K.R. MANGALAM UNIVERSITY

THE COMPLETE WORLD OF EDUCATION

SCHOOL OF ENGINEERING AND TECHNOLOGY

Programme Handbook

(Programme Structure and Evaluation Scheme)

B. Tech CSE

(Specialization in Cybersecurity)

Programme Code: 41

FOUR YEAR UNDERGRADUATE PROGRAMME

(with effect from 2025-26 session)

Approved in the 38th Meeting of Academic Council

Held on 28th June 2025

K.R. MANGALAM UNIVERSITY

TABLE OF CONTENTS

1. Preface

About University:

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

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

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

Uniqueness of KRMU

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

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

Education Objectives

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

ii. To undertake research programs with industrial interface.

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

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

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

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.

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

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

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

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

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

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.

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

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

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

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

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 centers, industries, and professional bodies.
- Enhance leadership qualities among the youth having understanding of ethical values and environmental realities

4. About the School

5. About School of Engineering and Technology:

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

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

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

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

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

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

6. 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.

7. About the Programme

7.1 Definitions

- **Programme Outcomes (POs)**

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

- **Programme Specific Outcomes (PSOs)**

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

- **Programme Educational Objectives (PEOs)**

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

- **Credit**

Credit refers to a unit that measures the amount of academic work required for a course. It typically reflects the number of instructional hours per week. For theory courses, one credit is equivalent to 14-15 hours of classroom instruction over a semester. For practical courses, such as workshops, labs, or tutorials, one credit is equivalent to 28-30 hours of instructional or hands-on work over a semester.

7.2 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.

7.3 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 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.

7.4 PROGRAMME SPECIFIC OUTCOMES (PSO's)

PSO1: Understanding the concepts, theories, tools, techniques, and methodologies of Cyber Security.

PSO2: Applying Cyber Security concepts, techniques, methodologies & principles to protect systems and networks from threats.

PSO3: Analyzing security context, threats, vulnerabilities, and incidents to identify risks.

PSO4: Evaluating and implementing security measures and solutions.

PSO5: Designing and developing innovative security strategies to address complex cyber threats.

7.5 Career Avenues

Diverse career avenues available to graduates of the B. Tech CSE with specialization in Cybersecurity programme:

Cybersecurity Analyst: In an era of heightened cybersecurity threats, organizations require experts who can protect their systems and data. Cybersecurity analysts identify vulnerabilities, implement security measures, conduct risk assessments, and respond to security incidents to safeguard computer systems and networks.

Security Consultant: Security Consultants advise organizations on how to improve their security posture. They assess existing security measures, design and recommend security solutions, and help implement best practices to protect against cyber threats

Cybersecurity Engineer: Cybersecurity Engineers develop and implement security solutions to protect information systems. They work on creating security infrastructure, developing tools for threat detection, and ensuring the overall security of IT environments.

Malware Analyst: Malware Analysts study and analyse malicious software to understand how it works, how it spreads, and how to defend against it. They reverse-engineer malware, analyse threats, and develop countermeasures to protect systems.

Ethical Hacking Instructor/Trainer: Ethical Hacking Instructors or Trainers educate others in cybersecurity practices. They develop training materials, conduct workshops, and teach ethical hacking techniques and methodologies to aspiring cybersecurity professionals.

Forensic Computer Analyst: Forensic Computer Analysts investigate cybercrimes by collecting, preserving, and analysing digital evidence. They work on legal cases, recover data from compromised systems, and provide expert testimony in court.

Network Security Engineer: Network Security Engineers focus on protecting network infrastructures from cyber threats. They design and implement

network security solutions, manage firewalls and intrusion detection systems, and ensure network security compliance.

Cryptographic Engineer: Cryptographic Engineers develop and implement cryptographic systems to secure data and communications. They work on encryption algorithms, secure key management practices, and cryptographic protocols for protecting sensitive information

Security Researcher: Security Researchers investigate new vulnerabilities and emerging threats. They conduct original research, develop new security technologies, and publish findings to advance the field of cybersecurity.

Digital Forensics Expert: Digital Forensics Experts analyse electronic evidence in criminal investigations. They collect, preserve, and examine digital evidence from devices to support legal cases and investigate security breaches. Research and Development: Graduates can pursue careers in research and development, working on innovative projects, exploring new technologies, and pushing the boundaries of computer science and cybersecurity fields. This can involve academic research, industry research labs, or research and development departments within companies.

Prospective Companies

- Symantec Corp.
- Quick Heal Technologies
- Microsoft
- IBM
- RSA
- McAfee
- AVG Technologies
- Valency Networks
- Alten Calsoft Labs
- Palo Alto Networks
- Deloitte Consulting
- KPMG
- Ernst & Young

7.6 Duration

4 Years (8 Semesters) - Full-Time Program

7.7 Eligibility Criteria for Award of Degree

Students must successfully complete the minimum required credits of 176 to be eligible for award of degree.

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

a. Education Philosophy and Purpose:

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

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

b. University Education Objective

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

c. Importance of Structured Learning Experiences:

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

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

Inside the Classroom:

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

Outside the Classroom:

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

d. **Educational Planning and Execution**

The B.Tech in Computer Science & Engineering (CSE) with Specialization in Cyber Security 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 with specialization in Cybersecurity programme, students are introduced to the foundational concepts of engineering mathematics, physics/chemistry, and programming. This phase is designed to strengthen their understanding of core scientific and technical principles. Courses such as Engineering Calculus, Fundamentals of Computer Programming using Python, and Basics of Electrical &

Electronics Engineering provide a strong foundation. Students also engage in hands-on laboratory sessions to complement theoretical learning, which helps them connect classroom knowledge with real-world applications.

Orientation Program: The university conducts a **one-day orientation program** for first-year students to familiarize them with the university's environment and key aspects. During the program, students are introduced to the university's highlights, important procedures, key functionaries, and the code of conduct. This orientation serves to ensure that students are well-informed and prepared for a smooth transition into university life.
In the first year, students are exposed to critical problem-solving approaches, basic programming, and ethics in engineering, laying the groundwork for their technical and professional growth.

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

Core Learning

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

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

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

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

Skill Development

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

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

Capstone and Exit Phase

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

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

Co-Curricular and Extra-Curricular Activities

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

Community Connect

Aligning with the NEP 2020's vision of social responsibility, the B.Tech CSE with specialization in Cybersecurity programme includes community engagement through activities like Extension Projects and social service initiatives. Students work on community projects and participate in programs aimed at addressing local and national challenges, promoting civic responsibility, and developing empathy towards society.

Ethics and Professional Values

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

Career Counseling and Entrepreneurship

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

Course Registration

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

e. Student Support Services

Mentor-Mentee: At K.R. Mangalam University, the **Mentor-Mentee Program** plays a crucial role in fostering academic and personal growth. Each student is assigned a faculty mentor who serves as a guide throughout their academic journey. This program ensures continuous interaction, where mentors assist students with academic planning, help in resolving personal issues, and provide career guidance. The mentor-mentee

relationship transcends the classroom and often involves personal development, professional growth, and overall well-being. The program aims to nurture a supportive environment that enhances the learning experience and helps students reach their full potential.

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

f. Evaluation of Learning:

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

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	**50 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.

Program Structure & Evaluation Scheme

Program Name	B.Tech CSE (Specialization in Cybersecurity)
Total Credits	176
Total Semesters	8

Credit Distribution Summary

Program Name	I	II	III	IV	V	VI	VII	VIII	Total Credits
B.Tech CSE (Cyber)	23	22	27	26	24	24	16	14	176

SEMESTER I

K.R. MANGALAM UNIVERSITY

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
7	SEC	SEC	Design Thinking & Prototyping	1	0	2	2
8	VAC	VAC-I	Environmental Studies & Disaster Management	2	0	0	2
		TOTAL		19	0	8	23

K.R. MANGALAM UNIVERSITY

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	SEC	SEC	Maker Lab: Tinkering with Technology	1	0	2	2
7	OEC	Open Elective	*Open Elective-I	3	0	0	3
		TOTAL		16	0	12	22

* Summer Internship (6-8 Weeks)

K.R. MANGALAM UNIVERSITY

SEMESTER III

S.N		Category	Course Title	L	T	P	Credits
1	ETCCNT301	Major	Nand to Tetris-I	3	0	2	4
2	ETCYCS305	DSE	Specialization Course-I	3	0	2	4
3	ETCCWD303	Major	Web Dev -III (Node.js & Express Backend)	3	0	2	4
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

***Open Elective: Students can choose one of the electives from the pool of open electives of University**

K.R. MANGALAM UNIVERSITY

SEMESTER IV

		Category	Course Title	L	T	P	Credits
1	ETCCNT401	Major	Nand to Tetris-II	3	0	2	4
2	ETCYNS 402	DSE	Specialization Course-II	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

***Open Elective: Students can choose one of the electives from the pool of open electives of University**

*** Summer Internship (6-8 Weeks)**

VAC-II

K.R. MANGALAM UNIVERSITY

SEMESTER V

S.N		Category	Course Title	L	T	P	Credits
1	ETCYCT503	DSE	Specialization Course-III	3	0	2	4
2	ETCYAS505	DSE	Specialization Course-IV	3	0	2	4
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

VAC-III

K.R. MANGALAM UNIVERSITY

SEMESTER VI

		Category	Course Title	L	T	P	Credits
1	ETCYPT602	DSE	Specialization Course-V	3	0	2	4
2	ETCYIS703	DSE	Specialization Course-VI	3	0	2	4
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

*** Summer Internship (6-8 Weeks)**

K.R. MANGALAM UNIVERSITY

Discipline Specific Elective

(Cybersecurity)

SN	Category	COURSE CODE	COURSE TITLE	L	T	P	C
1	DSE	ETCYCS305	Foundations of Cyber Security	3	0	2	4
2	DSE	ETCYNS402	Network Security	3	0	2	4
3	DSE	ETCYCT503	Cryptography	3	0	2	4
4	DSE	ETCYAS505	Application Security	3	0	2	4
5	DSE	ETCYPT602	Ethical Hacking and Penetration Testing	3	0	2	4
6	DSE	ETCYCS605	Cloud Security	3	0	2	4
7	DSE	ETCYIS703	Digital Forensics and Incident Response	3	0	2	4

SEMESTER VII

S.N		Category	Course Title	L	T	P	Credits
1	ETCYCS605	DSE	Specialization Course-VII	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**

SEMESTER VIII

	Category	Course Title	L	T	P	Credits
ETCCIN801	**INT**	**Summer Internship-IV (Assessment)**	**0**	**0**	**16**	**8**
ETCCPR802	**PROJ**	**Major Project-II**	**0**	**0**	**8**	**4**
MOOC	**MOOC-II**	**MOOC (Swayam/ NPTEL/AICTE's ELIS)**	**2**	**0**	**0**	**2**
	TOTAL		**2**	**0**	**24**	**14**
	TOTAL Credits=176					

Syllabus

K.R. MANGALAM UNIVERSITY

SEMESTER: I

Computational Mathematics-I

Program Name	**B. Tech CSE (Specialization in Cyber Security)**			
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/optimisation, 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 minimisation
- 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.

