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

COs	**Statements**
CO1	Understanding the foundations of emotional intelligence and the neuroscience of mindfulness.
CO2	Practicing mindfulness techniques to improve attention, reduce stress, and manage emotions.
CO3	Applying emotional regulation and empathy tools in personal and academic situations.
CO4	Developing a personal mindfulness and EI toolkit for long-term self-awareness and growth.

Course Outline:

Unit Number: 1	**Foundations of Emotional Intelligence & Mindfulness (8 hours**)	**No.of hours: 8**
Content:		

- Introduction to EI (Daniel Goleman's Model): Self-awareness, self-regulation, motivation, empathy, social skills
- Mindfulness defined: Present moment awareness with non-judgment
- The neuroscience of attention, emotions, and stress response
- Guided Activity: Body scan meditation and mindful observation
- Assignment: Begin "Mindful Moments" daily journal (short entries)

Unit Number: 2	**Emotional Self-Awareness & Regulation (8 hours)**	**No. of hours: 8**

Content:

- Identifying emotional triggers, patterns, and reactions
- Naming emotions to tame them (emotion vocabulary)
- Breath-based calming techniques (box breathing, 4-7-8 method)
- Guided Activity: "STOP" method practice (Stop, Take a breath, Observe, Proceed)
- Reflection Task: Emotional audit of one challenging situation

Unit Number: 3	**Empathy, Compassion, and Relational Intelligence (7 hours)**	**No. of hours: 8**

Content:

- Daily habits for emotional wellness: Gratitude, journaling, mindful breaks
- Digital mindfulness: Tech boundaries and mindful screen time
- Tools for sustained mindfulness: Apps, trackers, self-coaching
- Capstone: "My EI and Mindfulness Toolkit" – a personal plan with reflections, habits, and practices
- Presentation: Short video or oral presentation of transformation experience

Unit Number: 4	**Designing a Mindful Life & Capstone Practice (7 hours)**	**No. of hours: 7**

- Daily habits for emotional wellness: Gratitude, journaling, mindful breaks
- Digital mindfulness: Tech boundaries and mindful screen time
- Tools for sustained mindfulness: Apps, trackers, self-coaching

- Capstone: "My EI and Mindfulness Toolkit" – a personal plan with reflections, habits, and practices
- Presentation: Short video or oral presentation of transformation experience

Learning Experiences

Inside Classroom Learning

- Practice guided mindfulness exercises such as breathing techniques, body scans, and focused attention meditation.
- Explore the components of Emotional Intelligence (EQ)—self-awareness, self-regulation, motivation, empathy, and social skills—through interactive sessions.
- Analyze real-life situations or case studies involving emotional challenges and discuss mindful responses.
- Participate in group activities and role-plays to improve emotional expression and active listening.
- Maintain a reflective journal to log daily emotional triggers and mindful responses.

Outside Classroom Learning

- Implement a daily mindfulness practice (5–10 minutes) for 2 weeks and record observations on mental clarity and stress.
- Apply emotional intelligence techniques in real interactions (e.g., conflict resolution, peer communication) and reflect on the outcome.
- Interview 2 individuals (mentor, parent, or peer) about how they manage emotions and stress in life or work.
- Design and conduct a mini-awareness campaign (poster, video, Instagram reel) on "Why Mindfulness Matters."
- Create a personal EQ Growth Plan with short- and long-term goals for improving emotional resilience.

Textbooks & Resources:

Goleman, D. (2006). *Emotional Intelligence: Why It Can Matter More Than IQ*. Bantam

Building Your Personal Digital Brand with AI Tools

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name 7: **Building Your Personal Digital Brand with AI Tools**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- II			
Pre-requisite(s):				

Course Perspective This course teaches students how to craft and amplify their personal brand using AI-enabled tools for design, content creation, video production, and audience engagement. From defining a personal niche to producing brand-aligned posts, videos, and visuals, students will leverage tools like ChatGPT, Canva AI, Copy.ai, Notion AI, Lumen5, and LinkedIn analytics to showcase their skills, story, and unique voice across platforms.

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

COs	**Statements**
CO1	Defining their personal brand identity and positioning aligned to their career goals.
CO2	Using AI tools to generate high-quality content, visuals, and videos.
CO3	Building a consistent digital presence across platforms like LinkedIn, YouTube, and personal websites.
CO4	Analyzing and improve their online engagement through analytics and campaign tracking.

Course Outline:

Unit Number: 1	**Discovering Your Personal Brand Identity (8 hours)**	**No.of hours: 8**
Content: What is a personal brand? Understanding positioning, values, niche, and voice.		

Self-assessment: passions, strengths, career vision Crafting brand statements: "Who am I?", "What do I stand for?" Tools: ChatGPT for mission statement, audience definition Hands-on: Write your personal brand pitch + tagline using AI prompts		
Unit Number: 2	**Content Creation with AI for Branding (8 hours)**	**No. of hours: 8**
Content: Copywriting with AI: Headlines, bios, posts (using **Copy.ai**, **Jasper**, or **ChatGPT**) Generating content ideas using prompt engineering Automating newsletters/blogs: Using **Notion AI** and **Substack** Hands-on: Create 3 social media posts and a blog article using AI content generators		
Unit Number: 3	**Visual & Video Branding with AI Tools (7 hours)**	**No. of hours: 8**
Content: Designing logos, templates, and visual identity using **Canva AI**, **Looka**, **Fotor AI** AI video creators: **Lumen5**, **Pictory**, **Animoto** Hands-on: Produce a 60-second brand intro video using text-to-video AI Activity: Build a simple portfolio site using **Carrd** or **Notion**		
Unit Number: 4	**Building Influence & Engagement with Analytics (7 hours)**	**No. of hours: 7**
Content: LinkedIn strategy: Profile optimization, content calendar, and analytics Tracking performance: Google Analytics basics, LinkedIn dashboard Creating engagement campaigns: 5-day visibility challenge using AI-generated content Capstone: Launch a personal campaign (e.g., "Build in Public" series or 30-day post challenge)		

Learning Experiences

Inside Classroom Learning

- Practice guided mindfulness exercises such as breathing techniques, body scans, and focused attention meditation.
- Explore the components of Emotional Intelligence (EQ)—self-awareness, self-regulation, motivation, empathy, and social skills—through interactive sessions.
- Analyze real-life situations or case studies involving emotional challenges and discuss mindful responses.
- Participate in group activities and role-plays to improve emotional expression and active listening.
- Maintain a reflective journal to log daily emotional triggers and mindful responses.

Outside Classroom Learning

- Implement a daily mindfulness practice (5–10 minutes) for 2 weeks and record observations on mental clarity and stress.
- Apply emotional intelligence techniques in real interactions (e.g., conflict resolution, peer communication) and reflect on the outcome.
- Interview 2 individuals (mentor, parent, or peer) about how they manage emotions and stress in life or work.
- Design and conduct a mini-awareness campaign (poster, video, Instagram reel) on "Why Mindfulness Matters."
- Create a personal EQ Growth Plan with short- and long-term goals for improving emotional resilience.

Textbooks & Resources:

Labrecque, L. I., Markos, E., & Milne, G. R. (2011). *Online Personal Branding: Processes, Challenges, and Implications*. Journal of Interactive Marketing, 25(1), 37–50.

Semester: VI

Specialization Course-V

Usability Testing & Accessibility

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Usability Testing & Accessibility**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETUXUT602	3-0-2	4	45
Type of Course:	DSE			
Pre-requisite(s), if any: Understanding of Design Basics, Graphic Design Tools such as Adobe Photoshop, Illustrator, Figma, Sketch, or Adobe XD.				

Course Perspective: This course equips students with the knowledge and skills to evaluate and improve the usability and accessibility of digital products. It covers a range of usability testing methods including think-aloud, A/B testing, remote and in-person testing, as well as the standards and tools used to evaluate accessibility based on WCAG 2.1. Students will gain hands-on experience in planning, executing, and analyzing usability tests and conducting accessibility audits using tools such as Figma, Axe, Wave, and screen readers. The course enables students to build inclusive, user-friendly designs for all users, including those with disabilities.

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

COs	Statements
CO 1	Explaining key principles of usability and accessibility in digital interfaces.
CO 2	Conducting usability tests using appropriate methods and analyze user feedback.
CO 3	Evaluating digital products against WCAG 2.1 guidelines and other accessibility standards.
CO 4	Using tools like Axe, Wave, and screen readers for accessibility testing.

CO 5	Evaluate usability and accessibility testing results to recommend strategic design enhancements for improved user experience.

Course Outline:

Unit Number: 1	**Title: Principles of Usability & Accessibility**	**No. of hours: 12**

Principles of Usability & Accessibility

Objective: Understand foundational principles and importance of usability and accessibility.

Contents:

- Usability definitions, dimensions (effectiveness, efficiency, satisfaction)
- Overview of usability heuristics (Nielsen's 10 principles)
- What is accessibility? Legal and ethical importance
- Introduction to WCAG 2.1 guidelines (Perceivable, Operable, Understandable, Robust)
- Types of impairments: visual, auditory, motor, cognitive
- Assistive technologies: screen readers, keyboard navigation, voice control

Hands-On:

- Critique 2 websites using usability heuristics
- Accessibility walkthrough using keyboard-only navigation and a screen reader

Unit Number: 2	**Title: Usability Testing Methods & Planning**	**No. of hours: 12**

Objective: Learn how to plan, design, and prepare usability tests.

Contents:

- Types of usability tests: formative vs summative
- Think-aloud method, remote vs in-person testing, moderated vs unmoderated
- Defining test goals, participant profiles, tasks, and success criteria
- Writing usability testing scripts

- Ethics in usability testing: consent, privacy, bias reduction
- Sample size and test frequency

Hands-On:

- Draft a usability test plan for a selected app or website
- Write 3 task-based usability scenarios
- Run a dry-run test with a peer and document test logistics

Unit Number: 3	**Title: Conducting Usability Tests & Analyzing Results**	**No. of hours: 10**

Objective: Conduct effective tests and translate findings into design insights. Contents:

- Test facilitation tips, note-taking, observation techniques
- Capturing behavior, pain points, time-on-task, success/failure
- Tools: Lookback, Maze, Useberry, and manual testing
- Creating usability scorecards
- Synthesizing feedback using affinity mapping
- Reporting test results: visuals, severity rankings, actionable insights

Hands-On:

- Conduct a usability test with at least 2 participants
- Log and analyze findings using affinity mapping and test metrics
- Create a usability report with prioritized recommendations

Unit Number: 4	**Title: Accessibility Testing & Inclusive Design**	**No. of hours: 11**

Objective: Evaluate and redesign digital interfaces to be inclusive and accessible. Contents:

- Deep dive into WCAG 2.1 success criteria
- Introduction to WAI-ARIA and semantic HTML
- Accessibility tools: Axe, WAVE, Lighthouse, NVDA screen reader

- Color contrast testing and accessible typography
- Inclusive design: design for edge cases and diverse users
- Retrofitting accessibility in existing products

Hands-On:

- Conduct an accessibility audit on a university portal
- Use WAVE and Axe to identify accessibility violations
- Redesign 2–3 screens in Figma incorporating accessibility improvements

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	Lab Assignment 1 (Unit 1) Title: Heuristic Evaluation and Accessibility Diagnosis of a Government Website Sub-Objectives: 1. Apply Nielsen's 10 heuristics to identify usability issues 2. Evaluate color contrast, keyboard navigation, and screen reader compatibility 3. Summarize findings and propose 3 quick fixes	CO 1
2	Lab Assignment 2 (Unit 2) Title: Designing a Remote Usability Test for a Health App Sub-Objectives: • Draft a usability test plan with 3 task-based scenarios • Prepare test script, consent form, and metrics to be measured	CO 2

	• Pilot the test with a peer using remote screen sharing tools	
3	Lab Assignment 3 (Unit 3) Title: Conducting and Analyzing Usability Tests for an Educational Portal Sub-Objectives: 1. Recruit and conduct test sessions with at least 2 participants 2. Collect metrics like time-on-task, task completion, and errors 3. Synthesize data and prepare a usability insights report	CO 3
4	Lab Assignment 4 (Unit 4) Title: Accessibility Audit and Redesign of an E-commerce Site Sub-Objectives: 1. Perform an accessibility audit using Axe and WAVE 2. Identify WCAG 2.1 violations and annotate findings 3. Redesign two selected screens in Figma using accessible design principles	CO 4
5	Capstone Lab Assignment (All Units) Title: Designing and Testing an Inclusive City Services Portal Sub-Objectives: 1. Conduct a complete usability test and accessibility audit of a city services site 2. Collect feedback from users with diverse needs (including at least one with a disability)	CO 5

	3. Redesign 4–5 key screens incorporating usability and accessibility feedback 4. Prepare and present a final report including a test plan, findings, before/after designs, and justification	

Textbook:

Katz-Haas, R. (2022). UX Research and Usability Testing: A Hands-On Guide for Beginners. Rosenfeld Media.

Learning Experiences

Inside the Classroom

- **Interactive Lectures:**
  Key topics like usability heuristics, WCAG 2.1, ARIA, and inclusive design taught through real examples and tool walkthroughs (e.g., Figma, Axe).
- **Live Demos & Tools:**
  In-class demonstrations of usability testing tools (Maze, Lookback) and accessibility tools (WAVE, NVDA, Lighthouse).
- **Group Work & Case Studies:**
  Analyze real websites, conduct test planning, and suggest improvements through collaborative activities.
- **Integrated Lab Sessions:**
  Apply concepts hands-on—conduct usability tests, run audits, and redesign interfaces during lab hours.

Outside the Classroom

- **Lab Assignments:** Weekly tasks include usability test plans, accessibility audits, and screen redesigns using Figma.
- **Capstone Project:** End-to-end testing and redesign of a city portal, including feedback from diverse users.

- **Team Collaboration:** Peer testing, group analysis, and shared reporting to foster real-world UX teamwork.
- **Self-Learning:** Explore resources like WCAG guides, Figma plugins, and usability tutorials for deeper understanding.

Specialization Course-VI

Advanced UI Development & Handoff

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: **Advanced UI Development & Handoff**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETUXUT602	3-0-2	4	45
Type of Course:	DSE			
Pre-requisite(s), if any: Understanding of Design Basics, HTML, CSS, Javascript				

Course Perspective: This course equips students with advanced front-end development skills and professional handoff practices crucial to real-world UI/UX workflows. It bridges the gap between design and development through responsive UI building, component libraries, design system implementation, code-based prototyping, and developer collaboration tools. Students will learn to convert Figma designs into functional front-end code using HTML, CSS (Flexbox/Grid), JavaScript, and frameworks like React, along with using platforms such as Zeplin, Figma Inspect, and Storybook for smooth design-to-development transition.

By the end of the course, students will be able to work in front-end teams, implement responsive UIs, build design systems, and communicate clearly with developers during the handoff phase.