Classroom 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 Cyber Security**			
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: 15**
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: 13**
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: 10**

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

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.

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. 5. Present results in a formatted table and allow repeated analysis until the user exits.	CO 2
3	**Library Inventory Manager"**	CO 3

	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.	
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.** • Data ingestion: read multiple monthly CSV files into	CO 4

	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	

ENGINEERING CHEMISTRY

Department:	**B. Tech CSE with Specialization in Cyber Security**			
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 analyze 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.

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.

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

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

Program Name	**B. Tech CSE (Specialization in AI & Robotics)**			
Course Name:	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**

Web Dev -I (HTML ,CSS, JS Basics)	**ETCCWD103**	3-0-2	4	45
Type of Course:	Major			
Pre-requisite(s): Familiarity with basic computer operations and logical thinking; prior exposure to any programming fundamentals is helpful but not mandatory				

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	Building structured and accessible web pages using HTML5.
CO 2	Designing responsive layouts using modern CSS (Flexbox, Grid, media queries, animations).
CO 3	Writing basic JavaScript programs using functions, conditions, loops, arrays, and objects.
CO 4	Creating a complete multi-section website integrating HTML, CSS, and JavaScript basics.

Course Outline:

Unit Number: 1	**Title: HTML & Web Foundation**	**No. of hours: 10**
• What is HTML and its role in web development • Structure: <!DOCTYPE html>, <html>, <head>, <body> • Common tags: headings, paragraphs, , <hr> • Links using <a>: external, internal, email		

- Images with <img>: src, alt, responsive setup
- Multimedia: <audio>, <video> with controls
- Lists: <ul>, <ol>,
- Tables: <table>, <tr>, <td>, <th>
- Forms: <form>, <input>, <textarea>, <select>, <button>
- Form attributes: type, placeholder, required, label
- Semantic tags: <header>, <footer>, <section>, <nav>, etc.
- Accessibility & SEO basics with semantic HTML
- Tools: VS Code, Live Server, HTML validators
- Practice: mini projects like recipe cards & signup forms

Unit Number: 2	**Title –CSS : Core & Advanced**	**No. of hours: 15**

Content:

- What is CSS and its role in styling web pages
- Types of CSS: Inline, Internal, External — with pros & cons
- CSS Colors: using HEX, RGB, HSL formats
- Custom fonts: Google Fonts & @font-face usage
- CSS Selectors: basic, element, attribute, class, ID, combinators
- Applying CSS to HTML elements correctly
- CSS Box Model: content, padding, border, margin
- CSS Positioning: static, relative, absolute, fixed
- Basic layouts: centering elements, column layout, image galleries
- Styling workflow: external stylesheets, organizing rules, debugging via dev tools
- Display properties in CSS: inline, block, inline-block, and flex
- Flexbox basics: justify-content, align-items, flex-wrap, and layout structure
- Creating responsive layouts using Flexbox techniques
- Introduction to CSS Grid: defining columns/rows, grid areas, and layout control
- CSS units: relative (rem, em, %, vw, vh) vs absolute (px)
- Media queries: syntax and usage for mobile-first responsive design
- CSS transitions & transformations for smooth visual effects
- Keyframe animations: defining and triggering custom animations
- Adding hover effects and interactive feedback with CSS.
- Practice: styling real HTML pages and comparing CSS methods
- Practice: responsive layouts, media queries, and UI animations using Flexbox & Grid.

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

Content:

- Intro & Setup: What is JS, using <script>, setting up in browser & VS Code.
- Variables & Data Types: var, let, const; primitive (string, number, boolean, null, undefined, symbol, bigint) vs reference (arrays, objects, functions).
- Core Methods:
- String: length, slice(), replace(), toUpperCase(), split(), etc.
- Number: toFixed(), parseInt(), Math.random(), etc.
- Operators & Comments: Arithmetic, assignment, comparison, logical; // and /* */.
- Control Flow: Conditionals (if, else, switch), loops (for, while, do-while, break, continue).
- Functions: Declaration, arrow, anonymous, parameters, return, default values.
- Scope & Closures: Global, local, block scope, lexical scope.
- Callbacks & Higher-Order Functions.
- Arrays: CRUD operations and methods (push(), map(), filter(), reduce(), etc).
- Objects: Creation, property access, Object.keys(), deep vs shallow copy, for...in loop.

Hands-on focus: progressively enhance earlier layouts with interactive behaviors.

Unit Number: 4	Title: Capstone Project	No. of hours: 10

Content:

- Build a fully structured, responsive web interface using HTML5 & CSS3, demonstrating real-world layout, semantic structure, and visual design.
- Choose one theme:
- Online Learning Platform: Landing page with course cards, testimonials, pricing, and contact form.
- Real Estate Listing Platform: Property grid with filters, image galleries, and agent info sections.
- Fintech Dashboard: UI for financial services showing transaction cards, stats, and pricing plans.
- Food Delivery App: Menu page with featured items, category filters, restaurant cards.
- Travel Agency Website: Tour packages with banners, reviews, itinerary sections, and inquiry form.
- E-commerce Store UI: Product listing, filter sidebar, and product detail layout with add-to-cart UI.

- Requirements:
- Use only HTML5 & CSS3 (Flexbox/Grid mandatory).
- Apply responsive design using media queries.
- Structure must follow semantic HTML practices.
- Include forms, lists, images, and navigation links.
- Showcase custom styling, hover effects, and good use of fonts & colors.
- Outcome: A fully responsive, visually polished, and semantically correct static site representing a real-world use case.

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. **UI/UX Feedback Loops:** Share designs with peers or mentors and incorporate feedback to improve aesthetics and usability.
7. **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

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	• **Lab 1** – "Personal Portfolio Website" (Unit 1) • Build a one-page portfolio using semantic tags: <header>, <nav>, <section>, and <footer>. • Add internal navigation links to About, Projects, and Contact sections. • Include a hero section with a title and short intro. • Add a profile image with proper alt text in the About section. • List 2–3 projects using or <article>. • Create a contact form with name, email, and message fields. • Use label, required, and placeholder attributes in the form. • Add a "Skip to main content" link for accessibility. • Ensure all links work and are clearly labeled. • Pass the W3C HTML validator with zero errors. • Structure HTML cleanly to allow easy CSS styling later.	CO 1

	• Practice Concepts:
	  • Use , , and <blockquote> in the About section for emphasis.
	  • Add a simple <table> to list technical skills or tools.
	  • Embed a placeholder image in the hero or projects section using <img> with proper
	
	Practice Concepts:
	
	• Use , , and <blockquote> in the About section for emphasis.
	• Add a simple <table> to list technical skills or tools.
	• Embed a placeholder image in the hero or projects section using <img> with proper
	• alt.
	• Practice <a> tags linking to your resume or GitHub profile	

2	**Lab 2 – Responsive Personal Portfolio Website (HTML + CSS) (Unit 2)**
	
	• One-page responsive layout using semantic HTML and modern CSS.
	• Fully styled sections: Hero, About, Projects, and Contact.
	• Clean and readable typography, spacing, and layout.
	• Flexbox layout for header and footer.
	• Internal navigation with smooth scrolling.
	• Accessible contact form with styled input fields and validation hints.
	• Mobile-friendly design with media queries.
	• Basic hover effects on links and buttons.
	• Consistent color theme and section spacing.
	• Practice Concepts:
	  • Apply different font pairings using Google Fonts.
	  • Try two different layout methods: Flexbox for header/footer, Grid for project cards.
	  • Create a simple media query to change layout at 768px.
	  • Add a smooth hover transition to buttons or project cards.
		CO 2

3	**Lab 3 – "Prompt Quizzer – A Quiz Game" (Unit 3)** • Built using only basic JavaScript, no HTML or DOM required • Uses prompt() to take user input • Uses alert() to give immediate feedback • Contains a set of predefined quiz questions • Compares answers using toLowerCase() and trim() • Tracks the user's score throughout the game • Displays the final score at the end • Uses a for loop to go through each question • Runs directly in the browser console • Helps practice conditionals, loops, arrays, and functions **Practice Concepts:** ▪ Practice writing at least 5 question objects with question, options, and answer. ▪ Use toLowerCase() and trim() to handle varied user input. ▪ Add a final message based on score (e.g., "Well done!" or "Try again"). ▪ Write your own custom scoring logic (e.g., +2 for correct, -1 for wrong).	CO 3
4	**Lab 4 – Capstone – Responsive Website Project (Choose One Theme) (All Units)** • One-page or multi-section responsive website • Clean layout using Flexbox and/or CSS Grid • Semantic HTML structure with proper use of <header>, <section>, <footer>, etc. • Custom styling using CSS variables, colors, fonts, and spacing • Responsive design with media queries for mobile and desktop views • Interactive navigation bar with internal links • Use of cards to display items (e.g., products, courses, properties) • Image usage with proper alt text for accessibility • Organized and consistent section spacing	CO 3, CO4

	• Includes a call-to-action area (e.g., inquiry form, sign-up form) • Footer with contact links or social media icons • Fully functional layout using only HTML and CSS (no JavaScript required) **Practice Concepts:** • Use consistent CSS variables for fonts, colors, and spacing across the layout. • Structure your site using a CSS Grid or flex for section. • Add a testimonial or pricing section as an extra enhancement. • Create a custom responsive navbar that adapts on small screens.	