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

COs	Statements
CO 1	Translating design mockups into responsive and accessible front-end code.
CO 2	Building reusable UI components using modern HTML, CSS, and JavaScript frameworks.
CO 3	Applying version control, design tokens, and documentation in scalable design systems.

CO 4	Applying developer collaboration tools (Figma Inspect, Zeplin, Storybook) for effective handoff.
CO 5	Performing quality checks on UI implementation and ensure design consistency.

Course Outline:

Unit Number: 1	**Title: From Design to Code – UI Foundations**	**No. of hours: 12**

Objective: Learn how to interpret design artifacts and build semantic, accessible, responsive UI.

Contents:

- Inspecting UI from Figma/Zeplin: extracting assets, tokens, spacing
- HTML5 semantic elements: structure for accessibility
- CSS Fundamentals: Flexbox, Grid, box model, spacing, typography
- Media queries for responsive breakpoints
- Introduction to SCSS/SASS for component styling
- Basic accessibility coding practices (WCAG principles)

Hands-On:

- Convert a Figma mobile screen into a responsive HTML/CSS page
- Apply accessibility tags and ARIA roles

Unit Number: 2	**Title: UI Componentization & State Management**	**No. of hours: 12**

Objective: Develop modular, scalable UI components and manage interaction logic.

Contents:

- Component-based architecture: reusability, scalability
- Intro to JavaScript interactivity (DOM manipulation, events)
- UI state handling basics (form validation, toggle menus)
- Component design patterns using HTML + JS

- Overview of React (JSX, props, component hierarchy)
- Using CSS-in-JS or Styled Components (conceptual intro)

Hands-On:

- Build 3 reusable components (e.g., Button, Modal, Form Field) using vanilla JS or React
- Document component usage in Storybook

Unit Number: 3	**Title: Design Systems & Front-End Workflows**	**No. of hours: 10**

Objective: Integrate design systems and collaborate in agile environments.

Contents:

- Introduction to Design Systems: tokens, grids, typography scales
- Figma Libraries vs Code Components (design-development sync)
- Design handoff process using Figma Inspect, Zeplin
- Intro to Git/GitHub for UI projects (version control)
- Developer collaboration workflow in Agile (branching, commits, PRs)
- CI/CD basics for front-end deployment

Hands-On:

- Create a mini design system with 4–5 tokens/components in Figma & Code
- Handoff to another student acting as a developer

Unit Number: 4	**Title: UI QA, Performance, & Accessibility Validation**	**No. of hours: 11**

Objective: Ensure that built interfaces are accessible, performant, and consistent.

Contents:

- QA testing for UI: browser testing, resolution checks
- Visual regression testing (tools: Percy, Storybook)
- Lighthouse testing: performance, accessibility, SEO
- Keyboard navigation and focus order

- Color contrast validation, alt text, aria-labels
- Documentation practices for handoff

Hands-On:

- Run accessibility audits using Lighthouse and WAVE
- Fix issues and prepare QA handoff documentation

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	Lab Assignment 1 (Unit 1) Title: "Responsive Coding of a Campus Events Page" Sub-Objectives: • Use Figma design to extract layout & spacing specs • Convert mockup to responsive HTML5/CSS3 page • Apply semantic tags for headers, navigation, content areas • Implement mobile and desktop views using media queries • Perform basic accessibility enhancements	CO 1
2	Lab Assignment 2 (Unit 2) Title: Building a Visual Identity Using Typography and Color Sub-Objectives: Title: "Reusable UI Component Library for Student Dashboard" Sub-Objectives: • Build components: Sidebar Menu, Button, Modal, Dropdown	CO 2

	• Add interactivity using vanilla JS or React (toggle, hover, click) • Structure components for reuse across pages • Document component usage in Storybook or Notion • Test components in different screen sizes	
3	Lab Assignment 3 (Unit 3) Title: "Design System Handoff for a College Registration System" Sub-Objectives: • Create Figma design tokens: typography, spacing, color • Build reusable UI kit (form fields, cards, buttons) in code • Version components in GitHub repo • Perform design-to-code handoff using Zeplin or Figma Inspect • Simulate developer-deigner feedback cycle	CO 3
4	Lab Assignment 4 (Unit 4) Title: "Accessibility & Performance QA for NGO Website" Sub-Objectives: • Run Lighthouse test for page performance & accessibility • Perform keyboard navigation test and WCAG audit • Fix contrast, aria-label, tab order issues • Write QA report on mismatches between design and code • Prepare final QA checklist for handoff	CO 4
5	Capstone Lab Assignment (All Units)	CO 5

	Title: "End-to-End Responsive UI Build & Developer Handoff for a University Portal" Sub-Objectives: • Start from Figma design of a university portal (Home, Login, Course, Events) • Build entire interface using modular components (HTML, CSS, JS/React) • Apply accessibility, responsiveness, and performance principles • Conduct handoff via Figma Inspect + GitHub repo + README documentation • Present the final product with design-code consistency report	

Textbook:

Smashing Magazine. (2021). Smashing Book 6: New Frontiers in Web Design. Smashing Media AG.

Learning Experiences

Inside Classroom Learning

- **Interactive Lectures**:
  Explain responsive UI, semantic HTML, Flexbox/Grid, React basics, design systems, accessibility, and QA testing using real-world UI mockups.
- **Live Demos & Code Walkthroughs**:
  Convert Figma screens to HTML/CSS; demonstrate reusable components and state handling in vanilla JS/React; simulate handoff using Zeplin and Storybook.

- **Concept Visualizations**:
  Use design tokens, UI libraries, and accessibility tools (e.g., Lighthouse, WAVE) to connect design theory with frontend coding practices.
- **Problem-Solving & Case-Based Discussion**:
  Analyze poor UI code vs. good component-based code. Class debates on accessibility violations or performance issues.
- **Collaborative Peer Work**:
  In-class group exercises on handoff, version control practices, or responsive bug-fixing.

Outside Classroom Learning

- **Hands-On Projects & Lab Work**:
  Create full-page responsive UIs, build reusable UI kits, integrate accessibility features, and simulate handoff processes.
  Capstone Assignment:
  Work on a university portal UI from scratch, implementing design-to-code transition, accessibility, QA, and documentation in teams.
- **Self-Paced Exploration**:
  Recommended tools: Figma Inspect, Zeplin, Storybook, GitHub, Lighthouse.
  Courses: FreeCodeCamp, Figma tutorials, React Docs.
  Textbook: Smashing Book 6: New Frontiers in Web Design.
- **Peer Feedback & Reviews**:
  Exchange handoff files with classmates, review implementation quality, and suggest improvements for responsiveness or accessibility.

Computer Networks

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: Computer Networks	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCCN603	3-0-2	4	60
Type of Course:	Major			
Pre-requisite(s), if any: Basic understanding of operating systems, programming fundamentals (C/Python), and binary number systems.				

Course Perspective: This course provides a comprehensive introduction to the principles, design, and implementation of computer networks. It covers the OSI and TCP/IP models, data transmission, topologies, protocols, addressing, and network services. The course integrates hands-on labs involving simulation, protocol analysis, subnetting, routing, and basic network programming to enhance understanding and practical skills in configuring, maintaining, and troubleshooting networks.

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

COs	Statements
CO1	**Explaining** fundamental network concepts, architecture models (OSI, TCP/IP), and network topologies.
CO2	**Describing** and **applying** data link layer concepts including error control, flow control, and multiple access protocols.
CO3	**Analyzing** and configure network layer addressing (IPv4/IPv6) and routing protocols.
CO4	**Explaining** and analyze transport and application layer protocols and use network tools to troubleshoot and analyze performance.

Course Outline:

Unit Number: 1	**Foundations of Computer Networks & Data Communication**	**No. of hours: 15**

Content:

- Introduction to computer networks: LAN, WAN, MAN, PAN
- Network topology: Bus, Star, Ring, Mesh, Hybrid
- OSI and TCP/IP models
- Data transmission concepts: Analog vs. digital, bandwidth, throughput, latency, jitter
- Network devices: Hubs, switches, routers, bridges, gateways
- Transmission media: Twisted pair, coaxial, fiber optic, wireless

Real-World Use Case:

- Designing a basic campus network topology, identifying network components in a typical office, Understanding data transmission rates and delays

Unit Number: 2	**Data Link Layer & Network Layer Fundamentals**	**No. of hours: 15**

Content:

- Data link layer functions: Framing, error detection and correction (parity, checksum, CRC), flow control
- Multiple access protocols: ALOHA, CSMA, CSMA/CD, CSMA/CA
- Ethernet (IEEE 802.3): Frame format
- Wireless LANs (IEEE 802.11): Architecture and MAC
- Logical addressing: IPv4 (classful, classless, subnetting, CIDR), IPv6

Real-World Use Cases:

- Analyzing Ethernet frames using packet sniffers, troubleshooting network collisions, Creating subnets for organizational departments.

Unit Number: 3	Network Layer Routing & Transport Layer	No. of hours: 15

Content:

- Routing concepts: Forwarding vs. routing
- Routing algorithms: Distance vector, link state
- Routing protocols: RIP, OSPF (basics), BGP (basics)
- Transport layer services: Process-to-process delivery, multiplexing/ demultiplexing
- UDP: Header, features, applications
- TCP: Header, connection establishment (3-way handshake), flow control, congestion control

Real-World Use Cases:

- Analyzing routing tables and path of packets, Understanding TCP vs. UDP in different applications, Observing TCP connection setup and teardown using tools.

Unit Number: 4	Application Layer & Network Utilities	No. of hours: 15

Content:

- DNS: Structure, name resolution
- HTTP: Methods, persistent vs. non-persistent, cookies, caching
- FTP and SMTP protocols
- DHCP and IP configuration
- Basic network security: CIA triad, firewall basics
- Network tools: Ping, traceroute, netstat, ipconfig/ifconfig

Real-World Use Cases:

- Understanding webpage retrieval using DNS and HTTP, Configuring DHCP server in a simulated environment, Troubleshooting using ping, traceroute, and netstat, Observing DNS queries and resolving errors.

Classroom Learning Experience

Inside Classroom Learning:

1. **Layered Architecture Mapping:** OSI and TCP/IP models are introduced using layered visualizations, interactive whiteboarding, and device-oriented case studies. Students trace packet flow across layers using real and simulated topologies.
2. **Protocol and Frame Structure Analysis:** Ethernet, IP, TCP, and UDP headers are examined through packet dissection activities using Wireshark. Students analyze fields, flags, and frame structures from actual network captures.
3. **Subnetting and Addressing Sessions:** IP addressing concepts (IPv4/IPv6, CIDR, subnetting) are explored through guided calculation exercises and IP planning simulations for small networks or organizational departments.
4. **Routing Algorithms and Transport Protocol Labs:** Routing and transport layer protocols are explored with algorithm walkthroughs and demo simulations. TCP connection establishment, congestion control, and UDP communication are tested using tools and network emulators.

Outside Classroom Learning Experience

1. **Packet Capture & Protocol Inspection Labs:** Students use Wireshark to inspect and document different protocol behaviors including Ethernet, ARP, IP, TCP, and HTTP. Tasks include identifying handshakes, retransmissions, and malformed packets.
2. **Network Design Projects:** Teams design and simulate multi-layered network topologies using tools like Cisco Packet Tracer or GNS3. These projects include logical addressing, subnetting, router/switch configuration, and routing setup.
3. **Troubleshooting Scenarios with Utilities:** Practice scenarios involve identifying and resolving connectivity issues using command-line tools such as ping, traceroute, ipconfig, and netstat. Each case is documented with cause-effect-outcome analysis.

4. **Application Layer Simulations:** Students simulate DNS resolution, HTTP requests/responses, and email transmission using packet sniffers or browser-based tools. Activities include analyzing HTTP methods, cookies, caching, and DNS errors.
5. **Security Configuration Exercises:** Labs involve basic firewall rule setup, DHCP misconfiguration identification, and discussion on CIA triad through use-case simulations and misconfiguration impact tracing.

Text and Reference Book

1. Kurose, J. F., & Ross, K. W. – *Computer Networking: A Top-Down Approach*, Pearson.
2. Forouzan, B. A. – *Data Communications and Networking*, McGraw-Hill.
3. Tanenbaum, A. S., & Wetherall, D. – *Computer Networks*, Pearson Education.

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab Task 1: OSI/TCP-IP Model Exploration & Network Topology Simulation (Aligned with Unit 1 – Foundations of Computer Networks & Data Communication)** **Real-World Scenario:** You are hired by a small company to design a basic network for a 3-floor office. Your task is to visualize the layout using different topologies and identify devices needed for each layer of communication. **Objectives:** • Simulate star, mesh, and hybrid topologies using Cisco Packet Tracer or similar tools • Map devices to OSI/TCP/IP layers and analyze their roles	CO1

	• Analyze network performance: latency, throughput, jitter via simulation • Document and present topology design and justification **Tools:** Cisco Packet Tracer / GNS3, Wireshark (basic), Draw.io (for diagrams)	
2	**Lab Task 1: OSI/TCP-IP Model Exploration & Network Topology Simulation (Aligned with Unit 1 – Foundations of Computer Networks & Data Communication)** **Real-World Scenario:** You are hired by a small company to design a basic network for a 3-floor office. Your task is to visualize the layout using different topologies and identify devices needed for each layer of communication. **Objectives:** • Simulate star, mesh, and hybrid topologies using Cisco Packet Tracer or similar tools • Map devices to OSI/TCP/IP layers and analyze their roles • Analyze network performance: latency, throughput, jitter via simulation • Document and present topology design and justification **Tools:** Cisco Packet Tracer / GNS3, Wireshark (basic), Draw.io (for diagrams)	CO2
3	**Lab Task 3: Routing Tables, TCP/UDP Behavior & Packet Analysis (Aligned with Unit 3 – Routing & Transport Layer)** **Real-World Scenario:** A logistics company faces data delay and unreliable packet delivery between remote branches. You must compare TCP and UDP use cases and simulate routing using RIP and OSPF. **Objectives:** • Configure static and dynamic routing using RIP/OSPF in Packet Tracer	CO3, CO4

	• Analyze TCP handshake and UDP headers using Wireshark • Compare connection establishment and delivery guarantees of TCP vs. UDP • Simulate a packet loss scenario and observe retransmission behavior **Tools:** Packet Tracer / GNS3, Wireshark, IP/route command utilities	
4	**Lab Task 4: DNS, HTTP, DHCP & Network Utility Tools (Aligned with Unit 4 – Application Layer & Network Tools)** **Real-World Scenario:** Your client's internal website is failing to load. You're asked to troubleshoot the issue using DNS resolution checks, HTTP response codes, and verify dynamic IP allocation. **Objectives:** • Simulate DNS and HTTP interactions (e.g., HTTP GET, DNS A record) • Use tools: ping, traceroute, netstat, ipconfig/ifconfig for network diagnostics • Configure a DHCP server and verify IP leasing process • Analyze HTTP headers and cookies using browser dev tools or Wireshark **Tools:** Wireshark, Browser Dev Tools, nslookup, ping, traceroute, DHCP server (in simulator)	CO4
5	**Capstone Project: Design, Simulate, and Troubleshoot a Departmental Network** **Real-World Scenario:** You are hired by a university IT team to build a network for departments like Admin, Library, and Labs. The network must support dynamic IP, secure HTTP access, proper routing, and subnetting across departments.	CO4

	Objectives: • Design the logical and IP-level architecture with subnetting • Simulate routers, switches, DHCP, and DNS configuration • Implement routing protocols and inter-VLAN communication • Use network tools to verify configuration and troubleshoot latency issues **Tools:** Cisco Packet Tracer / GNS3, Wireshark, DNS/DHCP simulation, net-tools	

Comprehensive Placement Preparation

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: Comprehensive Placement Preparation	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	AEC	2-0-0	2	30
Type of Course:	AEC			
Pre-requisite(s), if any:				

Course Perspective: The Comprehensive Placement Preparation Program is strategically designed to foster employability by equipping students with essential skills in aptitude, communication, personal branding, and professional behavior. Rooted in industry-specific demands and global expectations, the program integrates mock placement simulations, digital portfolio development, and structured evaluation to bridge the gap between academic learning and professional readiness.