Computer Science Fundamentals & Career Pathways

Program Name	**B. Tech CSE (Specialization in AI & Robotics)**			
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

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

Program Name	**B. Tech CSE (Specialization in Cybersecurity)**			
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:

Unit Number: 1	**Title: Empathy and Problem Framing**	**No. of hours: 5**
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

Unit Number: 2	**Title: Ideation and Paper Prototyping**	**No. of hours: 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
- 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

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

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.

Instructor Guidelines:

- Adopt a coaching mindset, encouraging autonomy and creativity.
- Regularly review progress through shared trackers (Google Sheets/Notion).
- Conduct periodic design critiques in the "design studio" format.
- Schedule mid-course retrospectives for adjustments in teaching strategies.

Teaching Resources:

- Core References:
  - 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
- Figma Resources:
    - Figma Education Resources
    - FreeCodeCamp Figma Crash Course (YouTube)

Software / Tools Required

- **Figma** (Free for students)
- **Balsamiq** (or similar lo-fi wireframing tool)
- **Miro / Jamboard** for collaboration
- **Canva / Notion / Google Docs** for documentation

Design Thinking & Prototyping Lab Equipment Requirements

Equipment Name	Specifications / Description	Qty (for 60 students)
Whiteboards	Mobile whiteboards for ideation, sketching, and mapping user journeys	6 large panels
Sticky Notes (Post-its)	Assorted colors and sizes for brainstorming and affinity mapping	100 pads
Dot Stickers / Markers	For dot voting and ideation ranking	60 sheets
Empathy Mapping Templates	A3 printed templates for group empathy mapping exercises	30 templates
Persona Canvas Sheets	Pre-designed persona sheets for design research	30 templates
User Journey Mapping Sheets	Pre-formatted A2/A1 paper or foam boards	30 sheets or boards
A3/A2 Sketching Pads	For wireframing and paper prototyping	60 pads
Marker Sets (Fine & Bold)	For sketching, mapping, and whiteboard work	30 sets
Scissors / Paper Cutters	For physical prototyping exercises	20 pairs
Glue Sticks / Tape	Standard adhesive tools for low-fidelity physical mockups	30 sets
Prototyping Stationery Kit	Paper, foam board, cardboard sheets, popsicle sticks, straws, etc.	5 shared kits
Digital Drawing Tablets (Optional)	Wacom or XP Pen for students wanting to do digital sketching	5 units (shared)
Laptops / Desktops	With internet access, and pre-installed software (Figma, Balsamiq, Canva, Google tools)	30 systems (or BYOD)
High-Speed Internet	Wi-Fi with access to cloud tools (Figma, Miro, Canva, Google Docs)	Lab-wide

Projector & Screen	For design presentations, walkthroughs, tutorials	1 set
Color Printers (A4/A3)	For printing personas, user journey maps, wireframes	2 printers
Storage Lockers / Cabinets	For safely storing project materials and supplies	10–12 lockers
Collaborative Software Licenses	Free/Edu plans for: Figma, Canva, Notion, Miro, Jamboard	As needed
Audio Recorders / Smartphones	For recording user interviews and feedback	10 shared devices or BYOD
Video Recording Setup	For recording final project walkthroughs / pitching sessions	1 basic camera or tripod
Tabletop Presentation Boards	For showcasing final team projects during review	15–20 boards
Flexible Furniture	Movable desks, modular seating, idea zones for team collaboration	Configurable for 10 teams
LED Desk Lamps (Optional)	For close-up design work or usability testing ambiance	10 shared

Software Requirements

Software / Platform	**Purpose**	**License Type**
Figma	UI/UX design and prototyping	Free for students/teams
Balsamiq	Lo-fi wireframing	Free trial / Edu license
Canva	Graphic design, reports, presentation	Free / Edu version
Miro / Jamboard	Real-time collaborative whiteboarding	Free tier for education
Notion	Project documentation, templates	Free for students
Google Docs/Slides	Collaborative writing and presentation	Free

K.R. MANGALAM UNIVERSITY

SEMESTER: II

Computational Mathematics – II

Program Name	B.Tech (CSE) with specialization in Cybersecurity			
Course Name:	Course Code	L-T-P	Credits	Contact Hours

Computational Mathematics - II	**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


K.R. MANGALAM UNIVERSITY

K.R. MANGALAM UNIVERSITY

Data Structures

Program Name	**B.Tech (CSE) with specialization in Cybersecurity**			
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.

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 books:

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

Lab Experiments

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).

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:

1. **Dynamic Array Simulation:**
    a. 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).
    b. Analyze the amortized time complexity of append().
2. **Linked List Operations:**
    a. Implement a SinglyLinkedList class with methods for insert_at_beginning(), insert_at_end(), delete_by_value(), and traverse().
    b. Extend it to a DoublyLinkedList and implement insert_after_node() and delete_at_position().
3. **Stack & Queue from Scratch:**
    a. Implement a Stack ADT using your SinglyLinkedList implementation as the underlying storage. Include push(), pop(), and peek() methods.

b. Implement a Queue ADT using your SinglyLinkedList implementation as the underlying storage. Include enqueue(), dequeue(), and front() methods.
c. Discuss the time complexity of each operation for your linked-list based Stack and Queue.

4. **Application: Parentheses Checker:**
   a. Use your Stack implementation to check for balanced parentheses in a given string (e.g., "({[]})").

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.

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.

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:

1. **Binary Search Tree (BST) Operations:**
    a. 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).
    b. Test your BST with a series of insertions and deletions.
2. **Graph Traversal:**
    a. Represent a small, directed, weighted graph (e.g., 5-7 nodes, 8-10 edges) using an adjacency list.
    b. Implement Breadth-First Search (BFS) starting from a given node.
    c. Implement Depth-First Search (DFS) starting from a given node.
    d. Print the traversal order for both BFS and DFS.
3. **Hash Table with Collision Handling:**
    a. Implement a HashTable class using separate chaining for collision resolution.
    b. Include insert(key, value), get(key), and delete(key) methods.
    c. Choose a simple hash function (e.g., modulo operator) and demonstrate its usage.
    d. Insert several key-value pairs, including some that cause collisions, and demonstrate retrieval and deletion.

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.
    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).

K.R. MANGALAM UNIVERSITY

Web Dev -II (Advanced JS & React)

Program Name	B.Tech (CSE) with specialization in Cybersecurity			
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: Foundations of Node.js	No. of hours: 12

- Overview of Backend Development & HTTP fundamentals
- Introduction to Node.js and event-driven architecture
- JavaScript for Node: ES6+ syntax, async/await, Units
- OOP in JavaScript: constructor functions, classes, this, and prototypes
- Core Node.js Units: fs, http, path, url

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

Unit Number: 2	Title: Express.js & REST API Development	No. of hours: 10

Content Summary:

- Introduction to Express.js
  - Routing: static/dynamic routes, route parameters
  - Middleware concepts: built-in, custom, error-handling
  - Controllers, routers, modular structure for scaling apps
  - Express Generator and folder best practices
  - REST Principles: statelessness, resources, HTTP verbs
  - Full CRUD API:
- Part 1 – POST (Create), GET (Read)
- Part 2 – PUT/PATCH (Update), DELETE
  - Error handling patterns: try-catch, centralized middleware, status codes
  - Building and testing APIs using Postman

Unit Number: 3	Title: Databases, Authentication & Security	No. of hours: 12

- Intro to Databases: SQL vs NoSQL, MongoDB setup
- Data modeling with Mongoose or Sequelize
- Schema design, validations, and references
- Querying & Populating: .find(), .populate(), aggregation basics
- Real-time data flow with full database CRUD integration
- Authentication basics: password hashing with bcrypt
- Token-based Auth: JWT setup, signing, and verification
- Protecting routes with role-based middleware
- Environment variables with dotenv and secret management
- Best practices for securing credentials and APIs

Unit Number: 4	Title: AI, Design & Deploymen	No. of hours: 11

Content Summary:

- Low-Level Design (LLD): modularity, SOLID principles, code reuse
- Automated Testing: writing basic unit & integration tests with Jest or Mocha
- Integrating Generative AI APIs (e.g., OpenAI, HuggingFace)
- Retrieval-Augmented Generation (RAG) for smarter backend features
- Docker basics: Dockerfile, image, container, and Docker Compose
- Deployment: cloud hosting on Render, Vercel, or Railway
- Wrap-up and final review

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

Lab Experiments

Lab 1 – Simple Express API (Unit 1)

- Build a basic Express server with 3–4 endpoints
- Use GET and POST methods to serve dummy data
- Add custom middleware to log requests
- Implement 404 and global error handler Practice Concepts: Routes, status codes, middleware chain, modular routing

Lab 2 – CRUD API with MongoDB (Units 2)

- Use Mongoose or Sequelize to model a simple resource (e.g., Products or Tasks)
- Implement CRUD: Create (POST), Read (GET), Update (PUT/PATCH), Delete (DELETE)
- Handle validation and errors Practice Concepts: Schema design, .save(), .find(), .findByIdAndUpdate(), .deleteOne()

Lab 3 – Auth & JWT (Unit 3)

- Add user model with password encryption
- Setup login and register routes
- Generate JWT on successful login
- Protect a private route using middleware Practice Concepts: bcrypt, jsonwebtoken, protected routes, status codes

Lab 4 – AI-Powered Endpoint (Unit 3,4)

- Build a POST API route that integrates a generative AI API
- Send user input to the AI and return a formatted response
- Handle errors and API keys via .env Practice Concepts: External APIs, fetch/axios, prompt formatting, env vars

Capstone Project – Book My Show Full Stack Project

Objective: Build and deploy a full-featured Full stack Project

- Project Requirements:
- Design a modular Express API with CRUD and JWT auth
- Connect to a MongoDB or PostgreSQL database
- Add a route powered by an AI API (e.g., summarizer, etc.)
- Write at least 3 unit tests
- Containerize and deploy the API
- Document endpoints using Swagger or simple markdown
- Deliverables: Live deployed backend, GitHub repo with code and README, demo video (optional)

ENGINEERING PHYSICS

Program Name:	**B. Tech CSE (Specialization in Cyber Security)**			
Course Name: Engineering Physics	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCPH204	2-0-2	3	30
Type of Course:	Major			
Pre-requisite(s), if any: Knowledge of Basic physics and calculus (differentiation, integration).				