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

COs	**Statements**
CO1	**Developing** a digital professional identity through optimized LinkedIn profiles, customized resumes, and tailored cover letters, showcasing readiness for industry and entrepreneurship.
CO2	**Applying** quantitative, analytical, and verbal reasoning skills to solve placement-oriented problems, enhancing employability through structured problem-solving approaches.
CO3	**Demonstrating** effective communication and writing skills, including professional email drafting, paragraph structuring, and vocabulary enhancement, aligning with workplace expectations.
CO4	**Displaying** confidence, ethical behavior, and professional etiquette during group discussions, mock interviews, and public interactions, reflecting

	leadership and responsible citizenship.
CO5	**Engaging** in experiential and outcomes-based learning through practical simulations and peer-reviewed exercises that promote critical thinking, self-assessment, and continuous improvement.

Course Outline:

Unit Number: 1	**Professional Branding & Profiling**	**No. of hours: 8**
Content: • **Session 1:** Digital Profile Workshop & Photoshoot • **Session 4:** Resume & Cover Letter Writing Workshop • **Session 6:** Resume & Cover Letter Submission & Feedback • **Session 14:** Mock Interview + Video Resume Workshop • **Session 15:** Mock PI Round + Student Video Resume Showcase		
Unit Number: 2	**Quantitative & Analytical Reasoning Practice**	**No. of hours: 8**
Content: • **Session 2:** Ratio, Proportion, Averages, Percentages & Shortcuts • **Session 5:** Number & Alphabet Series, Divisibility & Patterns • **Session 8:** Time, Work, Time-Speed-Distance & Shortcuts • **Session 11:** Remainders, Unit Digits & Last Two Digits • **Session 12:** Profit, Loss, S.I., C.I., Discounts & Shortcuts		
Unit Number: 3	**Communication Mastery & Etiquette**	**No. of hours: 6**
Content: • **Session 3**: Vocabulary Quest – Word Power Enhancement • **Session 9**: Email Etiquette + Paragraph Writing Workshop		

• **Session 10**: Professional Etiquette + Body Language Workshop.		
Unit Number: 4	**Placement Simulation, Engagement & Evaluation**	**No. of hours: 8**
Content: • **Session 7:** Company-Specific Test-1 + Discussion • **Session 13:** Group Discussion Workshop + Mock GD Rounds • **Session 14:** Mock Interview + Video Resume Workshop • **Session 15:** Mock PI Round + Student Video Resume Showcase		

Classroom Learning Experience

Inside Classroom Learning:

1. **Professional Profile Development Workshops:** Students participate in structured sessions focused on building LinkedIn profiles, writing resumes and cover letters, and refining digital branding assets. Live feedback and peer assessments help students align their profiles with industry standards.
2. **Aptitude & Analytical Problem Solving Sessions:** Quantitative and analytical reasoning concepts are practiced through problem-solving drills and concept-specific worksheets. Shortcuts and mental math strategies are reinforced using timed quizzes and mock test simulations.
3. **Communication & Etiquette Labs:** Interactive workshops focus on vocabulary building, professional email writing, paragraph structuring, and body language. Etiquette sessions include role-playing to simulate workplace communication scenarios.
4. **Mock Interview and GD Practice Rounds:** Students engage in classroom-based simulations of HR and technical interviews, group discussions, and pitch presentations. Structured rubrics and instructor feedback are used to evaluate verbal expression, content structure, and behavioral cues.

Outside Classroom Learning Experience

1. **Digital Portfolio & Resume Building Assignments:** Students independently update and publish their LinkedIn profiles, develop customized resumes for different roles, and record professional video resumes. Submissions are peer-reviewed and refined based on rubrics.
2. **Practice Problem Sets & Sectional Tests:** Learners complete placement-style sectional tests on topics like time-speed-distance, profit & loss, number series, and verbal reasoning on platforms like HackerRank, AMCAT, or CoCubes.
3. **Peer Feedback and Reflection Exercises:** Students conduct mock interviews and GDs in small groups, providing each other with constructive feedback based on evaluation criteria. Sessions are video-recorded for self-assessment and reflection.
4. **Company-Specific Test Simulation:** External company test papers and coding problems are simulated in timed conditions to familiarize students with real assessment formats. After-test discussions focus on solution strategies and mistake patterns.
5. **Public Showcases and Confidence Building Activities:** Students showcase video resumes and portfolios in class exhibitions. Participating in formal presentations and feedback panels strengthens articulation, presence, and professional readiness.
6. **Model Evaluation Challenges:** Tune and validate models using cross-validation, grid search, and pipelines.

Competitive Coding –IV

Programme Name:	**B.Tech CSE (Specialization in UI & UX)**			
Course Name: Competitive Coding -II	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	SEC	2-0-0	2	30
Type of Course:	SEC			
Pre-requisite(s), if any: Competitive programming III, Fundamentals of programming & data structure				

Course Perspective: This course focuses on strengthening students' competitive programming skills by introducing advanced data structures and algorithms such as Tries, Heaps, Segment Trees, and Dynamic Programming. It prepares learners to solve complex coding problems efficiently, enabling success in technical interviews and national-level coding contests.

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

COs	**Statements**
CO1	**Applying** advanced string algorithms and data structures, such as Trie and Huffman Coding, to solve complex problems.
CO2	**Analyzing** and implement efficient solutions to dynamic programming problems using memoization and tabulation approaches.
CO3	**Evaluating** and **applying** tree and segment tree operations to solve traversal, range queries, and interval-based problems.

Course Outline:

Session:1	Trie	No. of hours: 2
Content summary: what is trie DS, use of trie, hashmap vs trie, implementation (representation, insert node, search node)		
Session: 2	Trie-II	No. of hours: 2
Content Summary: delete node, application of trie, count word in trie, word break		
Session: 3	Huffman coding	No. of hours: 2
Content Summary: huffman coding algorithm, decompression in huffman coding		

Session: 4	Binary Search-II	No. of hours: 1
Content Summary: Search in rotated sorted array, Search in rotated sorted array II, aggressive cows		
Session: 5	Binary Tree Introduction	No. of hours: 1
Content Summary: Introduction of Tree, type of tree, implementation of tree.		
Session: 6	Binary Tree Traversal	No. of hours: 1
Content Summary: Tree Traversal, preorder traversal, inorder traversal, postorder traversal, level order traversal(Morris traversal).		
Session: 7	Binary Tree-III.	No. of hours: 1
Content Summary: Height of the tree, same tree, symmetric tree,		
Session: 8	Binary Tree-IV.	No. of hours: 1
Content Summary: diameter of tree, path sum, print left/right view of Binary tree.		
Session : 9	Binary Search Tree.	No. of hours: 2
Content Summary: Implementation of BST, check valid BST		
Session : 10	Binary Search-II	No. of hours: 1
Content Summary: convert sorted array to BST, Delete node in BST, lowest common ancestor		
Session : 11	Hashmap Introduction.	No. of hours: 2
Content Summary: HashMap Implementation (operations put, get, containsKey,		

KeySet)		
Session: 12	HashMap-II.	No. of hours: 2
Content Summary: Two Sum, highest frequency character, missing number		
Session:13	HashMap-III.	No. of hours: 1
Content Summary: intersection of two arrays, set matrix zeros, valid anagram		
Session: 14	hashmap/Sliding window-technique Algorithm	No. of hours:2
Content Summary: longest consecutive sequence, longest substring without repeating character, bulls and cows		
Session: 15	hashmap/Sliding window-technique Algorithm	No. of hours: 2
Content Summary: largest subarray with 0 sum, count of zero sum subarray, length of largest subarray with contiguous element		
Session: 16	Priority Queue	No. of hours: 1
Content Summary: Implementation of Priority queue, min and max Heap		
Session: 17	priority Queue-II	No. of hours: 1
Content Summary: Inplace heap sort, kth largest element, kth smallest element		
Session: 18	priority Queue-III	No. of hours: 1
Content Summary: check max heap, top k frequent element, sliding window maximum		
Session: 19	Sum up Binary tree and Binary search Tree	No. of hours: 2
Content Summary: sum of leaves, top view, bottom view,		
Session: 20	Sum up Hashmap / Sliding window technique.	No. of hours: 2
Content Summary: find all anagram in string, isomorphic string		

Reference Books:

- "Introduction to Algorithms" by Cormen, Leiserson, Rivest, and Stein
- "Cracking the Coding Interview" by Gayle Laakmann McDowell
- "Elements of Programming Interviews" by Adnan Aziz, Tsung-Hsien Lee, and Amit Prakash

Value Added Course (VAC)

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name 1: Product Deployment & Growth Hacking	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- III			
Pre-requisite(s):				

Course Perspective. This course dives deep into the practical aspects of launching and scaling digital products. Students will learn how to deploy MVPs, optimize performance across platforms, collect actionable analytics, and apply growth-hacking techniques to drive traction and user retention. Real-world projects and tools will be leveraged to simulate startup-like growth environments.

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

COs	**Statements**
CO1	**Deploying** MVPs to live cloud or web environments using modern tools.
CO2	**Setting** up analytics and monitor product metrics in real time
CO3	**Implementing** SEO, A/B testing, referral loops, and viral mechanics
CO4	**Applying** rapid experimentation to optimize growth strategies
CO5	**Creating** and run sustainable user acquisition campaigns using minimal budgets

Course Outline:

Unit Number: 1	**Product Deployment & Infrastructure Basics**	**No.of hours: 8**
Content: • Deploy MVPs using platforms like Vercel, Netlify, Firebase • Connect custom domains and enable HTTPS/SSL • Use GitHub for CI/CD-style version control		

- Integrate backend services using APIs or Firebase functions
- Monitor uptime using services like UptimeRobot
- Use Docker basics to containerize the app (optional, bonus)

Unit Number: 2	Metrics, Analytics & Conversion Optimization	No. of hours: 8

Content:

- Install GA-4, Mixpanel, or PostHog for user behavior tracking
- Define and track funnel metrics: acquisition, activation, retention
- Identify North Star Metric (NSM) and KPIs
- Conduct basic A/B testing using tools like Google Optimize
- Create dashboards for data-driven decision-making (e.g., Google Data Studio)

Unit Number: 3	Growth Hacking Playbook	No. of hours: 8

Content:

- Hook model: Trigger → Action → Reward → Investment
- Create viral loops: referral programs, waitlists, gamification
- Run low-budget user acquisition: Reddit, Product Hunt, Discord
- Apply SEO fundamentals: keyword research, meta, backlinks
- Build email onboarding & retention flow with tools like Mailchimp
- Optimize CTAs & landing pages with copywriting best practices

Unit Number: 4	Growth Experiments & Scaling	No. of hours: 6

Content:

- Build a growth experiment backlog using ICE scoring
- Design and run experiments: Hypothesis → Execution → Learnings
- Use tools like Notion or Airtable for experiment tracking
- Run surveys and collect feedback via Typeform or Google Forms
- Case studies on blitzscaling vs sustainable scaling
- Understand growth loops vs funnels.

Learning Experiences

Inside Classroom Learning*:*

1. Independent landing page deployments and configuration of real-time metrics.
2. Growth experiments through A/B testing, SEO adjustments, and email campaign launches.
3. Group projects for designing low-budget viral campaigns or onboarding flows.
4. Portfolio submissions of product deployment pipelines and growth reports.

Outside Classroom Learning

1. Independent landing page deployments and configuration of real-time metrics.
2. Growth experiments through A/B testing, SEO adjustments, and email campaign launches.
3. Group projects for designing low-budget viral campaigns or onboarding flows.
4. Portfolio submissions of product deployment pipelines and growth reports.

Textbooks and Reference Books:

1. Ellis, S., & Brown, M. (2017). *Hacking growth: How today's fastest-growing companies drive breakout success*. Crown Business.
2. Bang, J. (2021). *Fullstack D3 and Data Visualization: Build beautiful data visualizations with D3*. Newline Media.

Case Studies to Demonstrate

1. **Dropbox** – Referral loop strategy with exponential growth
2. **Slack** – Bottom-up product-led growth and onboarding
3. **AirBnB** – SEO hack using Craigslist
4. **Duolingo** – Growth loop through gamified UX and notifications
5. **Notion** – Waitlist and community-based growth flywheel

Lab/Hands-On Tasks

- Deploy a landing page with custom domain + analytics
- Create a referral-based share mechanism for the MVP
- Run a 3-day A/B test with headline/copy/image changes
- Design and launch an email campaign with basic automation
- Run growth audit and suggest 3 experiments based on MVP metrics

Storytelling, Fundraising & Investor Pitch Crafting

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name 2: Storytelling, Fundraising & Investor Pitch Crafting	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- III			
Pre-requisite(s):				

Course Perspective. This course empowers aspiring founders with the art and science of startup storytelling and fundraising. It blends narrative psychology with investor dynamics to teach students how to build trust, craft compelling startup stories, and confidently pitch to angels, VCs, and grant programs. By the end, students will have investor-ready decks, scripts, and an actionable funding plan.

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

COs	**Statements**
CO1	**Understanding** the psychology and frameworks behind persuasive startup storytelling
CO2	**Structuring** and design compelling investor pitch decks and financial narratives
CO3	**Identifying** appropriate funding stages, types, and investor personas
CO4	**Preparing** due diligence documents, startup data rooms, and term sheet basics
CO5	**Delivering** professional pitch presentations and negotiate funding terms confidently

Course Outline:

Unit Number: 1	**The Art of Startup Storytelling**	**No.of hours: 8**
Content: • Understand narrative arcs: Hero's Journey, Pixar Pitch, StoryBrand • Identify the core story: founder's vision, origin, and mission • Emotion vs Logic: storytelling frameworks that hook and convert		

- Map pain-solution narratives tailored to audience personas
- Practice writing 2-min and 5-min investor-friendly founder stories
- Case studies: Steve Jobs (Apple), Blake Mycoskie (TOMS)

Unit Number: 2	Fundraising Fundamentals	No. of hours: 8

Content:

- Stages of funding: idea, MVP, traction, growth, scale
- Types of funding: bootstrapping, grants, angel, VC, crowdfunding
- Investor types and expectations at each stage
- What VCs look for: market size, defensibility, founding team
- Build a funding roadmap and identify investor-fit matrix
- Learn the SAFE, convertible notes, equity vs debt trade-offs

Unit Number: 3	Pitch Deck Structure & Financial Storytelling	No. of hours: 8

Content:

- The classic 10-slide pitch deck (Guy Kawasaki framework)
  Problem – Solution – Market – Product – Model – Traction – Team – Competition – Financials – Ask
- Visual storytelling: design principles for slide impact
- Crafting compelling "Ask" slides with use-of-funds breakdown
- Financial storytelling: burn rate, runway, CAC, LTV, projections
- Practice recording 90-second elevator pitches and 4-min decks

Unit Number: 4	Live Pitching & Investor Simulation	No. of hours: 6

Content:

- Pitching body language and voice control techniques
- Handling Q&A from mock investors (legal, financial, growth)
- Create startup data rooms (Notion, Drive) with key documents
- Review actual term sheets and mock negotiation simulation
- Analyze successful pitch videos (YC Demo Day, Shark Tank)
- Final pitch event with panel and feedback

Learning Experiences

Inside Classroom Learning*:*

1. Framework-based sessions on Hero's Journey, Pixar Pitch, and Guy Kawasaki deck.
2. Slide-by-slide construction of 10-slide pitch decks with storytelling principles.
3. Live case breakdowns of Airbnb, Razorpay, and Figma investor pitches.
4. In-class financial storytelling practices: burn rate, CAC, LTV, and fundraising ask.

Outside Classroom Learning

1. Video-recorded elevator pitch and founder story assignments with peer reviews.
2. Group critiques of real startup pitch decks to evaluate structure and clarity.
3. Term sheet simulation exercises and mock Q&A with guest mentors.
4. Final pitch presentation to a mock investor panel for evaluation.

Textbooks and Reference Books:

1. Gallo, C. (2014). *Talk like TED: The 9 public-speaking secrets of the world's top minds*. St. Martin's Press.
2. Kawasaki, G. (2015). *The art of the start 2.0: The time-tested, battle-hardened guide for anyone starting anything*. Portfolio.

Case Studies to Demonstrate

1. **Airbnb –** Original pitch deck analysis
2. **Coinbase –** Fundraising through story-driven decks
3. **OYO Rooms –** Founder's pitch journey and Softbank raise
4. **Figma –** Strategic storytelling to secure design-first VCs
5. **Razorpay –** Series A pitch and funding growth over time

Lab/Hands-On Tasks

- Create and deliver a full 10-slide investor pitch deck
- Write a compelling founder origin story under 300 words
- Practice 90-second elevator pitch and get peer-reviewed
- Analyze 3 real startup decks and critique structure & delivery

- Mock investor interview: answer 5 tough funding questions
- Prepare data room checklist and organize all startup docs

Creativity, Imagination & Disruptive Thinking

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name 3: Creativity, Imagination & Disruptive Thinking	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- III			
Pre-requisite(s):				

Course Perspective. This course explores how creativity and imagination fuel disruptive innovations in business, design, and technology. Students will learn structured ideation techniques, problem reframing, lateral thinking, and how to break mental models to unlock unconventional solutions. It combines reflective exercises with collaborative challenges to cultivate a mindset for radical innovation.

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

COs	**Statements**
CO1	**Understanding** cognitive processes behind imagination, creativity, and innovation
CO2	**Applying** structured creative thinking techniques (SCAMPER, 6 Hats, TRIZ, etc.)
CO3	**Reframing** problems to unlock alternative perspectives and solutions
CO4	**Analyzing** case studies of disruptive innovations across industries
CO5	**Designing** and present original, disruptive concepts addressing real-world challenges

Course Outline:

Unit Number: 1	**Foundations of Creative Thinking**	**No.of hours: 8**
Content: • Divergent vs. convergent thinking • Neuropsychology of creativity and imagination • Types of creativity: Deliberate, spontaneous, exploratory		

• Creativity blocks and how to overcome them • Mind mapping, free writing, visual sketching techniques		
Unit Number: 2	**Creativity Tools & Frameworks**	**No. of hours: 8**
Content: • SCAMPER: Substitute, Combine, Adapt, Modify, Put to another use, Eliminate, Reverse • Edward de Bono's Six Thinking Hats • TRIZ (Theory of Inventive Problem Solving) basics • Synectics and metaphorical thinking • Random entry and forced association methods		
Unit Number: 3	**Disruptive Thinking & Innovation**	**No. of hours: 8**
Content: • Clayton Christensen's theory of disruptive innovation • Blue Ocean Strategy and non-consumption markets • Exponential technologies and disruption mapping • First-principles thinking and inversion • Analyze Tesla, Netflix, and Uber as disruption case studies		
Unit Number: 4	**Creative Labs & Disruption Challenges**	**No. of hours: 6**
Content: • 5 design sprints: Empathize → Define → Ideate → Prototype → Test • Reimagine a mundane product using lateral thinking • Brainstorm 10x solutions for real-world wicked problems • Peer-based feedback and idea refinement • Final "Disrupt the Norm" presentation: pitch a disruptive idea		

Learning Experiences

Inside Classroom Learning*:*

1. Fieldwork-based observation and creative redesign of everyday systems.
2. Team design sprints focused on wicked problems across campus or community.

3. Final disruptive solution pitch incorporating empathy, ideation, and prototyping.
4. Peer feedback sessions and reflection logs on creative process evolution.

Outside Classroom Learning

1. Fieldwork-based observation and creative redesign of everyday systems.
2. Team design sprints focused on wicked problems across campus or community.
3. Final disruptive solution pitch incorporating empathy, ideation, and prototyping.
4. Peer feedback sessions and reflection logs on creative process evolution.

Textbooks and Reference Books:

1. De Bono, E. (2017). Six thinking hats. Penguin UK.
2. Christensen, C. M. (2016). The innovator's dilemma: When new technologies cause great firms to fail. Harvard Business Review Press.

Case Studies to Demonstrate

1. **IDEO –** Deep dive method for structured creativity
2. **Netflix –** Disruption through business model innovation
3. **Tesla –** First-principles and market reframing
4. **Post-it Notes (3M) –** Accidental creativity turned into innovation
5. **Airbnb –** Breaking traditional hospitality with design-led thinking

Lab/Hands-On Tasks

- Create 3 mind maps on different themes
- Redesign a household object using SCAMPER
- Host a Six Thinking Hats debate on a social issue
- Identify 5 disruptive startups and map their strategies
- Complete a 5-stage design sprint on a local campus problem
- Final pitch: disruptive product or solution presentation with peer review

Digital Entrepreneurship: Monetizing Skills in the Creator Economy

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name 4: Digital Entrepreneurship: Monetizing Skills in the Creator Economy	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- III			
Pre-requisite(s):				

Course Perspective. This course equips students to become digital entrepreneurs by leveraging their skills and personal brand in the creator economy. It covers platform selection, content monetization, niche building, community engagement, and digital product creation. By the end of the course, students will have launched their own monetizable digital presence and growth roadmap.

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

COs	**Statements**
CO1	**Understanding** the economics and dynamics of digital entrepreneurship and the creator economy
CO2	**Identifying** a niche and build a personal brand around unique skills
CO3	**Launching** and optimize digital content or products on relevant platforms
CO4	**Monetizing** through ads, sponsorships, digital products, and community models
CO5	**Analyzing** and scale growth using content metrics and digital tools

Course Outline:

Unit Number: 1	**Foundations of the Creator Economy**	**No.of hours: 8**
Content: • What is the creator economy? History and trends • Types of creators: educators, influencers, entertainers, builders		

- Passion economy vs gig economy vs traditional entrepreneurship
- Platform deep dive: YouTube, Instagram, Substack, Gumroad, Patreon, etc.
- Finding your niche: Ikigai model and audience discovery

Unit Number: 2	**Personal Branding & Digital Presence**	**No. of hours: 8**

Content:

- Building a personal brand: story, values, aesthetics
- Creating a content strategy: formats, posting frequency, pillars
- Profile optimization across platforms (LinkedIn, Instagram, X, etc.)
- Tools for no-code websites & portfolios (Carrd, Notion, Canva)
- Leveraging newsletters, blogs, and podcasts

Unit Number: 3	**Monetization & Business Models**	**No. of hours: 8**

Content:

- Monetization channels: ads, affiliate, sponsorship, merch, subscriptions
- Selling digital products: templates, courses, e-books
- Membership and community monetization (Discord, Patreon, Circle)
- Setting up payment systems (Stripe, Razorpay, Gumroad)
- Legal basics: GST, income tax, digital contracts, copyright

Unit Number: 4	**Scaling Growth & Content Metrics**	**No. of hours: 6**

Content:

- Monetization channels: ads, affiliate, sponsorship, merch, subscriptions
- Selling digital products: templates, courses, e-books
- Membership and community monetization (Discord, Patreon, Circle)
- Setting up payment systems (Stripe, Razorpay, Gumroad)
- Legal basics: GST, income tax, digital contracts, copyright

Learning Experiences

Inside Classroom Learning*:*

1. Niche discovery workshops using the Ikigai model and platform strategy mapping.

2. Sessions on content monetization methods: affiliate, ads, products, subscriptions.
3. Optimization activities for LinkedIn, YouTube, Substack, or Notion portfolios.
4. Content analytics workshops and feedback on digital product ideas.

Outside Classroom Learning

1. Launch of branded creator accounts and weekly content production.
2. Building and publishing of eBooks, templates, or online mini-courses.
3. Campaign experiments using creator tools like Mailchimp, TubeBuddy.
4. Final showcase of brand identity, monetization plans, and analytics.

Textbooks and Reference Books:

1. Patel, N., & Flynn, P. (2022). *Crush it with content: How to build your brand, grow your audience, and monetize your skills*. Indie Publishing.
2. Subramanian, S. (2023). *The creator economy: A guide to building and scaling a profitable digital career*. Creator Press.

Case Studies to Demonstrate

1. **Ali Abdaal –** Monetizing content through YouTube, Skillshare, and Notion templates
2. **Ranveer Allahbadia (BeerBiceps) –** Creator to digital entrepreneur
3. **Ankur Warikoo –** Digital course empire via Instagram + LinkedIn
4. **MKBHD –** Multi-platform revenue streams and brand collaborations
5. **Saloni Gaur (Nazma Aapi) –** Comedy, social influence & creator branding

Lab/Hands-On Tasks

- Identify and define your niche using the Ikigai model
- Launch a branded content channel (Instagram/YouTube/Substack/etc.)
- Create and publish one digital product (template, eBook, mini-course)
- Design a 4-week content calendar and implement one post per week
- Set up a basic analytics dashboard for content performance
- Final pitch: present your creator brand, monetization plan, and growth roadmap

Design Thinking for Social Innovation

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name 5: Design Thinking for Social Innovation	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- III			
Pre-requisite(s):				

Course Perspective. This course introduces students to design thinking as a human-centered framework to tackle complex social challenges. Learners will explore empathy-driven problem-solving, ideation, prototyping, and iterative development, applying these principles to real-world issues such as sustainability, public health, education, and inclusion.

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

COs	**Statements**
CO1	**Understanding** the design thinking process and its relevance to social impact
CO2	**Conducting** empathy research and identify unmet needs in marginalized communities
CO3	**Generating** creative, user-centric solutions to complex social problems
CO4	**Prototyping** and test social innovations using low-cost methods
CO5	**Developing** and present an implementable impact solution

Course Outline:

Unit Number: 1	**Introduction to Design Thinking & Social Impact**	**No.of hours: 8**
Content: • Principles and stages of design thinking • Systems thinking vs linear problem solving • Role of empathy, ethics, and equity in social innovation		

• Challenges in the social sector: complexity, context, and constraints • Examples from IDEO.org, Stanford d.school, and Ashoka Fellows		
Unit Number: 2	**Empathy & Need-Finding**	**No. of hours: 8**
Content: • Principles and stages of design thinking • Systems thinking vs linear problem solving • Role of empathy, ethics, and equity in social innovation • Challenges in the social sector: complexity, context, and constraints • Examples from IDEO.org, Stanford d.school, and Ashoka Fellows		
Unit Number: 3	**Ideation & Prototyping**	**No. of hours: 8**
Content: • Divergent and convergent thinking • Ideation tools: brainwriting, worst possible idea, SCAMPER • Low-fidelity prototyping: paper, cardboard, digital mockups • Feedback loops and rapid iteration • Testing prototypes with real users		
Unit Number: 4	**Social Innovation & Impact Deployment**	**No. of hours: 6**
Content: • Impact canvas and theory of change • Pilot testing and measuring social impact • Design for scalability and sustainability • Communicating solutions: storytelling and pitch deck • Building partnerships with NGOs, government, or communities		

Learning Experiences

Inside Classroom Learning*:*

1. Empathy-based workshops on journey mapping, stakeholder analysis, and POV.
2. Brainstorming techniques like SCAMPER, worst idea, and visual ideation.

3. Case studies of Embrace Warmer, Aravind Eye Care, and Recyclebank.
4. In-class prototyping labs with low-cost materials and group critiques.

Outside Classroom Learning

1. Field empathy interviews and ethnographic observations.
2. "How Might We" challenge framing and ideation outside the classroom.
3. Prototype testing and refinement with real users or peer groups.
4. Final capstone presentations showcasing innovation solution with impact plan.

Textbooks and Reference Books:

1. Brown, T. (2009). *Change by design: How design thinking creates new alternatives for business and society*. Harvard Business Press.
2. Liedtka, J., & Ogilvie, T. (2011). *Designing for growth: A design thinking toolkit for managers*. Columbia University Press.

Case Studies to Demonstrate

1. **Embrace Infant Warmer –** Low-cost innovation for premature babies in rural India
2. **Aravind Eye Care** – High-quality, low-cost healthcare delivery
3. **Digital Green –** Using tech for agricultural knowledge sharing
4. **Recyclebank –** Gamified recycling awareness
5. **Selco Solar –** Sustainable solar solutions for underserved areas

Lab/Hands-On Tasks

- Conduct 2 empathy interviews on a chosen social theme
- Create a journey map for a target stakeholder
- Frame 3 HMW problem statements based on field research
- Ideate 15+ ideas using brainwriting in groups
- Build and test a low-fidelity prototype for a social innovation
- Final capstone: pitch your impact idea with a prototype and implementation plan

No-Code & Low-Code Product Development

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name 6: No-Code & Low-Code Product Development	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- III			
Pre-requisite(s):				

Course Perspective. This course empowers students to build fully functional digital products — apps, websites, automations, and dashboards — using no-code and low-code platforms. Learners will master tools like Glide, Bubble, Webflow, and Make.com to prototype, launch, and scale products without writing traditional code.