Course Perspective. This Engineering Physics course is designed to provide B.Tech 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	**Semiconductor Physics for Electronic and Computing Devices**	**No. of hours: 10**
Content: • **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: 5**
Content: • 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	**Optical Physics in Modern Computing**	**No. of hours: 8**
Content: • 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		

<table>
<tr><td colspan="3">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.</td></tr>
<tr><td>Unit Number: 4</td><td>Optical Communication and Fiber Optics</td><td>No. of hours: 7</td></tr>
<tr><td colspan="3">Content:
• 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.</td></tr>
</table>

Learning Experience

Inside Classroom Learning Experience:

- Interactive Lectures: Use PPTs to explain key concepts.
- Conceptual Understanding: Cover topics like SHM, polarization, and laser principles through real-world examples and problem-solving.
- Problem-Solving Sessions: Engage students with in-class exercises on mechanics, optics, and electromagnetic waves to strengthen theoretical understanding.
- Theory Assignments: Assign problems on polarization and nanomaterials to be solved during class and discussed with peers.
- Group Work: Students collaborate on real-world engineering tasks, such as analyzing optical systems using lasers or fiber optics.
- Case Studies: Discuss engineering applications of superconducting materials and smart materials to connect theory with industry practices.
- Continuous Feedback: Conduct quizzes and short assessments in class to provide immediate feedback on students' grasp of core physics principles.

Outside Classroom Learning Experience:

- Theory Assignments: Assign take-home problems that require applying physics concepts like laser technology or material properties to practical situations.
- Question Bank: Encourage students to practice with model questions related to mechanics, optics, and materials for self-assessment and revision.
- Online Forums: Students participate in online discussions to collaborate on solving physics problems, particularly in areas like polarization and diffraction.
- Self-Study: Research and explore modern physics technologies, such as the latest advancements in superconducting materials or nanotechnology.
- Collaborative Projects: Work in teams to design innovative applications of physics concepts, such as building models based on polarization or developing smart materials for engineering challenges.

Textbooks:

1. Fundamentals of Physics" by David Halliday, Robert Resnick, and Jearl Walker
2. University Physics with Modern Physics" by Hugh D. Young and Roger A. Freedman

Reference Book

1. Engineering Physics" by Gaur and Gupta
2. Concepts of Modern Physics" by Arthur Beiser
3. Physics for Scientists and Engineers" by Raymond A. Serway and John W. Jewett.

LAB EXPERIMENTS

Ex. No	Experiment Title	Mapped CO/COs
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	CO2

	with the help of spectrometer.	
4	To determine the specific rotation of cane sugar solution with the help of half shade polarimeter.	CO2
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 center 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

Minor Project-I

Program Name	**B.Tech CSE (Specialization in Cybersecurity)**		
COURSE NAME:	**Course Code**	**L-T-P**	**Credits**

Minor Project-I	**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 CSE (Specialization in Cybersecurity) 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.

K.R. MANGALAM UNIVERSITY

Maker Lab: Tinkering with Technology

Program Name	**B.Tech CSE (Specialization in Cybersecurity)**			
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.

Learning Experiences

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

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.

K.R. MANGALAM UNIVERSITY

Open Elective-I

Course 1: AI & Machine Learning for Everyone

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
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

Course 2: Digital Transformation & Industry 4.0

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
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: Introduction to Digital Transformation & Industry 4.0	No. of hours: 12
Content Summary: • Evolution from Industry 1.0 to 4.0 • Drivers and enablers of digital transformation • Digital maturity and organizational change • Global case studies of digital disruption		
Unit Number: 2	**Title: Technologies Enabling Industry 4.0**	**No. of hours: 13**
Content Summary: • Internet of Things (IoT) and Industrial IoT (IIoT) • Cloud computing and edge computing • Artificial Intelligence and machine learning in industrial settings • Cyber-Physical Systems (CPS) and digital twins		

Unit Number: 3	Title: Smart Manufacturing & Automation	No. of hours: 10
Content Summary: • Smart factories and intelligent production lines • Robotics and automation systems • Additive manufacturing (3D printing) • Role of big data analytics and real-time decision-making		
Unit Number: 4	**Title: Strategy, Implementation & Impact**	**No. of hours: 10**
Content Summary: • Digital transformation roadmap • Change management and digital leadership • Economic, ethical, and workforce implications • National policies and global standards for Industry 4.0		

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:

Schwab, K. (2017). *The Fourth Industrial Revolution*. Crown Business. ISBN: 978-1524758868

Course 3: Cybersecurity Essentials & Digital Privacy

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
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	Make informed decisions about digital safety in personal and professional settings

Course Outline:

Unit Number: 1	Title: Introduction to Digital Transformation & Industry 4.0	No. of hours: 12
Content Summary: • Evolution from Industry 1.0 to 4.0 • Drivers and enablers of digital transformation • Digital maturity and organizational change • Global case studies of digital disruption		
Unit Number: 2	**Title: Technologies Enabling Industry 4.0**	**No. of hours: 13**
Content Summary: • Internet of Things (IoT) and Industrial IoT (IIoT)		

• Cloud computing and edge computing • Artificial Intelligence and machine learning in industrial settings • Cyber-Physical Systems (CPS) and digital twins		
Unit Number: 3	**Title: Smart Manufacturing & Automation**	**No. of hours: 10**
Content Summary: • Smart factories and intelligent production lines • Robotics and automation systems • Additive manufacturing (3D printing) • Role of big data analytics and real-time decision-making		
Unit Number: 4	**Title: Strategy, Implementation & Impact**	**No. of hours: 10**
Content Summary: • Digital transformation roadmap • Change management and digital leadership • Economic, ethical, and workforce implications • National policies and global standards for Industry 4.0		

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:

Schwab, K. (2017). *The Fourth Industrial Revolution*. Crown Business. ISBN: 978-1524758868

K.R. MANGALAM UNIVERSITY

SEMESTER: III

Nand To Tetris-I

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
Course Name: **Nand To Tetris-I**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCNT301	**3-0-2**	4	45
Type of Course:	Major			
Pre-requisite(s): Basic understanding of Boolean algebra, binary number systems, and fundamental programming concepts.				

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.

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

List of Experiments

Ex. No	Experiment Title	Mapped CO/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.	CO 2

	• 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.	
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. • 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.	CO 3
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	CO 3

	program execution (e.g., a tight loop, conditional jump).	
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

K.R. MANGALAM UNIVERSITY

Web Dev -III (Node.js &Express Backend)

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
Course Name: **Web Dev -III (Node.js &Express Backend)**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCWD303	3-0-2	4	45
Type of Course:	DSE			
Pre-requisite(s): Basic knowledge of HTML, CSS, JavaScript, and Web Development I & II (Frontend and introductory backend concepts).				

Course Perspective. This advanced course equips students with hands-on skills in designing, developing, testing, and deploying modern full-stack backend applications. Learners build powerful RESTful APIs using Node.js and Express.js, connect with databases like MongoDB, implement secure authentication, and write production-ready backend code. A special focus is given to integrating cutting-edge AI services using APIs and patterns like RAG (Retrieval-Augmented Generation). The course culminates in deploying applications to cloud platforms using tools like Docker. By the end, students will be job-ready backend developers.

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

COs	Statements
CO 1	Building server-side applications using Node.js and Express.
CO 2	Designing and develop RESTful APIs for web-based solutions.
CO 3	Integrating backend systems with databases for data persistence.
CO 4	Implementing authentication, authorization, and middleware in backend applications.

Course Outline:

Unit Number: 1	**Title: Foundations of Node.js**	**No. of hours: 12**

Contents:

- Overview of Backend Development & HTTP fundamentals
- Introduction to Node.js and event-driven architecture
- JavaScript for Node: ES6+ syntax, async/await, Units
- OOP in JavaScript: constructor functions, classes, this, and prototypes
- Core Node.js Units: fs, http, path, url

Unit Number: 2	**Title: Express.js & REST API Development**	**No. of hours: 10**

Content:

- Introduction to Express.js
- Routing: static/dynamic routes, route parameters
- Middleware concepts: built-in, custom, error-handling
- Controllers, routers, modular structure for scaling apps
- Express Generator and folder best practices
- REST Principles: statelessness, resources, HTTP verbs
- Full CRUD API:
- Part 1 – POST (Create), GET (Read)
- Part 2 – PUT/PATCH (Update), DELETE
- Error handling patterns: try-catch, centralized middleware, status codes
- Building and testing APIs using Postman

Unit Number: 3	**Title: Databases, Authentication & Security**	**No. of hours: 12**

Content:

- Intro to Databases: SQL vs NoSQL, MongoDB setup
- Data modeling with Mongoose or Sequelize
- Schema design, validations, and references
- Querying & Populating: .find(), .populate(), aggregation basics
- Real-time data flow with full database CRUD integration
- Authentication basics: password hashing with bcrypt
- Token-based Auth: JWT setup, signing, and verification
- Protecting routes with role-based middleware
- Environment variables with dotenv and secret management
- Best practices for securing credentials and APIs

Unit Number: 4	Title: AI, Design & Deployment	No. of hours: 11
Content: • Low-Level Design (LLD): modularity, SOLID principles, code reuse • Automated Testing: writing basic unit & integration tests with Jest or Mocha • Integrating Generative AI APIs (e.g., OpenAI, HuggingFace) • Retrieval-Augmented Generation (RAG) for smarter backend features • Docker basics: Dockerfile, image, container, and Docker Compose • Deployment: cloud hosting on Render, Vercel, or Railway • Wrap-up and final review		

Learning Experiences

Inside Classroom

- **Hands-on Lab Sessions**: Build and test RESTful APIs, implement CRUD operations, use middleware, and manage authentication workflows using Node.js and Express.
- **Code Walkthroughs & Debugging**: Analyze existing backend codebases to understand routing, middleware flow, and error handling; practice real-time debugging techniques.
- **Project-Based Learning**: Design and develop full backend systems in stages—routing, database integration, and deployment—guided by instructor feedback.
- **Live Demos**: Explore tools and concepts like Postman, MongoDB Compass, JWT, and session management with live demonstrations and guided examples.
- **Peer Reviews**: Conduct collaborative code reviews to improve backend structure, security, and API design, simulating real-world development practices.