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

COs	**Statements**
CO1	**Understanding** the landscape of no-code/low-code tools and their business value
CO2	**Designing** and **developing** functional MVPs using drag-and-drop builders
CO3	**Integrating** databases, APIs, forms, and third-party services
CO4	**Automating** workflows and optimize digital operations using visual logic
CO5	**Launching** and **testing** a complete no-code/low-code product solving a real-world problem

Course Outline:

Unit Number: 1	**Introduction to No-Code/Low-Code Development**	**No.of hours: 8**
Content: • History, rise, and ecosystem of no-code/low-code platforms • No-code vs low-code vs traditional dev		

- Use cases: internal tools, MVPs, SaaS, e-commerce, marketplaces
- Tool overview: Glide, Bubble, Webflow, Make, Airtable, Softr, Adalo
- Wireframing & UI/UX basics with Figma

Unit Number: 2	Web & Mobile App Building	No. of hours: 8

Content:

- Build responsive apps using Glide and Adalo
- Design landing pages with Carrd/Webflow
- Connect and display dynamic data from Airtable or Google Sheets
- Add forms, filters, buttons, and native mobile components
- Publish web apps with custom domain & SEO optimization

Unit Number: 3	Backend Logic & Integration	No. of hours: 8

Content:

- Use Make.com/Zapier to automate workflows
- Set up logic flows: conditional logic, triggers, APIs
- Send data to Google Sheets, Airtable, or email
- Create dashboards, databases, and automated reports
- Add authentication, payments (Stripe), and third-party plugins

Unit Number: 4	Product Launch & Optimization	No. of hours: 6

Content:

- User testing and feedback loop setup
- Analytics integration: Google Analytics, PostHog
- Optimize for performance, UI, onboarding
- Product Hunt & BetaList launch strategy
- Legal: privacy policy, terms, GDPR basics for product builder

Learning Experiences

Inside Classroom Learning*:*

1. Platform walkthroughs of Bubble, Glide, Softr, Airtable, Webflow.
2. Drag-and-drop project labs for dashboard, landing page, or app creation.
3. Backend automation demos using Make.com, Zapier, or Google Sheets workflows.

4. Guest sessions from no-code founders and live critique of MVPs.

Outside Classroom Learning

1. Independent MVP development and custom domain publishing.
2. Automation flows connecting forms, spreadsheets, and email notifications.
3. User testing with feedback logs and UI optimization.
4. Final MVP demo and walkthrough with peer review.

Textbooks and Reference Books:

1. Wignal, S. (2021). *The No-Code Startup: Launch a Business Without Writing a Line of Code*. Independently published.
2. Koenig, N., & Pistor, C. (2022). *No-Code: Build your idea, launch your startup, scale your business*. NoCode Publications.

Case Studies to Demonstrate

1. **Comet –** A no-code talent platform built using Webflow + Airtable + Make
2. **Dividend Finance –** Built internal CRM using Bubble
3. **Pory.io –** Showcase of powerful Airtable-based site builders
4. **Makerpad Projects –** Showcasing 100+ products launched with no-code
5. **Uncut.fm –** NFT podcast platform built without writing traditional backend

Lab/Hands-On Tasks

- Build a personal portfolio site using Webflow or Carrd
- Create a to-do app or job board using Glide or Softr
- Automate a form-to-email workflow using Make.com or Zapier
- Integrate Google Sheets with a mobile app for dynamic content
- Publish a working MVP solving a real-world problem
- Final presentation: Showcase MVP, integrations, and live demo

Indian Logic and Epistemology (Nyaya and Mimamsa Schools)

Program Name	**B.Tech CSE (Specialization in UI & UX)**			
Course Name 7: Indian Logic and Epistemology (Nyaya and Mimamsa Schools)	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- III			
Pre-requisite(s):				

Course Perspective. This course introduces students to the foundational frameworks of Indian logic (Nyaya) and epistemology (Mīmāṃsā), exploring how ancient Indian philosophers developed systematic methods for reasoning, debate, and knowledge validation. Students will analyze core concepts like pramāṇas (means of knowledge), inference, fallacies, and scriptural interpretation, along with comparisons to Western logical traditions.

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

COs	**Statements**
CO1	**Understanding** the key philosophical concepts of Nyāya and Mīmāṃsā traditions
CO2	**Explaining** the classification and application of valid knowledge (pramāṇa)
CO3	**Applying** the Nyāya system of inference and identify logical fallacies (hetvābhāsa)
CO4	**Analyzing** epistemological debates and their relevance in classical and modern contexts
CO5	**Comparing** Indian and Western logical frameworks to develop critical and intercultural reasoning

Course Outline:

Unit Number: 1	**Foundations of Indian Logic and Philosophy**	**No.of hours: 8**

Content:

- Introduction to the six classical schools (ṣaḍ-darśanas)
- Historical background and significance of Nyāya and Mīmāṃsā
- Basic terminologies: pramā, pramāṇa, pramātṛ, prameya
- Comparison of Indian and Aristotelian logic

Unit Number: 2	**Nyāya System – Tools of Reasoning**	**No. of hours: 8**

Content:

- The four pramāṇas of Nyāya: perception (pratyakṣa), inference (anumāna), comparison (upamāna), and testimony (śabda)
- Structure of inference: pakṣa, hetu, sādhya, dṛṣṭānta, nigamana
- Types of anumāna: pūrvavat, śeṣavat, sāmānyatodṛṣṭa
- Fallacies (hetvābhāsas): asiddha, viruddha, anaikāntika, etc.
- Naiyāyika debate models (vāda, jalpa, vitaṇḍā)

Unit Number: 3	**Mīmāṃsā Epistemology – Scriptural Reasoning**	**No. of hours: 8**

Content:

- Focus on śabda-pramāṇa (verbal testimony) and dharma as knowledge
- Types of sentences in Vedic interpretation
- Concept of apūrva and its role in ritual causality
- Differences between Bhāṭṭa and Prābhākara sub-schools
- Role of intention, context, and non-contradiction in scriptural exegesis

Unit Number: 4	**Contemporary Relevance and Applications**	**No. of hours: 6**

Content:

- Application of Nyāya principles in Indian law, jurisprudence, and pedagogy
- Ethical reasoning and argumentation in Mīmāṃsā
- Dialogue with Western logic (Russell, Frege, Wittgenstein)
- Use of classical logic in contemporary interfaith, intercultural, and philosophical discourse
- Practical workshop: argument analysis using Indian methods

Learning Experiences

Inside Classroom Learning*:*

1. **Create** a 5-step Nyāya-style inference chain from real-world observations
2. **Identify and classify** 5 logical fallacies in daily arguments or media
3. **Interpret** a Vedic sentence using Bhāṭṭa and Prābhākara perspectives
4. **Conduct** a mini-debate following the vāda-jalpa-vitaṇḍā structure
5. **Group activity:** compare Nyāya inference with a Western syllogism

Outside Classroom Learning

1. Create a 5-step Nyāya-style inference chain from real-world observations
2. Identify and classify 5 logical fallacies in daily arguments or media
3. Interpret a Vedic sentence using Bhāṭṭa and Prābhākara perspectives
4. Conduct a mini-debate following the vāda-jalpa-vitaṇḍā structure
5. Group activity: compare Nyāya inference with a Western syllogism

Textbooks and Reference Books:

1. Ganeri, J. (2001). *Philosophy in Classical India: The Proper Work of Reason*. Routledge.
2. Matilal, B. K. (1990). *The Word and the World: India's Contribution to the Study of Language*. Oxford University Press.

Case Studies to Demonstrate

1. **Nyāya debate structure –** Analysis of a classical Nyāya dialogue
2. **Mīmāṃsā and the Gītā –** Scriptural interpretation using Mīmāṃsā methodology
3. **Nyāya vs Carvaka debate –** Logic and materialism clash
4. **Bhāṭṭa vs Prābhākara views –** Conflict in ritual understanding
5. **Contemporary use of pramāṇa in Indian legal judgments**

Lab/Hands-On Tasks

- Create a 5-step Nyāya-style inference chain from real-world observations
- Identify and classify 5 logical fallacies in daily arguments or media

- Interpret a Vedic sentence using Bhāṭṭa and Prābhākara perspectives
- Conduct a mini-debate following the vāda-jalpa-vitaṇḍā structure
- Group activity: compare Nyāya inference with a Western syllogism

Semester: VII

(Specialization Course-VII)
UI/UX Integration

Program Name:	**B. Tech (CSE with specialization in UI/UX)**			
Course Name: UI/UX Integration	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	DSE	3-0-2	4	45
Type of Course:	DSE			
Pre-requisite(s), if any:				

Course Perspective. This capstone-level course focuses on bridging the gap between User Interface (UI) and User Experience (UX) design disciplines by integrating design thinking, research insights, interaction design, accessibility standards, and front-end implementation into a cohesive product workflow. Students will learn how to collaboratively manage design systems, unify user journeys with interface flows, align stakeholder expectations, and deliver developer-ready outputs. Emphasis is placed on industry tools, real-world case studies, cross-functional communication, and hands-on projects simulating actual product design environments.

By the end of the course, students will be able to execute an end-to-end design solution, from user research to functional UI implementation, enabling readiness for roles like Product Designer, UX Strategist, or Design Technologist.

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

COs	Statements
CO 1	Integrating user research, design strategy, and interface design into a unified UX workflow.
CO 2	Collaborating across teams using design systems, tokens, and shared documentation.

CO 3	Applying accessibility, usability, and interaction standards consistently across platforms.
CO 4	Conducting iterative design-validation-development cycles and stakeholder presentations.
CO 5	Delivering a complete product prototype with functional UI handoff artifacts.

Course Outline:

Unit Number: 1	**Title: Bridging UX Strategy and UI Implementation**	**No. of hours: 10**

Objective: Translate UX research and business strategy into design decisions.

Detailed Contents:

- Understanding how UX research outputs (personas, journey maps, empathy maps) inform UI design
- Prioritizing features and flows based on Jobs-To-Be-Done and pain point mapping
- Transforming user flows into interaction blueprints using user stories and scenarios
- Creating storyboards, service blueprints, and task flows to align UX objectives with UI structure
- Aligning business goals, KPIs, and product strategy with user interface decisions

Hands-On:

- Map real user journey data into interaction flows using Miro or FigJam
- Define UI priorities and behaviors based on user pain points
- Create a UX-to-UI traceability matrix for a case study app

Unit Number: 2	**Title: Component Integration & Design System Governance**	**No. of hours: 11**

Objective: Create and manage consistent UI using design systems and reusable components.

Detailed Contents:

- Modular UI thinking: breaking down interfaces into atoms, molecules, and organisms (Atomic Design)

- Introduction to design tokens: spacing, typography, color systems, elevation
- Creating and using shared Figma libraries, components, and variants
- Naming conventions, version control, and documentation practices
- Component lifecycle management and syncing updates across teams

Hands-On:

- Build and publish a reusable Figma design system (buttons, inputs, alerts)
- Set up design tokens using Figma variables or style guide tools
- Audit an existing app's UI for inconsistency

Unit Number: 3	**Title: Cross-Platform Interaction and Developer Collaboration**	**No. of hours: 12**

Objective: Prepare interfaces that are adaptive across devices and ready for developer handoff.

Detailed Contents:

- Responsive and adaptive layout principles for mobile-first and desktop-first approaches
- Platform-specific interaction behaviors: Android Material Design vs iOS HIG vs Web
- Creating layout breakpoints and adaptive design flows
- Documenting design decisions for developer use (e.g., redlines, annotations, spacing rules)
- Using developer collaboration tools: Figma Inspect, Zeplin, Storybook, or Git-based UI libraries

Hands-On:

- Design responsive versions of a user interface for 2+ devices
- Annotate interactive behaviors and spacing rules for development
- Generate and export developer-friendly handoff assets (SVGs, stylesheets, tokens)

Unit Number: 4	**Title: UX Validation, Feedback Loops, and Final Delivery**	**No. of hours: 12**

Objective: Test, iterate, and finalize product interfaces for stakeholder signoff and developer implementation.

Detailed Contents:

- Usability testing for final interactive prototypes: task-based flows, A/B testing, click heatmaps
- Setting success metrics: time on task, satisfaction, NPS, error rate

- Gathering internal (peer/stakeholder) and external (user) feedback
- Iterative refinement based on feedback
- Preparing final documentation: annotated UI, style guide, accessibility checklist, developer instructions

Hands-On:

- Run a usability testing session using a prototype
- Map feedback into a design change log
- Compile a UI delivery package (annotated Figma, interaction notes, and handoff files)

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab Assignment 1 (Unit 1):** Title: "Mapping Research Insights into UI Flow for an e-Learning App" Sub-Objectives: • Choose a research-driven persona and journey map • Identify pain points and translate them to interface goals • Design 3 key screen flows that address user needs • Document UX-to-UI traceability in Notion or Miro	CO 1
2	**Lab Assignment 2 (Unit 2):** Title: "Creating and Managing a Design System for a Microservice Portal" Sub-Objectives: • Define a token set (colors, fonts, spacing) • Create button, card, input field components with documentation • Organize Figma library with naming conventions • Collaborate with team using shared library workflow	CO 2
3	**Lab Assignment 3 (Unit 3):**	CO 3

	Title: "Cross-Platform UI Implementation for a Booking App" Sub-Objectives: • Design 4 screens for desktop and mobile views • Ensure responsive layout and interaction parity • Annotate screens with developer-friendly specs Export Zeplin/Figma Inspect handoff files	
4	**Lab Assignment 4 (Unit 4):** Title: "Interactive Usability Test of a Digital Product Flow" Sub-Objectives: • Conduct usability testing on a working prototype (Figma or Framer) • Design task-based testing and record session outcomes • Collect and synthesize feedback using affinity mapping • Iteratively improve the design and justify decisions	CO 4
5	Title: "End-to-End Product Design and Developer Handoff for a Campus Companion App" Sub-Objectives: • Use existing research or conduct new surveys to identify design needs • Design full interface flow for mobile and desktop screens • Create a shared design system and reusable component library • Prepare complete handoff docs: annotated screens, style guide, component usage • Conduct usability test + developer review session + final iteration	CO 5

Learning Experiences:

In-Class Engagement

- Strategy-to-screen guidance on transforming UX research into UI artifacts.
- Peer critiques and design reviews simulating product design stand-ups.
- Tool walkthroughs (Figma, Miro, Zeplin, Notion) with real-world case integration.

Hands-On Labs

- Weekly labs focused on component creation, responsive design, and usability testing.
- Emphasis on managing design systems and developer-ready outputs.
- Scaffolded capstone project that mirrors professional product team workflows.

Beyond the Classroom

- Cross-team collaboration using shared libraries and documentation tools.
- Usability testing with real users or peer groups for iterative feedback.
- Developer collaboration and stakeholder presentation simulations to validate final deliverables.

Textbooks:

1. Fadeyev, D. (2020). Practical UI: Design Better Interfaces Faster. UXPin Publishing

Generative AI Tools and Techniques

Program Name:	B.Tech CSE (Specialization in UI & UX)			
Course Name: Generative AI Tools and Techniques	Course Code	L-T-P	Credits	Contact Hours
	ETCCTT704	3-0-2	4	45
Type of Course:	Major			
Pre-requisite(s): A foundational understanding of AI Tools and Techniques				

Course Perspective. This course provides a hands-on, practical approach to Generative AI (GenAI) and Large Language Models (LLMs), focusing on writing effective prompts, understanding various LLM architectures, and applying free and open-source models for real-world applications. The course covers both fundamental AI/ML concepts and industry use cases, ensuring students become smart learners and rapid AI solution builders.

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

COs	Statements
CO 1	Understanding the fundamental principles of Generative AI and Large Language Models (LLMs).
CO 2	Writing effective and optimized prompts for text, image, and code generation.
CO 3	Exploring and deploying free open-source LLMs for various industry applications.

CO 4	Building real-world AI-powered applications using Hugging Face, LangChain, OpenAI API, and other open models.

Course Outline:

Unit Number: 1	**Title: Foundations of Generative AI & LLMs**	**No. of hours: 10**
Content • Neural Language Modeling: n-gram vs. neural vs. transformer approaches. • Transformer Architecture: self-attention, multi-head attention, positional encoding, feed-forward blocks, layer normalization. • Tokenization Algorithms: BPE, WordPiece, SentencePiece; impact on vocabulary size & OOV handling. • Training Objectives: causal vs. masked language modeling; next-token prediction; perplexity as a loss surrogate.		
Unit Number: 2	**Title: Prompt Engineering & Model Optimization In-Context Learning Theory**	**No. of hours: 11**
Contents: • **Prompt Taxonomy:** zero-shot, few-shot, Chain-of-Thought (CoT), instruction vs. system prompts. • **Parameter-Efficient Tuning:** LoRA, PEFT, adapters vs. full fine-tuning; trade-offs in compute & data. • **Evaluation & Bias:** calibration, hallucination, toxicity; metrics (BERTScore, factual consistency).		
Unit Number: 3	**Title: Applying Free LLMs for Real-World Use**	**No. of hours: 12**
Contents: • **Retrieval-Augmented Generation (RAG):** vector stores, embeddings, similarity search. • **Code Generation Models:** instruction tuning for coding (StarCoder, Code Llama); evaluation via pass@k. • **Ethical & Legal Considerations:** licensing (Apache 2.0, MIT, CC-BY-SA), data privacy, copyright.		

• **Safety & Alignment:** Reinforcement Learning from Human Feedback (RLHF), Direct Preference Optimization (DPO).		
Unit Number: 4	**Title: Deploying & Integrating LLMs**	**No. of hours: 12**
Contents: • **System Design for LLM Apps: latency vs. throughput, batching,** quantization (8-bit, 4-bit), GPU vs. CPU. • APIs & Micro-services: REST vs. WebSocket, rate limiting, authentication. • Monitoring & Observability: logging prompts, tracing, red-teaming, feedback loops. • Scaling Strategies: sharding, MoE (Mixture-of-Experts), serverless GPU endpoints.		

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	Mini-Project 1 — Unit 1: Foundations of GenAI & LLMs "News-Flash Generator" A digital-media start-up wants an automated service that turns breaking-news bullet points into concise, 280-character social posts. Students must: 1. Choose an open-source causal language model (e.g., GPT-NeoX or Mistral-7B-Instruct). 2. Pre-process live RSS headlines into model-friendly tokens. 3. Generate multiple candidate posts and rank them by perplexity. 4. Visualize a sample attention map to show how the model weighs different tokens.	CO 1

	5. Deliverables: a Colab notebook, a brief technical memo, and a three-minute in-class demo.	
2	**Mini-Project 2 — Unit 2: Prompt Engineering & Optimization** **"Prompt Clinic for Corporate Communications"** An HR department needs AI-drafted policy emails in tones such as formal, friendly, or urgent. Students will: 1. Design at least three prompt templates, each controlling tone, length, and clarity. 2. Compare zero-shot, few-shot, and Chain-of-Thought prompting on a ten-email test set. 3. Measure readability with metrics like the Flesch score and semantic similarity to HR reference docs. 4. Refine prompts—or apply a small LoRA adapter—until the outputs meet style guidelines. 5. Deliverables: a prompt library (JSON or YAML), an evaluation summary, and a short screencast of the workflow.	CO 2
3	**Mini-Project 3 — Unit 3: Applying Free LLMs** **"Code Buddy for Data Analysts"** A company wants an assistant that turns plain-English questions into working Python/Pandas code and explains the result. Students will: 1. Use StarCoder or Code Llama with LangChain to generate code. 2. Build a vector store of Pandas documentation for retrieval-augmented replies. 3. Auto-test generated code on sample CSV files and display explanations. 4. Log and analyze common failure modes such as syntax errors or hallucinations.	CO 3

	5. Deliverables: a Streamlit app, a test report, and a two-minute demo video.	
4	**Mini-Project 4 — Unit 4: Deployment & Integration** **"On-Site Retail Support Chatbot"** A retail chain needs a lightweight chatbot that runs on in-store tablets—offline if the network goes down. Students will: 1. Quantize an 8-b Llama 3 model to 4-bit GGUF and benchmark CPU latency. 2. Integrate a local vector store of inventory and return-policy documents. 3. Wrap the model with FastAPI and design a Gradio kiosk interface. 4. Add basic logging plus a small red-teaming script to check for policy violations. Deliverables: a Dockerfile, a deployment guide, latency metrics, and a live kiosk demonstration.	CO 4
5	**Capstone — Comprehensive Integration** **"AI Knowledge Hub for University FAQs"** Build an end-to-end GenAI platform that answers admissions, course, and campus-life questions for your university. Students will: 1. Collect and clean FAQ PDFs or webpages, then embed them with Sentence-Transformers. 2. Select a base LLM and fine-tune (or LoRA-adapt) it for the university domain. 3. Design prompts, fallback flows, and a feedback loop that records user ratings. 4. Deploy on Hugging Face Spaces (or any free cloud) with CI/CD, add an analytics dashboard, and conduct an accessibility and bias audit.	CO 3, CO 4

	5. **Deliverables:** a fully functional web app, an eight-page technical report, and a final presentation with Q&A.	

Learning Experiences:

Inside the Classroom

- Engage with core theoretical concepts of Generative AI, LLM architectures, tokenization, and model training objectives.
- Learn and practice prompt engineering techniques like zero-shot, few-shot, and Chain-of-Thought.
- Analyze model behaviors, hallucinations, and alignment through guided lab exercises and discussions.
- Work with APIs and tools such as Hugging Face, LangChain, and OpenAI to build simple AI applications in real time.

Outside the Classroom

- Apply concepts in mini-projects simulating real-world use cases—e.g., content generation, chatbots, and code assistants.
- Participate in collaborative coding, prompt evaluation, and testing in teams using platforms like Colab and Streamlit.
- Explore open-source LLMs, deploy applications on free cloud platforms, and conduct ethical audits on AI behavior.
- Develop a Capstone project integrating all units—handling dataset preparation, fine-tuning, deployment, and user feedback mechanisms**.**

Textbooks:

1. Altaf Rehmani, Generative AI for Everyone: Understanding the Essentials and Applications of This Breakthrough Technology.
2. Generative AI in Software Development: Beyond the Limitations of Traditional Coding” Jesse Sprinter, 2024.
3. Deep Learning" by Ian Goodfellow, Yoshua Bengio, and Aaron Courville.
4. Neural Networks and Deep Learning: A Textbook" by Charu C. Aggarwal.

Major Project-I

Program Name:	B.Tech CSE (Specialization in UI & UX)		
Course Name: Major Project-I	Course Code	L-T-P	Credits
	ETCCPR701	0-0-8	4
Type of Course:	PROJ		

Standard Operating Procedure (SOP) for Project Development at School of Engineering & Technology (SOET)

1. Purpose

Project-based learning is a cornerstone of academic delivery at the School of Engineering & Technology (SOET), K.R. Mangalam University. In line with our commitment to experiential and outcome-based education, this SOP establishes structured guidelines for project execution, evaluation, and mentorship across all relevant programs. The objective is to ensure that every student applies theoretical learning to solve real-world problems using cutting-edge technologies and professional development practices.

This SOP outlines the **mandatory framework that will be implemented across SOET** to standardize project development, ensure academic integrity, and promote innovation, collaboration, and technical proficiency.

2. Objectives

All student projects must:

- Employ industry-relevant technologies and tools.
- Reflect a well-defined development or research process from ideation to final deployment or publication.
- Promote innovation and real-world problem-solving.
- Deliver meaningful outcomes such as a functional application, published research, or validated prototype.

3. Project Scope and Categories

Projects must fall under **one** of the following approved categories:

A. Research-Oriented Projects

- Must address futuristic domains like Generative AI (e.g., GANs, Transformers, LLMs), cybersecurity, quantum computing, etc.
- **Multidisciplinary Research Integration:** Projects are encouraged to bridge multiple disciplines by combining core technology domains (e.g., AI, cybersecurity, quantum computing) with fields such as **Healthcare**, **Finance**, **Law & Ethics**, **Education**, **Social Sciences**, or **Environmental Studies**.
- Require mandatory publication of a research paper in a reputed journal or international conference before final submission.
- Supervised by **internal faculty mentor**

B. Industry-Based Projects

- Should be based on real-world industry challenges with practical applications.
- Must demonstrate full implementation, deployment, and usability.
- Students will work under **external mentorship** from an industry along with a faculty mentor from SOET.
- External Industry **Mentorship Confirmation certificate which will** confirms that industry expert/mentor will be guiding the student(s) on their project work. The certificate must be **issued and signed by the external mentor**. It should be printed on the **official letterhead of the company/organization** (if available).
- A certificate of completion from the associated organization is mandatory.

C. University-Focused Interdisciplinary Projects

- Projects under this category should aim to identify and address real-world challenges, enhancement opportunities, or operational gaps within K.R. Mangalam University.

- Students are encouraged to collaborate across disciplines, integrating knowledge from diverse domains to develop innovative, practical, and sustainable solutions for campus improvement.
- Projects must demonstrate a clear understanding of institutional needs and propose **scalable or implementable models** that can be piloted or adopted by university stakeholders.

D. Start-Up Projects

- Projects under this category should focus on identifying real-world problems and developing innovative, technology-driven solutions with the potential for commercialization.
- Students are encouraged to think entrepreneurially, transforming their ideas into viable products or services by building functional prototypes or Minimum Viable Products (MVPs).
- Projects must include key elements of a start-up framework such as market research, user validation, business model design, and go-to-market strategy.
- The outcome should demonstrate not only technical feasibility but also market relevance, with the potential to **seek incubation or funding support from the university's innovation/start-up ecosystem.**

4. Technology Stack and Industry Tools

- Students must select a technology stack aligned with current industry standards, such as MERN, MEAN, Django, Flutter, or cloud-native architectures (AWS, Azure, GCP).
- Projects must incorporate industry-relevant tools and frameworks for **version control (Git/GitHub),** containerization (Docker), CI/CD (GitHub Actions, Jenkins), and agile development practices.
- Use of modern development environments, APIs, and deployment platforms is strongly encouraged to ensure scalability, maintainability, and real-world applicability of the solution.

5. Formation of Groups

5.1 Group Size & Composition

- **Optimal Group Size**: Groups will consist of **2 students** to allow effective collaboration while maintaining manageable group dynamics.

- **Skill Diversity**: Form groups with a mix of skill sets (e.g., coding, design, research, communication, etc.) to enhance collaboration. Students should be encouraged to select group members based on complementary skills.

5.2 Group Formation Process

- **Self-selection**: Allow students to form their own groups, encouraging them to choose teammates based on project interests.
- **Pre-formed Groups**: Alternatively, groups can be assigned by the coordinator to ensure diversity of skills and ideas across all teams.

5.3 Team Finalization Rule

Once teams are formed and registered, **no changes in team composition will be allowed** under any circumstances.

5.4 Timeline

Week	Activity/Stage
Week 0	• Onboarding on **Projexa** (Project Management Tool) • **Team formation** • **Faculty Mentor allotment**
Week 1–2	• **Problem Statement Finalization** • Initial **Mentor Interactions** and discussion
Week 3–4	• **Synopsis Submission** • **Project Proposal Evaluation and Approval**
Week 5,6 and 7	**Implementation Sprint 1:** Core Modules, MVP Build, Mentor Interaction

Week	Activity/Stage
Week 8	**Development Phase – Part 1** • Module-wise implementation begins
Week 9–10	• **Development Phase – Part 2** • Module-wise implementation begins
Week 11–12	**Mid Term Evaluation**
Week 13–14	• **Testing and Debugging** • Performance evaluation
Week 15–16	• **Final Report Submission** • **Project Video (1-2 mins) demonstrating working project** • **Presentation & Viva Preparation**

5.5 Guidelines for Project Selection

For any project category (Research-Based, Development-Based, Start-Up, or University-Focused Interdisciplinary), students may select their project through either of the following approaches:

1. **Faculty-Suggested Problem Statements:**
   - Faculty members will provide curated problem statements based on their domain expertise, current research trends, or institutional needs.
   - Students can choose from these problems and discuss with the faculty mentor before finalizing.

2. **Student-Proposed Problem Statements:**
   - Student teams are encouraged to identify real-world problems based on personal interests, industry trends, or societal challenges.

- The proposed problem must be original, relevant, and feasible within the given timeline and resources.
- Final approval is subject to evaluation by the project review committee.

5.6 Faculty Allocation Process

1. **Faculty-Suggested Problem Statement:**
   If a team selects a problem provided by a faculty member, the team will be allocated to that faculty **after peer review and approval** by the Project Coordinator.
2. **Student-Proposed Problem Statement:**
   If the team proposes its own problem statement, then **after peer review and validation** by the Project Coordinator, an **appropriate faculty mentor** will be assigned based on domain expertise.