Outside Classroom

- **Open Source Contribution**: Participate in backend-focused GitHub projects to enhance understanding of modular coding and real-world collaboration.
- **Cloud Deployment Practice**: Deploy Node.js applications to platforms like Heroku, Vercel, or AWS; manage environment variables and CI/CD basics.
- **Webinars & Tech Talks**: Attend industry sessions on backend scalability, microservices, API security, and modern backend trends.
- **Capstone Project Development**: Work on a team-based full-stack application integrating frontend and backend, focusing on real-world problem-solving.
- **Portfolio Building**: Document and publish backend projects on platforms like GitHub with README files and API documentation to showcase skills.

Textbooks:

1. Kim, D., & Solomon, M. G. (2022). Fundamentals of Information Systems Security (4th ed.). Jones & Bartlett Learning.

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab 1 – Simple Express API (Unit 1)** • Build a basic Express server with 3–4 endpoints • Use GET and POST methods to serve dummy data • Add custom middleware to log requests • Implement 404 and global error handler Practice Concepts: Routes, status codes, middleware chain, modular routing	CO 2
2	**Lab 2 – CRUD API with MongoDB (Units 2)** • Use Mongoose or Sequelize to model a simple resource (e.g., Products or Tasks)	CO 1

	• Implement CRUD: Create (POST), Read (GET), Update (PUT/PATCH), Delete (DELETE) • Handle validation and errors Practice Concepts: Schema design, .save(), .find(), .findByIdAndUpdate(), .deleteOne()	
3	**Lab 3 : Auth & JWT (Unit 3)** • Add user model with password encryption • Setup login and register routes • Generate JWT on successful login • Protect a private route using middleware Practice Concepts: bcrypt, jsonwebtoken, protected routes, status codes	CO 3
4	**Lab 4 – AI-Powered Endpoint (Unit 4)** • Build a POST API route that integrates a generative AI API • Send user input to the AI and return a formatted response • Handle errors and API keys via .env Practice Concepts: External APIs, fetch/axios, prompt formatting, env vars	CO 2
5	**Capstone Project – Book My Show Full Stack Project** **Objective:** Build and deploy a full-featured Full stack Project **Project Requirements:** • Design a modular Express API with CRUD and JWT auth • Connect to a MongoDB or PostgreSQL database • Add a route powered by an AI API (e.g., summarizer, etc.) • Write at least 3 unit tests • Containerize and deploy the API • Document endpoints using Swagger or simple markdown • Deliverables: Live deployed backend, GitHub repo with code and README, demo video (optional)	CO 4

K.R. MANGALAM UNIVERSITY

Foundations of Cybersecurity

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
Course Name: **Foundations of Cybersecurity**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCYCS305	3-0-2	4	45
Type of Course:	DSE			
Pre-requisite(s): Basic understanding of computer networks, operating systems, and introductory scripting using Python or Linux CLI.				

Course Perspective. This course provides a comprehensive foundation in the principles and practices of cyber security. Students will gain an understanding of core concepts such as confidentiality, integrity, availability, authentication, cryptography, and common vulnerabilities. Through hands-on labs and real-world simulations, learners will work with modern tools to secure systems, detect threats, and defend against cyberattacks. By the end of the course, students will be prepared for entry-level roles in cyber security such as SOC Analyst, Security Tester, or Security Engineer.

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

COs	Statements
CO 1	Explaining core principles and goals of cybersecurity including CIA triad and threat landscapes.
CO 2	Performing basic system hardening and vulnerability assessment using open-source tools.
CO 3	Implementing authentication, encryption, and secure communication protocols.
CO 4	Analysing and simulate common attacks such as phishing, SQL injection, and DoS.
CO5	Applying foundational incident response practices and analyze log data.

Course Outline:

Unit Number: 1	**Title: Introduction to Cyber Security & Threat Landscape**	**No. of hours: 12**
Contents: • Cybersecurity goals: Confidentiality, Integrity, Availability (CIA Triad) • Common threats: malware, phishing, ransomware, insider threats • Types of attackers: script kiddies, hacktivists, nation-state actors • Security domains: Network, Endpoint, Application, Cloud Security • Cyber kill chain & stages of attacks • Risk, vulnerabilities, and exposure • Security policies & governance basics **Hands-On:** • Lab: Map real-world cyber-attacks to the cyber kill chain. • Activity: Analyze recent cybersecurity breaches (e.g., Equifax, SolarWinds). • Create a personal cyber hygiene checklist.		
Unit Number: 2	**Title: System Security & Hardening**	**No. of hours: 10**
Content: • Windows/Linux security fundamentals • User access controls, file permissions • Patch management and system updates • Antivirus and endpoint protection • Firewalls (host-based), secure configurations • Introduction to Virtual Machines and Sandboxing **Hands-On:** • Lab: Harden a Linux virtual machine using user/group permissions and firewall settings. • Install and configure UFW/iptables on Ubuntu. • Identify open ports and services using nmap.		
Unit Number: 3	**Title: Network Security & Cryptography Basics**	**No. of hours: 12**
Content: • Network protocols (TCP/IP, DNS, HTTP, HTTPS) • Public Key Infrastructure (PKI) • Symmetric and Asymmetric encryption (AES, RSA) • Hashing (MD5, SHA-256) • Digital signatures & certificates • VPN and SSL/TLS basics • Wireshark basics and packet analysis **Hands-On:** • Lab: Use Wireshark to analyze HTTP vs HTTPS traffic. • Simulate file encryption and decryption using OpenSSL.		

• Create and validate digital signatures.		
Unit Number: 4	**Title: Web Security, Attacks & Incident Handling**	**No. of hours: 11**
Content: • OWASP Top 10 vulnerabilities (focus: XSS, SQLi, CSRF) • Web application firewalls (WAF) • Authentication flaws & session hijacking • Introduction to logs and SIEM concepts • Basics of threat hunting and forensic analysis • Incident response phases **Hands-On:** • Lab: Simulate XSS and SQLi on a DVWA (Damn Vulnerable Web Application) • Conduct basic log analysis using open-source tools (e.g., Splunk Free or ELK stack) • Build a simple incident report using a mock attack scenario		

Learning Experiences

Inside Classroom

- **Hands-on Lab Sessions:** Implement access control mechanisms, secure network configurations, and simulate malware analysis using virtual machines and security tools (e.g., Wireshark, Nmap).
- **Case Study Discussions:** Examine major cyber incidents such as ransomware attacks or data breaches and evaluate response strategies and preventive controls.
- **Simulated Cyber Attacks:** Perform mock attacks (e.g., SQL injection, phishing) in sandbox environments to understand real-time system vulnerabilities and response mechanisms like IDS and firewalls.
- **Interactive Demos:** Demonstrate tools such as VPNs, encryption software, digital certificates, and password cracking techniques.
- **Peer Learning:** Engage in collaborative tasks where students assume different roles (attacker, defender, analyst) to understand both offensive and defensive perspectives.

Outside Classroom

- **Industry Webinars:** Participate in sessions conducted by cybersecurity professionals on zero-day attacks, bug bounty programs, and evolving threat landscapes.
- **Capture the Flag (CTF) Competitions:** Apply practical knowledge in competitive environments to solve cryptographic, forensic, and web-based security challenges.
- **Cloud Lab Exercises:** Utilize platforms like AWS Educate or Azure to perform IAM configuration, VPC security, encryption of data at rest and in transit.
- **Research and Presentations:** Investigate specific security domains (e.g., IoT security, mobile security) and present findings with risk mitigation proposals.
- **Field Visits:** Tour security operations centers (SOCs), forensic labs, or data centers to observe how real-world organizations implement cybersecurity controls.

Textbooks:

2. Kim, D., & Solomon, M. G. (2022). Fundamentals of Information Systems Security (4th ed.). Jones & Bartlett Learning.

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab 1 (Unit 1)** Title: "Threat Mapping and Breach Analysis" **Sub-Objectives**: • Identify and classify threats using the CIA triad. • Analyze a real breach using news and research articles. • Map the breach to the cyber kill chain stages. • Create a simple risk matrix for the breach. • Present mitigation strategies.	CO 2

2	**Lab 2 (Unit 2)** **Title:** "Hardening a Personal Linux System" **Sub-Objectives**: • Install a Linux VM and update it with the latest security patches. • Configure file permissions and user roles. • Enable and test UFW firewall rules. • Disable unused services and check open ports. • Document all changes and explain their impact.	CO 1
3	**Lab (Unit 3)** **Title:** "Secure Communication and Encryption Basics" **Sub-Objectives**: • Use OpenSSL to generate symmetric keys and encrypt files. • Simulate a public-private key exchange. • Observe HTTPS traffic in Wireshark. • Create a hash and verify file integrity. • Analyze digital certificate details in a browser.	CO 3
4	**Lab 4 (Unit 4)** **Title:** "Vulnerability Testing with OWASP Tools" **Sub-Objectives**: • Set up DVWA in a secure environment. • Perform a basic SQL injection and log the impact. • Simulate an XSS attack and demonstrate prevention. • Analyze access logs to detect the intrusion pattern. • Suggest fixes for vulnerabilities found.	CO 2
5	**Capstone Assignment** **Title:** "Designing and Defending a Small Business Network" **Sub-Objectives**: • Create a network diagram with 2–3 services and security layers. • Configure a Linux-based firewall, harden endpoints, and apply access controls. • Use Wireshark and nmap for scanning and traffic analysis. • Simulate an incident (e.g., failed login brute force), collect logs, and create an incident response report.	CO 4

	• Present a summary of preventive and detective measures applied.	

K.R. MANGALAM UNIVERSITY

Database Management Systems

Programme Name:	**B. Tech CSE (Specialization in Cybersecurity)**			
Course Name: **Database Management System**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCMS304	3-0-2	4	45
Type of Course:	Major			
Pre-requisite(s), if any: Fundamental knowledge of data structures, basic programming concepts, and understanding of file organization.				

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:

<table>
<tr><td>Unit Number: 1</td><td>Title: Database Design & Relational Model</td><td>No. of hours: 12</td></tr>
<tr><td colspan="3">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.</td></tr>
<tr><td>Unit Number: 2</td><td>Title: SQL Queries & Advanced Operations</td><td>No. of hours: 12</td></tr>
<tr><td colspan="3">Content:

▪ SQL Fundamentals: DDL, DML, DCL, TCL Commands.
▪ Complex Queries: Joins (Inner, Outer, Self), Subqueries, Set Operations.</td></tr>
</table>

- 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).