6. Roles and Responsibilities of Faculty Mentors

1. **Regular Interactions:**
   Conduct regular meetings with the assigned team(s) through **Projexa** in online/offline mode. All meeting records, including agenda, minutes, and attendance, must be documented and uploaded on Projexa.
2. **Task Monitoring:**
   Assign tasks with clear deadlines and **track the progress and completion status** for each team through the project cycle.
3. **Evaluation After Interaction:**
   After every interaction, the **faculty mentor must evaluate the team's progress** and update the evaluation/comments directly on **Projexa** as part of the meeting record.
4. **Meeting Confirmation & Rescheduling:**
   Faculty must **respond to meeting requests** initiated by student teams. If unable to conduct a scheduled meeting, it should be **rescheduled within one week**.
5. **Justification for Meeting Cancellation:**
   Any cancellation of scheduled team meetings by the mentor must be supported with a **valid and documented reason**.

6. **GitHub Activity Monitoring:**
   Regularly monitor each team's **GitHub repository** to verify individual contributions and maintain **performance records** for all team members.

7. **Project-Related Achievements (Hackathons, Ideathons, etc.)**

Students who actively participate in events such as Hackathons, Ideathons, Innovation Challenges, or other project-related competitions during the project cycle are eligible for gaining marks under the **Achievements category**.

- Only the **participating member(s)** of the project team will be eligible for claiming marks under this category.
- Valid proofs such as a (certificate of participation, geo-tagged images) must be submitted for verification.
- A **maximum of 10 marks** can be awarded in this category.
    - **Participation in a National-level event**: **5 marks**
    - **Participation in an International-level event**: **10 marks**

All submissions will be reviewed and approved by the Project Coordinator before the marks are awarded.

8. **Evaluation Metrics**

Projects will be evaluated in 3 phases (Total 100 Marks)

a. Synopsis Presentation (20 Marks)
b. Mid Term Presentation (30 Marks)
c. Final Term Presentation (40 Marks)

For each evaluation, marks will be cumulated from the following components:

a. Project Mentor Evaluation
b. Project Evaluation Committee

Evaluation Stage	**Project Mentor**	**Project Evaluation Committee**

Synopsis Presentation(20)	**5**	**15**
Mid Term Presentation (30)	**10**	**20**
Final Term Presentation (40)	**10**	**30**
Recognition of Participation in technical events during Project Cycle(Achievements)		**10**

Project Evaluation Committee Marking Scheme

Evaluation Stage	**Criteria**	**Marks**
Synopsis Presentation	Creative/Naive/Real-world Problem	5
	Clarity & Feasibility of Project Objectives	5
	Preparation & Presentation of PPT	5
	Total	**15**
Mid-Term Presentation	Project Efforts – UI/Implementation	10
	Effective Use of Modern Technology Stack	5
	Effective Coding Practices / Use of Git	5
	Total	**20**
Final-Term Presentation	Final Production Demonstration & Completion w.r.t Objectives	10
	Impact & Use Cases of the Project	10
	Deployment	10
	Total	**30**

9. Policy on Missed Presentation Schedules

Students are **strictly required to adhere to the assigned schedules** for all project presentations, including the **Synopsis**, **Mid-Term**, and **Final-Term** evaluations.

- **Only one rescheduling opportunity** will be granted in case a team or individual misses their assigned presentation slot. This rescheduling must be approved in advance (where possible) or justified immediately after the missed session with valid reasons.
- A **penalty of 5 marks** will be **deducted for each missed presentation schedule**, irrespective of the evaluation stage.
- **Failure to appear** even after the rescheduled opportunity will result in **zero marks** being awarded for that evaluation component.

This policy ensures fairness, accountability, and professionalism in the evaluation process.

10. Project Documentation Requirements – Stage-Wise

1. Synopsis Stage

To be submitted before the Synopsis Presentation:

Synopsis Report

✅ Synopsis PPT

✅ Mentorship Confirmation Certificate

✅ Must be on industry letterhead, duly signed by the External Mentor

2.Mid-Term Evaluation Stage

To be submitted during the Mid-Term Presentation:

- ✅ Mid Evaluation PPT
- ✅ Project Use Case & Deployment Confirmation Certificate
    - Issued by the Industry Mentor, confirming project relevance and real-world deployment potential

3. Final-Term Evaluation Stage

To be submitted during or before Final Presentation:

- ✅ Final Project Report
- ✅ PPT for Final Presentation
- ✅ **Video Demonstration (1–2 minutes)**
    - Should clearly document:
- Key features of the project
- Execution workflow or demonstration of functionality
- Real-world impact or use-case

To be submitted in MP4 format or via a shared video link (as per instructions)

- ✅ **Plagiarism Certificate**
    - Must be compulsorily appended to the Final Project Report
    - Should be obtained using institutional or approved plagiarism checking tools

11. Plagiarism & Ethical Guidelines

- Maximum allowable plagiarism (e.g., ≤ 15%)
- Check plagiarism using open-source tools and proper citations
- No reuse of previous projects
- Intellectual property rights (if applicable)
- **Compulsory Plagiarism Certificate to be appended in the report file**

12. Project Outcome Requirements by each Project Category

Project Category	Expected Outcome
Industry-Based Project	• Project Use Case & Deployment Confirmation Certificate from **Industry Mentor** • **Fully deployed** solution (mandatory for final evaluation)
Research-Based Project	• **Proof of publication** in a reputed **conference or journal** (UGC CARE / Scopus / IEEE / Springer, etc.)
In-House Project	• Project Use Case & Deployment Confirmation Certificate from **Faculty Mentor** • **Fully deployed** solution (mandatory for final evaluation)
Start-Up Project	• **Incubation Letter** from recognized incubator/mentor/startup cell • Product **delivery and deployment** with functional MVP

Summer Internship-III
(Assessment)

Program Name:	B.Tech CSE (Specialization in UI & UX)		
Course Name: Summer Internship-III (Assessment)	Course Code	L-T-P	Credits
	ETCCIN702	0-0-4	2
Type of Course:	INT		

Course Perspective: The Summer Internship Program (1st June – 31st July) is designed to integrate academic learning with real-world professional experiences, enabling students to apply theoretical knowledge to practical situations. It forms a mandatory part of the Semester VII for students currently in Semester VI, carrying a weightage of **2 academic credits**.

The key objectives of the Summer Internship Program are:

- To enhance professional skills and industry readiness.
- To expose students to real-world technical, managerial, and research practices.
- To promote self-learning, professional responsibility, and critical thinking.
- To foster connections between academic knowledge and industry practices.

Duration

The duration of the internship will be 6-8 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):

Students must produce a joining letter at the start and a relieving letter upon completion.

Government/Research Institution Internship:

Students can engage in a research internship with premier government or research organizations such as IITs, IISc, ISRO, DRDO, CSIR, NPL, etc.

On-Campus Bootcamp/ Industry Internship Programs:

The university will offer on-campus internships in collaboration with industry partners.

Deliverables and Documentation:

Each student must submit the following after completing their internship/certification:

Deliverable	Description	Marks
Summer Internship File	A detailed report/file based on the provided format including objectives, methodology, learnings, and reflections.	10 Marks
Video Presentation	A 7–10-minute recorded video presentation showcasing work done during the internship/certification. The template of slides will be shared.	20 Marks
Certificate of Completion	A color-printed certificate on bond paper from the host organization/certification body, mentioning duration, role/project.	70 Marks

Evaluation Metrics

The Summer Internship will be evaluated based on the following comprehensive criteria:

Evaluation Component	Weightage	Description
Internship Report/File	10%	Completeness, professional formatting, relevance to internship tasks.

Video Presentation	20%	Content quality, clarity, communication skills, professional presentation.
Certificate of Completion	70%	Authenticity, completion of internship/certification within stipulated time, relevance to program objectives.

Internship Evaluation Rubric:

S. No.	Component	Sub-Component / Criteria	Marks
1	Internship Certificate	**Relevance to Core Subjects**	**20 Marks**
		- Directly relates to core subjects	20
		- Partially relates to core subjects	15
		- Minimally relates to core subjects	10
		- Not relevant	0
2	Report Submission	**Structure and Organization**	**10 Marks**
		- Well-structured and organized report	10
		- Moderately structured report	7
		- Poorly structured report	3
		- No structure	0
3	Solo Video-Based Evaluation	**a. Technical / Professional / Soft Skills Acquired**	**10 Marks**
		- Highly relevant and advanced technical skills	10
		- Moderately relevant technical skills	8
		- Basic technical skills	5
		- No new skills acquired	0

S. No.	Component	Sub-Component / Criteria	Marks
		b. Content Delivery	**10 Marks**
		- Clear, engaging, and thorough delivery	10
		- Clear but less engaging delivery	7
		- Somewhat clear and engaging delivery	3
		- Unclear and disengaging delivery	0
		c. Visual Aids & Communication Skills	**10 Marks**
		- Effective visual aids + excellent communication skills	10
		- Moderate visual aids + good communication skills	7
		- Basic visual aids + fair communication skills	3
		- No visual aids + poor communication skills	0
4	**Internship Duration**	**Weeks Completed**	**10 Marks**
		- 6–8 weeks completed	10
		- 4–6 weeks completed	8
		- Less than 1 month	5
5	**Outcome of the Internship**	**Application / Project / Key Learnings & Findings**	**30 Marks**
		- Clear, outcome-based project with applied learnings and key findings	25–30
		- Moderate outcome with partial application and findings	15–24
		- Minimal outcome, unclear learning/application	0–14

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 intern- ship 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, team- work, time management, and adaptability in a professional setting.
- **Present Findings Effectively:**
  - Deliver clear and engaging presentations to effectively communicate project outcomes, research findings, and acquire 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.

SEMESTER: VIII

Summer Internship-IV (Assessment)

Program	**B.Tech CSE (Specialization in UI & UX)**		
Course Name: **Summer Internship-IV (Assessment)**	**Course Code**	**L-T-P**	**Credits**
	ETCCIN801	**0-0-16**	**8**
Type of Course:	**INT**		
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 Summer Internship-IV (Assessment)

1. Introduction

This is an essential academic requirement aimed at providing students with the opportunity to apply theoretical knowledge to practical challenges. The project is designed to foster critical thinking, problem-solving, innovation, and research-oriented learning, with a focus on real-world industrial, research, and entrepreneurial domains. Students may choose from:

- **Industrial Project**: Solving real industrial problems in collaboration with an industry partner.
- **Research & Development (R&D) Project**: Contributing to academic and applied research, with external guidance from academic/research institutions.
- **Start-up Project**: Developing and launching innovative start-up ideas with entrepreneurial mentors.

The SOP ensures that the project aligns with **NEP 2020 guidelines**, emphasizing interdisciplinary, practical, and outcome-based learning.

2. Objectives

The primary objectives of the full-time project are:

- **Application of Theoretical Knowledge**: Enabling students to apply their academic learning to practical problems.
- **Holistic Development**: Promoting interdisciplinary learning, critical thinking, creativity, and problem-solving.
- **Research and Innovation**: Encouraging innovative solutions, leading to publications, patents, or prototypes.
- **Industry Collaboration**: Fostering partnerships with industries for real-world problem-solving.
- **Entrepreneurship Development**: Developing entrepreneurial skills and creating viable start-ups.
- **Global Competency**: Ensuring students develop the skills required to excel in global environments through research, innovation, and collaboration.

3. Types of Projects

a) Industrial Project

Students working on **Industrial Projects** will:

- Collaborate with an industry partner.
- Identify specific, real-world challenges faced by the company.
- Propose and implement a solution that provides value to the industry.
- Develop a final product or prototype that can be implemented in the industrial setting.

Project Proposal:

- Problem Statement and Objectives: Identify the industrial problem and outline the objectives.
- Proposed Solution: Present a detailed methodology for solving the problem.
- Deliverables: Define tangible deliverables, including prototypes, software, or hardware.
- Expected Impact: Outline the expected impact on the industry.

Evaluation Criteria:

- Practical implementation and solution viability (40%)
- Project innovation (20%)
- Industrial applicability and impact (20%)
- Final presentation and report quality (20%)

b) Research & Development (R&D) Project

The **R&D Project** focuses on creating innovative research outcomes through collaborations with academic or research institutions. This can result in publications, research reports, or new discoveries.

Project Proposal:

- Literature Review: Detailed research on existing work related to the chosen topic.
- Hypothesis/Research Questions: Define the specific research problem or question.
- Methodology: Include data collection, experimental design, and analysis techniques.
- Research Timeline: Step-by-step phases of research with milestones.

External Mentor: Collaboration with an **external academic expert** is mandatory for research projects. The external mentor must be a research professional with expertise in the specific field of study.

Internal Mentor: Each student will also be assigned an **internal faculty member** who will supervise the project. The internal mentor will ensure that the research meets academic standards and deadlines.

Evaluation Criteria:

- Quality of Research and Novelty (30%)
- Research Methodology (25%)
- Contributions to the field (20%)
- Final Report, Presentation, and Publication (25%)

c) Start-up Project

The **Start-up Project** involves developing a business model or creating a start-up venture. Students work on a product/service idea that addresses a significant market need or societal problem.

Project Proposal:

- Start-up Idea: Explain the business or product idea.
- Market Research: Detailed research on the market, target customers, competitors, and potential revenue streams.
- Business Plan: Define the steps needed to take the idea to market, including funding, development phases, marketing, and operational plans.
- Product Prototype: If applicable, develop a working prototype.

Mentorship:

- **External Mentor**: An industry/start-up expert will guide the student in refining the idea, business model, and market strategy.
- **Internal Faculty Mentor**: An internal mentor will provide academic guidance and ensure the start-up idea is feasible and innovative.

Evaluation Criteria:

- Start-up viability and market potential (30%)
- Product or service innovation (30%)
- Prototype/Business Model Development (20%)
- Final Pitch/Presentation and Start-up Plan (20%)

4. Roles and Responsibilities

a) Student's Responsibilities:

- Select a suitable project topic based on interests (industrial, R&D, or start-up).
- Draft and submit a detailed proposal with objectives, methodology, timelines, and deliverables.
- Coordinate with both external and internal mentors regularly for feedback and guidance.
- Maintain a weekly progress report for both mentors.
- Submit a final comprehensive report and present the project.

b) Internal Supervisor:

- Guide the student throughout the project.
- Provide academic input and ensure that the project aligns with the program outcomes.
- Conduct progress reviews and ensure timelines are adhered to.
- Evaluate the project at the mid-term and final stages.

c) External Mentor:

- Offer specialized industrial, research, or entrepreneurial guidance.
- Provide real-world problem insights for industrial and start-up projects.
- Ensure the project is relevant to the chosen industry, research domain, or start-up ecosystem.
- Participate in the final evaluation of the project.

5. Project Phases

Phase 1: Proposal Submission and Approval

- Students will submit a project proposal during the first two weeks of the final semester.
- The proposal must include the problem statement, objectives, literature review (for R&D projects), methodology, and expected outcomes.
- The proposal is subject to review and approval by the internal supervisor and external mentor.