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.

List of Experiments

Ex. No	Experiment Title	Mapped CO/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**	CO 1

	• 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.	
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. Learning Focus: Triggers, atomic transactions, stored procedures, real-time validations.	CO 2
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.	CO 3

	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.	
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). Learning Focus: Nested queries, referential integrity, joins, constraints, user role management.	CO4
5	**CAPSTONE PROJECT: Employee Payroll & Attendance Management System**	CO 4

	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.	

K.R. MANGALAM UNIVERSITY

Design & Analysis of Algorithms

Department:	**B. Tech CSE (Specialization in Cybersecurity)**			
Course Name: Design and Analysis of Algorithms	**Course Code : ETCCDA302**	**L –T-P**	**Credits**	**Contact Hours**
		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).

Learning Experiences

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 Randomised QuickSort (just the standard pivot-randomisation). 2. Search module o Binary Search vs. Hash-Table lookup on the same dataset.	CO 1

	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 behaviour you noticed. *Data provided:* three CSV files (random, nearly-sorted, reverse).	
2	**Greedy vs DP Optimiser** **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 o Implement *Greedy* by value/weight and the exact *DP* solution; compare total value achieved on ≤ 200 items. 2. Huffman Coding o Generate the binary codes for the packing-slip text and output compressed size. 3. Reflection o ½-page note: when did greedy fail and why did DP succeed?	CO 2
3	**Graph Navigator Lite** **Problem statement:** Create a simple bike-route finder for a small city graph (≤ 5 000 nodes). 1. Graph core o Read edge list, store it, and implement BFS to list connected components. 2. Shortest path o Write Dijkstra using a binary heap; output distance & path between two input intersections. 3. Search-assist	CO 3

	○ Build a Trie for street-name auto-completion (max 10 suggestions).	
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.	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. Sub-tasks (tie earlier work together): 1. Package Prioritiser ○ Sort parcels by deadline using the fastest algorithm discovered in Project 1. 2. Box Filler ○ Use your DP knapsack to decide which parcels fit in one ride. 3. Route Maker ○ Build customer graph, run Dijkstra for individual legs, then apply your TSP approximation for overall route.	CO 4

	4. Performance snapshot o One chart comparing runtime of each major step; one paragraph on possible future optimizations.	

K.R. MANGALAM UNIVERSITY

Verbal Ability

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
Course Name: **Verbal Ability**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		2-0-0	2	30
Type of Course:	AEC			
Pre-requisite(s): Basic proficiency in English grammar, vocabulary, and reading comprehension skills.				

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

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**
Co**ntent:** • 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 Testbook, Magoosh, or ReadTheory 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:

- 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/**

K.R. MANGALAM UNIVERSITY

COMPETITIVE CODING-I

Program Name:	**B. Tech CSE (Specialization in Cybersecurity)**			
Course Name: COMPETITIVE CODING-I	**Course Code : AUC001**	**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 (CO)

COs	Statements
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.

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.

K.R. MANGALAM UNIVERSITY

Summer Internship-I

Program Name:	C. Tech CSE (Specialization in Cybersecurity)		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Summer Internship-I	ETCCIN305	0-0-4	2
Type of Course:	INT		

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	**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.

K.R. MANGALAM UNIVERSITY

Community Service

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
Course Name: Community Service	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		1-0-0	1	15
Type of Course:	CS			
Pre-requisite(s): None				

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	**Engaging** students in meaningful social service activities.
CO 2	**Developing** socially responsible engineers
CO 3	**Applying** technical and non-technical skills for the benefit of society.
CO 4	**Fostering** community engagement and support.

Course Outline:

1. **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.

2. **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.

3. **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.

4. **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.

5. **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.

6. **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.

7. **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.

8. **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.

9. **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.

10. **Other Possible Domains**
1. **Environmental Conservation**: Participate in tree planting drives, clean-up campaigns, and conservation projects.
2. **Disaster Relief Support**: Assist in disaster relief efforts, providing aid and support to affected communities.

3. **Animal Welfare**: Volunteer at animal shelters, support animal rescue operations, and promote animal welfare initiatives.
4. **Cultural Preservation**: Work on projects to preserve and promote local cultural heritage and traditions.

11. **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.

12. **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) Specializations in Cybersecurity,
- 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.

K.R. MANGALAM UNIVERSITY

NAND to Tetris-II

Program Name	B. Tech CSE			
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): Successful completion of NAND to Tetris–Part I or equivalent knowledge of logic gates, machine language, and basic computer architecture.				

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

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:

1. 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. • 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.	CO 2

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** • Design phase: propose & storyboard a game or GUI utility (e.g., "Hack-Paint", "Breakout", "2048").	CO 4

	• 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 %.	

Network Security

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
Course Name: **Network Security**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCYNS4025	3-0-2	4	45
Type of Course:	DSE			
Pre-requisite(s): Basic understanding of Network Security, including protocols, firewalls, and secures communication principles.				

Course Perspective. This course provides in-depth, hands-on understanding of network security concepts, architectures, and real-world practices. Students will explore network defense strategies such as firewalls, intrusion detection/prevention systems (IDS/IPS), VPNs, and advanced threat detection techniques. The course includes practice with tools like Wireshark, Snort, Suricata, pfSense, and Packet Tracer to analyze and defend against network attacks. Emphasis is placed on layered network defense and secure design principles for modern enterprise networks.

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

COs	Statements
CO 1	**Explaining** secure network architecture models and layered defense concepts.
CO 2	**Identifying, analysing**, and **mitigating** common network-based attacks and threats.
CO 3	**Configuring** and **managing** firewalls, IDS/IPS, and VPNs using open-source tools.
CO 4	**Analysing** and **monitoring** real-time network traffic for signs of intrusion.
CO5	**Designing** and **simulating** a secure enterprise network using appropriate security technologies.

Course Outline:

<table>
<tr><td>Unit Number: 1</td><td>Title: Secure Network Architectures & Protocols</td><td>No. of hours: 12</td></tr>
<tr><td colspan="3">Contents:
.
• OSI vs TCP/IP security layers
• Threat modeling for network layers
• Secure vs insecure network protocols (Telnet vs SSH, HTTP vs HTTPS)
• Subnetting, NAT, VLAN, DMZs
• AAA: Authentication, Authorization, Accounting
• Security models: Defense in depth, Zero Trust, SDN overview
Hands-On:
• Lab: Design a secure small office network using VLANs and DMZ on Packet Tracer.
• Analyze protocol behavior in Wireshark (HTTP vs HTTPS, Telnet vs SSH).
• Simulate VLAN segmentation and isolate traffic.</td></tr>
<tr><td>Unit Number: 2</td><td>Title: Firewalls, NAT, and Access Control</td><td>No. of hours: 10</td></tr>
<tr><td colspan="3">Content:
• Types of firewalls: packet filtering, stateful, application-layer
• NAT, PAT, port forwarding and its role in security
• IP Tables, pfSense, UFW configuration
• ACLs (Access Control Lists) and rule writing
• Firewall rule design best practices

Hands-On:
• Lab: Set up and configure pfSense to allow/deny specific traffic.
• Simulate firewall rules in iptables or UFW to block/allow services.
• Design ACLs on routers using Cisco Packet Tracer.</td></tr>
<tr><td>Unit Number: 3</td><td>Title: Intrusion Detection, Prevention & Network Monitoring</td><td>No. of hours: 12</td></tr>
<tr><td colspan="3">Content:

• IDS vs IPS concepts
• Signature-based and anomaly-based detection
• Open-source IDS/IPS: Snort, Suricata
• Log monitoring and correlation
• Network behavior analysis and flow monitoring</td></tr>
</table>

- Honeypots and deception technologies

Hands-On:

- Lab: Install and configure Snort/Suricata on a test network.
- Analyze alerts and logs from IDS tools.
- Simulate brute force or port scan attacks and capture alerts.

Unit Number: 4	**Title: VPNs, Wireless Security & Advanced Threat Protection**	**No. of hours: 11**

Content:

- VPN types: IPSec, SSL VPN, site-to-site vs remote access
- Tunneling protocols: L2TP, PPTP, OpenVPN
- Wireless security protocols: WPA2, WPA3, 802.1X
- Rogue AP detection and wireless sniffing
- Advanced network attacks: ARP spoofing, MITM, DNS poisoning
- Security in cloud and hybrid networks (overview)

Hands-On:

- Lab: Set up an OpenVPN server and connect securely.
- Perform a wireless scan using Kismet or Airodump-ng (safe/tested environment).
- Simulate ARP poisoning using tools like Ettercap and mitigate with static ARP entries.
-

Learning Experiences

Inside Classroom

- **Hands-on Lab Sessions:** Configure firewalls, IDS/IPS, and simulate network attacks using tools like Wireshark, Packet Tracer, and Kali Linux.
- **Protocol Analysis Exercises:** Analyze packet-level data to study vulnerabilities in protocols such as TCP, UDP, HTTP, and DNS.
- **Encryption & Decryption Demos:** Demonstrate symmetric/asymmetric encryption techniques using OpenSSL and key management tools.
- **Simulated Network Attacks:** Observe and defend against attacks like ARP spoofing, DoS, and MITM in controlled environments.
- **Collaborative Threat Modeling:** Design secure network topologies and assess risks through peer-based brainstorming and defense planning.

Outside Classroom

- **Virtual Lab Environments:** Practice on platforms like TryHackMe or Cisco NetAcad to reinforce network defense skills.
- **Security Tool Exploration:** Explore advanced tools such as Snort, Suricata, and Wireshark through self-paced assignments.
- **Cybersecurity Awareness Campaigns:** Create and present infographics or short videos on best practices for secure networking.
- **Technical Blog Writing:** Publish short articles or reports analyzing current network-based attacks or protocol flaws.
- **Industry Webinars and Certifications:** Attend online sessions or pursue beginner certifications like Cisco CyberOps or CompTIA Security+.