Phase 2: Planning and Resource Allocation

- Once approved, the student will develop a project plan that includes:
    - **Project Milestones**: Break down the project into smaller tasks with defined milestones.
    - **Resource Requirements**: Identify any software, hardware, lab resources, or tools required for the project.
    - **Team Roles**: For group projects, define the roles of each team member.
    - **Risk Assessment**: Highlight potential risks and the corresponding mitigation strategies.

Phase 3: Mid-term Review

- A mid-term review will be conducted halfway through the project to assess progress.

- Students will present their work to a committee consisting of the internal supervisor, external mentor, and department head.
- The review will assess the progress against the timeline and suggest course corrections if needed.

Phase 4: Final Execution and Evaluation

- **Industrial Projects**: Students must submit a prototype or industrial report, demonstrating the solution's applicability to the industry.
- **R&D Projects**: Students must submit a final research report or publish findings in academic journals.
- **Start-up Projects**: Students must present a business plan, along with a working prototype, market analysis, and revenue model.

Phase 5: Final Report Submission and Presentation

- **Final Report**: The project report should contain a title page, abstract, introduction, problem statement, objectives, methodology, results, discussion, conclusions, future scope, references, and appendices.
- **Presentation**: Students will deliver a final presentation to a panel of evaluators, showcasing their work, findings, or product.
- **Evaluation**: Based on the final report and presentation, students will be awarded marks in accordance with the evaluation rubrics.

6. Collaboration and Mentorship

For **Research Projects**, the mentorship will involve both:

- **External Mentor**: An academic expert outside the institution, preferably from a reputed university or research institute.
- **Internal Mentor**: A faculty member from the student’s department to provide academic and administrative guidance.

For **Industrial Projects**:

- External mentorship will come from industry professionals, preferably from the partnering company.

For **Start-up Projects**:

- External mentorship will involve experienced entrepreneurs, start-up founders, or investors.

Mentors will:

- Provide critical inputs on the technical, business, or research aspects of the project.
- Offer feedback and advice during each phase of the project.

7. NEP 2020 Guidelines

The project structure is designed to ensure interdisciplinary learning and foster entrepreneurial and research innovation, in line with the **NEP 2020** guidelines:

- **Interdisciplinary Approach**: Students are encouraged to explore projects that bridge different fields of study.
- **Flexibility**: Students have the flexibility to choose between industrial, research, or start-up projects.
- **Experiential Learning**: Real-world problem-solving and hands-on project work are at the core of this initiative.
- **Collaboration**: The integration of external mentors ensures industry and academic collaboration.

8. Documentation and Submission Requirements

Students are required to:

- Submit their proposal, mid-term report, final report, and any supporting documents via the **Learning Management System (LMS)**.
- Maintain detailed project logs and weekly reports.

Major Project-II

Program	B.Tech CSE (Specialization in UI & UX)		
Course Name: Major Project-II	Course Code	L-T-P	Credits
	ETCCPR802	0-0-8	4
Type of Course:	PROJ		
Pre-requisite(s), if any:			

Standard Operating Procedure (SOP) for Project Development at School of Engineering & Technology (SOET)

1. Purpose

Project-based learning is a cornerstone of academic delivery at the School of Engineering & Technology (SOET), K.R. Mangalam University. In line with our commitment to experiential and outcome-based education, this SOP establishes structured guidelines for project execution, evaluation, and mentorship across all relevant programs. The objective is to ensure that every student applies theoretical learning to solve real-world problems using cutting-edge technologies and professional development practices.

This SOP outlines the **mandatory framework that will be implemented across SOET** to standardize project development, ensure academic integrity, and promote innovation, collaboration, and technical proficiency.

2. Objectives

All student projects must:

- Employ industry-relevant technologies and tools.
- Reflect a well-defined development or research process from ideation to final deployment or publication.
- Promote innovation and real-world problem-solving.
- Deliver meaningful outcomes such as a functional application, published research, or validated prototype.

3. Project Scope and Categories

Projects must fall under **one** of the following approved categories:

A. Research-Oriented Projects

- Must address futuristic domains like Generative AI (e.g., GANs, Transformers, LLMs), cybersecurity, quantum computing, etc.
- **Multidisciplinary Research Integration:** Projects are encouraged to bridge multiple disciplines by combining core technology domains (e.g., AI, cybersecurity, quantum computing) with fields such as **Healthcare**, **Finance**, **Law & Ethics**, **Education**, **Social Sciences**, or **Environmental Studies**.
- Require mandatory publication of a research paper in a reputed journal or international conference before final submission.
- Supervised by **internal faculty mentor**

B. Industry-Based Projects

- Should be based on real-world industry challenges with practical applications.
- Must demonstrate full implementation, deployment, and usability.
- Students will work under **external mentorship** from an industry along with a faculty mentor from SOET.
- External Industry **Mentorship Confirmation certificate which will** confirms that industry expert/mentor will be guiding the student(s) on their

project work. The certificate must be **issued and signed by the external mentor**. It should be printed on the **official letterhead of the company/organization** (if available).

- A certificate of completion from the associated organization is mandatory.

C. University-Focused Interdisciplinary Projects

- Projects under this category should aim to identify and address real-world challenges, enhancement opportunities, or operational gaps within K.R. Mangalam University.
- Students are encouraged to collaborate across disciplines, integrating knowledge from diverse domains to develop innovative, practical, and sustainable solutions for campus improvement.
- Projects must demonstrate a clear understanding of institutional needs and propose **scalable or implementable models** that can be piloted or adopted by university stakeholders.

D. Start-Up Projects

- Projects under this category should focus on identifying real-world problems and developing innovative, technology-driven solutions with the potential for commercialization.
- Students are encouraged to think entrepreneurially, transforming their ideas into viable products or services by building functional prototypes or Minimum Viable Products (MVPs).
- Projects must include key elements of a start-up framework such as market research, user validation, business model design, and go-to-market strategy.
- The outcome should demonstrate not only technical feasibility but also market relevance, with the potential to **seek incubation or funding support from the university's innovation/start-up ecosystem.**

4. Technology Stack and Industry Tools

- Students must select a technology stack aligned with current industry standards, such as MERN, MEAN, Django, Flutter, or cloud-native architectures (AWS, Azure, GCP).
- Projects must incorporate industry-relevant tools and frameworks for **version control (Git/GitHub),** containerization (Docker), CI/CD (GitHub Actions, Jenkins), and agile development practices.

- Use of modern development environments, APIs, and deployment platforms is strongly encouraged to ensure scalability, maintainability, and real-world applicability of the solution.

5. Formation of Groups

5.1 Group Size & Composition

- **Optimal Group Size**: Groups will consist of **2 students** to allow effective collaboration while maintaining manageable group dynamics.
- **Skill Diversity**: Form groups with a mix of skill sets (e.g., coding, design, research, communication, etc.) to enhance collaboration. Students should be encouraged to select group members based on complementary skills.

5.2 Group Formation Process

- **Self-selection**: Allow students to form their own groups, encouraging them to choose teammates based on project interests.
- **Pre-formed Groups**: Alternatively, groups can be assigned by the coordinator to ensure diversity of skills and ideas across all teams.

5.3 Team Finalization Rule

Once teams are formed and registered, **no changes in team composition will be allowed** under any circumstances.

5.4 Timeline

Week	Activity/Stage
Week 0	• Onboarding on **Projexa** (Project Management Tool) • **Team formation** • **Faculty Mentor allotment**
Week 1–2	• **Problem Statement Finalization** • Initial **Mentor Interactions** and discussion

Week	Activity/Stage
Week 3–4	• **Synopsis Submission** • **Project Proposal Evaluation and Approval**
Week 5,6 and 7	**Implementation Sprint 1:** Core Modules, MVP Build, Mentor Interaction
Week 8	**Development Phase – Part 1** • Module-wise implementation begins
Week 9–10	• **Development Phase – Part 2** • Module-wise implementation begins
Week 11–12	**Mid Term Evaluation**
Week 13–14	**• Testing and Debugging** • Performance evaluation
Week 15–16	• **Final Report Submission** • **Project Video (1-2 mins) demonstrating working project** • **Presentation & Viva Preparation**

5.5 Guidelines for Project Selection

For any project category (Research-Based, Development-Based, Start-Up, or University-Focused Interdisciplinary), students may select their project through either of the following approaches:

1. **Faculty-Suggested Problem Statements:**
   - Faculty members will provide curated problem statements based on their domain expertise, current research trends, or institutional needs.
   - Students can choose from these problems and discuss with the faculty mentor before finalizing.

2. **Student-Proposed Problem Statements:**
   - Student teams are encouraged to identify real-world problems based on personal interests, industry trends, or societal challenges.
   - The proposed problem must be original, relevant, and feasible within the given timeline and resources.
   - Final approval is subject to evaluation by the project review committee.

5.6 Faculty Allocation Process

1. **Faculty-Suggested Problem Statement:**
   If a team selects a problem provided by a faculty member, the team will be allocated to that faculty **after peer review and approval** by the Project Coordinator.
2. **Student-Proposed Problem Statement:**
   If the team proposes its own problem statement, then **after peer review and validation** by the Project Coordinator, an **appropriate faculty mentor** will be assigned based on domain expertise.

6. Roles and Responsibilities of Faculty Mentors

1. **Regular Interactions:**
   Conduct regular meetings with the assigned team(s) through **Projexa** in online/offline mode. All meeting records, including agenda, minutes, and attendance, must be documented and uploaded on Projexa.
2. **Task Monitoring:**
   Assign tasks with clear deadlines and **track the progress and completion status** for each team through the project cycle.
3. **Evaluation After Interaction:**
   After every interaction, the **faculty mentor must evaluate the team's progress** and update the evaluation/comments directly on **Projexa** as part of the meeting record.

4. **Meeting Confirmation & Rescheduling:**
   Faculty must **respond to meeting requests** initiated by student teams. If unable to conduct a scheduled meeting, it should be **rescheduled within one week**.
5. **Justification for Meeting Cancellation:**
   Any cancellation of scheduled team meetings by the mentor must be supported with a **valid and documented reason**.
6. **GitHub Activity Monitoring:**
   Regularly monitor each team's **GitHub repository** to verify individual contributions and maintain **performance records** for all team members.
7. **Project-Related Achievements (Hackathons, Ideathons, etc.)**

Students who actively participate in events such as Hackathons, Ideathons, Innovation Challenges, or other project-related competitions during the project cycle are eligible for gaining marks under the **Achievements category**.

- Only the **participating member(s)** of the project team will be eligible for claiming marks under this category.
- Valid proofs such as a (certificate of participation, geo-tagged images) must be submitted for verification.
- A **maximum of 10 marks** can be awarded in this category.
  - **Participation in a National-level event**: **5 marks**
  - **Participation in an International-level event**: **10 marks**

All submissions will be reviewed and approved by the Project Coordinator before the marks are awarded.

8. Evaluation Metrics

Projects will be evaluated in 3 phases (Total 100 Marks)

d. Synopsis Presentation (20 Marks)
e. Mid Term Presentation (30 Marks)
f. Final Term Presentation (40 Marks)

For each evaluation, marks will be cumulated from the following components:

c. Project Mentor Evaluation
d. Project Evaluation Committee

Evaluation Stage	Project Mentor	Project Evaluation Committee
Synopsis Presentation(20)	**5**	**15**
Mid Term Presentation (30)	**10**	**20**
Final Term Presentation (40)	**10**	**30**
Recognition of Participation in technical events during Project Cycle(Achievements)		**10**

Project Evaluation Committee Marking Scheme

Evaluation Stage	Criteria	Marks
Synopsis Presentation	Creative/Naive/Real-world Problem	5
	Clarity & Feasibility of Project Objectives	5
	Preparation & Presentation of PPT	5
	Total	**15**
Mid-Term Presentation	Project Efforts – UI/Implementation	10
	Effective Use of Modern Technology Stack	5
	Effective Coding Practices / Use of Git	5
	Total	**20**
Final-Term Presentation	Final Production Demonstration & Completion w.r.t Objectives	10
	Impact & Use Cases of the Project	10
	Deployment	10

Evaluation Stage	Criteria	Marks
	Total	**30**

9. Policy on Missed Presentation Schedules

Students are **strictly required to adhere to the assigned schedules** for all project presentations, including the **Synopsis**, **Mid-Term**, and **Final-Term** evaluations.

- **Only one rescheduling opportunity** will be granted in case a team or individual misses their assigned presentation slot. This rescheduling must be approved in advance (where possible) or justified immediately after the missed session with valid reasons.
- A **penalty of 5 marks** will be **deducted for each missed presentation schedule**, irrespective of the evaluation stage.
- **Failure to appear** even after the rescheduled opportunity will result in **zero marks** being awarded for that evaluation component.

This policy ensures fairness, accountability, and professionalism in the evaluation process.

10. Project Documentation Requirements – Stage-Wise

1. Synopsis Stage

To be submitted before the Synopsis Presentation:

- ✅ Synopsis Report
- ✅ Synopsis PPT
- ✅ Mentorship Confirmation Certificate
- Must be on industry letterhead, duly signed by the External Mentor

2. Mid-Term Evaluation Stage

To be submitted during the Mid-Term Presentation:

- ✅ Mid Evaluation PPT
- ✅ Project Use Case & Deployment Confirmation Certificate
    - Issued by the Industry Mentor, confirming project relevance and real-world deployment potential

3. Final-Term Evaluation Stage

To be submitted during or before Final Presentation:

- ✅ Final Project Report
- ✅ PPT for Final Presentation
- ✅ **Video Demonstration (1–2 minutes)**
    - Should clearly document:
- Key features of the project
- Execution workflow or demonstration of functionality
- Real-world impact or use-case

To be submitted in MP4 format or via a shared video link (as per instructions)

- ✅ **Plagiarism Certificate**
    - Must be compulsorily appended to the Final Project Report
    - Should be obtained using institutional or approved plagiarism checking tools

11. Plagiarism & Ethical Guidelines

- Maximum allowable plagiarism (e.g., ≤ 15%)
- Check plagiarism using open-source tools and proper citations
- No reuse of previous projects
- Intellectual property rights (if applicable)
- **Compulsory Plagiarism Certificate to be appended in the report file**

12. Project Outcome Requirements by each Project Category

Project Category	Expected Outcome
Industry-Based Project	• Project Use Case & Deployment Confirmation Certificate from **Industry Mentor** • **Fully deployed** solution (mandatory for final evaluation)
Research-Based Project	• **Proof of publication** in a reputed **conference or journal** (UGC CARE / Scopus / IEEE / Springer, etc.)
In-House Project	• Project Use Case & Deployment Confirmation Certificate from **Faculty Mentor** • **Fully deployed** solution (mandatory for final evaluation)
Start-Up Project	• **Incubation Letter** from recognized incubator/mentor/startup cell • Product **delivery and deployment** with functional MVP