Textbooks:

1. Kim, D., & Solomon, M. G. (2022). Fundamentals of Information Systems Security (4th ed.). Jones & Bartlett Learning.

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab 1 (Unit 1)** Title: "Designing a Segmented & Secure Office Network" **Sub-Objectives**: • Create a VLAN-segmented network using Packet Tracer. • Simulate traffic isolation using DMZ and firewall placements. • Use Wireshark to validate traffic visibility across VLANs. • Document the design with annotated diagrams.	CO 1
2	**Lab 2**	CO 2

	Title: "Firewall Configuration for Web and Mail Servers" **Sub-Objectives**: • Configure pfSense or iptables to only allow HTTP/HTTPS and SMTP. • Block unused ports and implement port forwarding rules. • Test access from internal and external simulated users. • Log and interpret denied traffic attempts.	
3	**Lab 3** Title: "Intrusion Detection and Alerting using Snort" **Sub-Objectives:** • Install Snort on a Linux VM and configure basic rules. • Generate test traffic: port scans, failed logins, XSS attempts. • Observe alert generation and interpret logs. • Create a summary report of attack types and system responses.	CO 3
4	**Lab 4 (Unit 4)** Title: "VPN Deployment and MITM Defense" **Sub-Objectives**: • Install OpenVPN and securely connect two test machines. • Capture and analyze encrypted vs unencrypted sessions in Wireshark. • Simulate a man-in-the-middle attack and explain its detection. • Discuss defenses like VPN, HTTPS, and static ARP.	CO 3
5	**Capstone Assignment** Title: "Design and Secure an Enterprise Network" **Sub-Objectives:** • Architect a secure network for a 3-department organization. • Implement VLAN segmentation, firewall rules, IDS, and VPN access. • Demonstrate a simulated attack and log detection via IDS.	CO 4

	• Document security policies, network diagram, and configurations. • Justify each defense layer used in terms of cost, complexity, and risk coverage.	

Programme Name:	**B. Tech CSE (Specialization in Cyber Security)**			
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.
CO 4	**Implementing** basic CI/CD pipelines for application build, testing, and deployment.

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

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

Reference Book:

- 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:**	CO 2

	• 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. • 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. • 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	CO 3
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.	CO 4

	• 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 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 Cybersecurity)**			
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.

Unit Number: 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.

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:** • 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).	CO 1

	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:** • 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).	CO 3

	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:** • 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.	CO 4

	• 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.	

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
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): None				

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: • Self Awareness, Personality Concepts (Personality Assessments -Johari Window, Myers & Brigg), Self-Management, Self Esteem, Self-Efficacy, Interpersonal skills, mindset, grit and working in teams.		
Unit Number: 2	Title: Enhancing Reading and Writing Skills	No. of hours: 8
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		
Unit Number: 3	Title: Effective Communication and Public Speaking	No. of hours: 12
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.		
Unit Number: 4	Title: Career Guide and readiness	No. of hours: 6
Content: • Cover Letter, ATS friendly resume, Elevator Pitch, Video Resume (Visume), Networking, Group Discussion, Mock Interviews. Capstone Project .		

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.

Inside Classroom Learning*:*

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/

Open Elective-II

Course 1: Creative Coding & Generative Art

Programme Name:	B.Tech CSE (Specialization in Cybersecurity)			
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 Cybersecurity)**			
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:

Unit Number: 1	**Title: Introduction to HCI and UX**	**No. of hours: 10**
Content: • HCI history, human factors, and cognitive psychology • UX vs UI, usability heuristics (Nielsen's principles) • Affordances, feedback, and user flows		
Unit Number: 2	**Title: User Research and Interaction Design**	**No. of hours: 10**
Content: • Conducting interviews, observations, surveys • Personas, scenarios, and storyboards • Paper prototyping and design sketching		
Unit Number: 3	**Title: Digital Prototyping and Evaluation**	No. of hours: 13
• Wireframing and clickable prototypes (e.g. Figma) • Usability testing and heuristic evaluation • A/B testing and analytics		
Unit Number: 4	**Title: Accessibility, Ethics, and Final Project**	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: 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 Cybersecurity)**			
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 Cybersecurity)**			
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 Cyber Security)**			
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.

K.R. MANGALAM UNIVERSITY

COMPETITIVE CODING -II

Programme Name:	**B. Tech CSE (Specialization in Cybersecurity)**			
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: 18	priority Queue-III	No. of hours: 2

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

Minor Project-II

Program Name	B. Tech CSE (Specialization in Cybersecurity)		
COURSE NAME: **Minor Project-I**	**Course Code**	**L-T-P**	**Credits**
	ETCCPR604	0-0-4	2
TYPE OF COURSE:	Project		
PRE-REQUISITE(S), IF ANY: NA			

Course Perspective:

Minor Project-II for the B.Tech (CSE) specialization in cybersecurity program focuses on deepening the students' practical skills in implementing and deploying cybersecurity solutions to address real-world problems. Building on theoretical foundations and analytical work from earlier semesters, students will design, develop, and test software/hardware systems or prototypes that provide innovative solutions in cybersecurity. This course fosters professional project management, teamwork, technical documentation, and presentation skills while encouraging innovation and adherence to ethical standards in cybersecurity practices. Minor Project-II aims to prepare students for industry challenges and research opportunities by providing hands-on experience with state-of-the-art tools and technologies.

Duration: 12-16 weeks.

Project must focus on following aspects:

Standard Operating Procedure (SOP)

1. **Purpose**

Minor Project-II immerses fourth-semester students in solution design, development, testing, and deployment of cybersecurity projects, translating prior research into tangible artifacts. All project activities—proposal, development, reviews, and assessments—will be managed and audited through Projexa from the 2025-26 session onward, ensuring standardized documentation and transparent evaluation.

2. Scope

- Applies to: All B.Tech CSE (with specialization) students registered for Minor Project-II.
- Excludes: Minor Project-I research and analysis phase (covered previously).
- Duration: 12–14 teaching weeks (one semester block).

3. Learning Outcomes (LO)

LO	Student will be able to…	Evidence Captured by Projexa
LO-1	Design and implement a cybersecurity system or prototype	Design documents + Code repository links
LO-2	Apply testing methodologies and validate solution efficacy	Test cases, results report, and demo video
LO-3	Collaborate and manage project tasks using Projexa tools	Activity logs, milestone completion
LO-4	Prepare professional project documentation and presentations	Final report PDF + slide decks + video demo
LO-5	Demonstrate ethical practices in cybersecurity development	Integrity Ledger entries + ethics declaration

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

Module	Purpose
Team Workspace	Task assignment, progress tracking, mentor chat
Milestone Engine	Proposal → Design Approval → Mid-term Review → Final Demo
Rubric Builder	Digital grading templates for mentors & PEC
Analytics Dashboard	Progress visualization, CO/PO attainment
Integrity Ledger	Submission audits, plagiarism monitoring

5. Roles & Responsibilities

Role	Key Responsibilities	Projexa Permissions
Student Team (2–4)	Design, implement, test, document, present	Upload files, comment, submit
Project Mentor	Guide design & implementation, approve milestones, grade	Approve/reject, grading, notes
Project Evaluation Committee (PEC)	Evaluate all phases, moderate marks, dispute resolution	Rubric scoring, moderation tools
Project Coordinator	Configure deadlines, monitor progress, manage changes	Admin dashboard, deadline override
Dept. Admin	Oversight, export accreditation data	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	Project Proposal (2 pages + basic design)	Feasibility feedback
2	Mentor-Approved	Detailed Design Document (5-7 pages)	Approve design
3-4	—	Prototype / Module Development start	Weekly progress reviews

5	Mid-Review	Mid-term Presentation + Demo video (3-5 min)	Phase B rubric (Mentor 15 / PEC 20)
6-9	—	Development continuation, testing	Mentor inline feedback
10	Draft → Submitted	Draft Final Report + Final Prototype	Mentor feedback, revisions
11	Mentor-Approved	Final report submission readiness	Mark "Ready for Final"
12	Final Review	Final Demo Presentation + Report	Phase C rubric (Mentor 15 / PEC 30)
13	—	Scholarly Output or Competition submission	Phase D score (0–10)
14	Closed	Reflection survey	Grade release, closure

7. Deliverables & Format Standards

Artefact	Mandatory Format	Upload Location
Project Proposal	PDF (Dept template, 2 pp)	Proposal module
Design Document	PDF (5-7 pages, IEEE format)	Docs upload
Mid-term Presentation	PPT/PDF (10-15 slides)	Presentation module
Demo Video	MP4 link (YouTube unlisted/Drive)	Media tab
Final Report	IEEE 2-column PDF (12-15 pp)	Report upload
Scholarly Output	PDF of submission or award certificate	Evidence upload

8. Evaluation Scheme (100 Marks)

Phase	Timing	Total	Mentor	PEC	Criteria (digital rubric)
A Proposal	Week 1-2	20	5	15	Clarity of problem, design feasibility, presentation
B Mid-term	Week 5	35	15	20	Prototype progress, design completeness, testing plans
C Final	Week 12	35	15	20	Final prototype, testing results, report quality, viva
D Scholarly / Outreach	Week 13	10	—	10	Manuscript submission / competition entry / awards
Continuous Effort Modifier	Whole semester	±3	Mentor	—	Consistent effort or non-compliance

Rubrics include 4 performance levels with detailed descriptors in Projexa.

9. Grading & Publication

1. Weighted calculation auto-executes on PEC submission of final rubrics.
2. Students view marks/comments but cannot edit rubrics.
3. 10% random sample second-marked for moderation.
4. Pass requirement: ≥50% overall AND ≥40% in each of Phases A-C.
5. Grades posted to LMS via Projexa API within 72 hours of final review.

SEMESTER: V

K.R. MANGALAM UNIVERSITY

Cryptography

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
Course Name: **Cryptography**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCYCT503	3-0-2	4	45
Type of Course:	DSE			
Pre-requisite(s): Foundational understanding of discrete mathematics, number theory, and basic programming.				

Course Perspective. This course provides a deep understanding of modern cryptographic principles, algorithms, and applications. It covers classical encryption techniques, modern symmetric and asymmetric cryptography, cryptographic hash functions, digital signatures, and secure protocols. Through practical labs using tools like OpenSSL, CrypTool, GPG, and Python-based cryptography libraries, students will develop hands-on experience in implementing and analyzing cryptographic solutions without overlapping content from prior cybersecurity or network security courses.

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

COs	Statements
CO 1	**Understanding** mathematical foundations and classical cipher techniques.
CO 2	**Applying** modern symmetric and asymmetric encryption algorithms securely.
CO 3	**Implementing** and **analysing** cryptographic hash functions and digital signatures.
CO 4	**Using** tools and scripts to encrypt/decrypt data, generate keys, and sign documents.
CO5	**Evaluating** cryptographic protocols for secure communication and storage.

Course Outline:

Unit Number: 1	**Title: Classical Cryptography & Mathematical Foundations**	**No. of hours: 12**

Contents:

- Terminology: plaintext, ciphertext, key, encryption, decryption, cryptanalysis
- Classical ciphers: Caesar cipher, monoalphabetic, polyalphabetic (Vigenère)
- Frequency analysis and brute-force attacks
- Modular arithmetic and congruences
- Euler's theorem, Fermat's little theorem, GCD, multiplicative inverse
- Introduction to number theory concepts relevant to cryptography
- Hands-On:
- Implement Caesar and Vigenère ciphers in Python
- Break classical ciphers using frequency analysis tools
- Solve modular arithmetic problems manually and via scripts

Unit Number: 2	**Title: Symmetric Key Cryptography**	**No. of hours: 10**

Content:

- Block cipher vs stream cipher
- DES: structure, round functions, key schedule
- AES: S-box, MixColumns, ShiftRows, SubBytes, key expansion
- Modes of operation: ECB, CBC, CFB, OFB, CTR
- Common attacks on symmetric systems: padding oracle, replay, key reuse
- Key management challenges

Hands-On:

- Encrypt and decrypt data using AES (Python Cryptography / PyCryptoDome / OpenSSL)

- Compare outputs of ECB vs CBC mode with same data
- Simulate a replay attack and its prevention via nonce/IV

Unit Number: 3	Title: Public Key Cryptography & Key Exchange	No. of hours: 12

Contents:

- Public-key vs private-key systems
- RSA: key generation, encryption, decryption, math background
- Diffie-Hellman Key Exchange: concept, use-case, vulnerabilities
- ElGamal Encryption
- Introduction to Elliptic Curve Cryptography (ECC)
- Certificate Authorities (CAs) and public key infrastructure (PKI)

Hands-On:

- Generate RSA key pairs using OpenSSL and GPG
- Implement RSA encryption and decryption using Python
- Perform Diffie-Hellman key exchange simulation
- Use GPG for signing and encryption

Unit Number: 4	Title: Hash Functions, MACs, Digital Signatures & Protocols	No. of hours: 11

Contents:

- Hash functions: MD5, SHA-1, SHA-2, SHA-3
- Message Authentication Codes (HMAC)
- Birthday attacks and collision resistance
- Digital Signatures: RSA, DSA, ECDSA
- TLS/SSL and secure email (S/MIME, PGP)
- Introduction to zero-knowledge proofs and blockchain use-cases

Hands-On:

- Hash files using SHA-256 and compare outputs
- Generate HMAC for messages and verify integrity
- Digitally sign and verify documents using GPG

- Capture and analyze TLS handshake using Wireshark
-

Learning Experiences

Inside Classroom

- **Hands-on Lab Sessions:** Implement classical and modern encryption algorithms (e.g., Caesar, RSA, AES) using Python or C.
- **Mathematical Walkthroughs:** Step-by-step solving of modular arithmetic, GCD, and Euler's theorem to understand algorithm foundations.
- **Algorithm Simulations:** Visualize cryptographic protocols like Diffie-Hellman key exchange and digital signatures.
- **Group Cipher Challenges:** Collaborate to encrypt and decrypt secret messages using custom or known ciphers.
- **Case Study Discussions:** Analyze real-world cryptographic failures (e.g., MD5 collisions, SSL vulnerabilities) and their consequences.

Outside Classroom

- **Online Cryptography Platforms:** Solve interactive problems on sites like Cryptopals or CyberSecLabs to build applied skills.
- **Research and Presentations:** Explore emerging cryptographic systems like post-quantum cryptography or zero-knowledge proofs.
- **Mini Projects:** Build secure messaging apps or encryption tools using libraries like PyCrypto or OpenSSL.
- **Cryptography in the Real World:** Study how cryptography secures domains like blockchain, digital payments, and secure communication.
- **Guest Talks and Webinars:** Attend sessions by experts on modern trends such as homomorphic encryption or quantum threats.

Textbooks:

1. William Stallings. Cryptography and Network Security: Principles and Practice (8th Edition), Pearson, 2023.

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab 1 (Unit 1)** Title: "Breaking Classical Ciphers" **Sub-Objectives:** • Encrypt a message using Caesar and Vigenère cipher manually and via script • Perform frequency analysis to break ciphertext • Understand cipher weaknesses through decryption attempts	CO 1
2	**Lab 2 (Unit 2)** . Title: "AES Encryption Modes in Practice" S**ub-Objectives:** • Encrypt a file using AES in ECB and CBC mode • Observe and explain patterns in ECB vs CBC encrypted images • Experiment with IV reuse and its impact on security	CO 2
3	**Lab (Unit 3)** Title: "Public-Key Cryptography and Key Exchange" **Sub-Objectives:** • Generate RSA key pairs • Encrypt and decrypt messages using OpenSSL/GPG • Simulate and visualize a Diffie-Hellman key exchange in Python	CO 2

4	**Lab 4 (Unit 4)** Title: "Digital Signatures and Message Integrity" **Sub-Objectives:** • Hash a document and verify integrity using SHA256 • Create and verify a digital signature using GPG • Implement and test HMAC using Python	CO 2
5	**Capstone Assignment** Title: "Design and Implement a Secure Communication Protocol" **Sub-Objectives:** • Design a secure messaging system using AES + RSA + HMAC • Implement encryption, decryption, signing and verification logic • Demonstrate secure key exchange (RSA/DH) in Python • Analyze threat models and prepare a security audit document	CO 3

Operating Systems

Program Name	**B. Tech CSE (Specialization in Cybersecurity)**			
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): Fundamentals of 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 synchronization for efficient, deadlock-free execution
CO 3	**Analyzing** and optimize memory and virtualization performance
CO 4	**Building** fault-tolerant storage and file-system components
CO 5	**Troubleshooting** and defend OS design choices in interview scenarios.

Course Outline:

Unit Number: 1	**Foundations, Process Model, & Command-Line Essentials**	**No. of hours: 10**

- 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.
- Synchronisation 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, Virtualisation & Advanced Administration**	**No. of hours: 10**

- 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: 10**

- 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.
- Network service administration: ss, ip, firewall basics (nftables/ufw), systemd-networkd, basic monitoring with iftop and tcpdump.
- High-impact troubleshooting workflow: log correlation, dmesg, journalctl, resource-pressure-analyser, stress testing with fio and iperf.

Learning Experience:

Inside Classroom Learning

Interactive Lectures

- Use visual diagrams, animations, and analogies to explain core concepts:
    - Process management, scheduling, memory management, file systems, I/O handling.
- Compare how different OS platforms (Linux, Windows, macOS) handle key functions.

Concept Clarity

- Step-by-step breakdowns of complex topics:
    - Deadlock detection/prevention, paging vs segmentation, scheduling algorithms (FCFS, SJF, RR, Priority), system calls.
- Use pseudocode, state transition diagrams, and analogies (e.g., Elevator algorithm for disk scheduling).

Live Demos & Simulations

- **Demonstrate real-time OS behaviour using**:
    - Linux terminal for process creation, file handling, and thread synchronization.
    - Simulators for CPU scheduling, memory allocation, and deadlock detection (e.g., OS Lab tools, EduOS).

Group Work & Case Studies

- Collaborative scenarios such as:
    - Building a mini OS simulation or a custom process scheduler.
    - Case studies of OS crashes, resource contention, or kernel panic.
    - Troubleshooting exercises on zombie processes or thread race conditions.

Problem-Solving Practice

- Practical tasks like:
    - Calculating waiting/turnaround time using different scheduling algorithms.
    - Memory management problems (internal vs external fragmentation).
    - Solutions to synchronization problems using semaphores or monitors.

Outside Classroom Learning

Hands-On Projects

- **Build:**
    - **A basic shell interpreter**
    - **Simulated CPU scheduler or memory manager**
    - **File system navigator in C/C++**

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.

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.

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.	CO 2

	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.	
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)***	CO 4

	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. 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.	

Programme Name:	**B. Tech CSE (Specialization in Cybersecurity)**			
Course Name: Principles and Practices of System Design	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCMD501	3-0-2	4	45
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: 15

Content:

- 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: 12

Content:

- 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: 10**

Content

- 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, Optimizations & Capstone Mock Panels**	**No. of hours: 8**

Content:

- 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 Learning:

Interactive Lectures

- Use ER diagrams, NoSQL vs SQL visual comparisons, and data modeling examples.
- Explain key topics: data normalization, indexing, query optimization, schema design, aggregation pipelines, and MongoDB architecture.
- Demonstrate ORM concepts using Mongoose with visual object-document mapping.

Concept Clarity

- Break down complex concepts like:
    - ACID vs BASE properties
    - Joins vs Document references in NoSQL
    - Population, middleware, and schema hooks in Mongoose
- Use analogies (e.g., library catalog for indexing, postal system for request-response).

Live Demos & Simulations

- **Live code** how MongoDB works using the Mongo Shell and Compass GUI.
- Simulate:
    - Aggregation pipelines

- Complex queries with filters and projections
- Mongoose model interactions in a Node.js project

Group Work & Case Studies

- Collaborate on:
  - Designing a database schema for an e-commerce platform or blog site.
  - Case study: Refactoring a poorly designed MongoDB collection.
  - Review open-source MERN projects and assess data layer implementations.

Problem-Solving Practice

- Exercises on:
  - Query optimization techniques using indexes.
  - Designing normalized and denormalized schemas.
  - Integrating Mongoose models with RESTful APIs.

Outside Classroom Learning

Hands-On Projects

- Projects may include:
  - A blogging platform with comment threads and author models.
  - Inventory management system using nested schemas and aggregates.
  - Admin dashboard with real-time analytics via aggregation pipeline.

Coding Assignments

- Tasks like:
  - Create, Read, Update, Delete (CRUD) operations using Mongoose.

- Build data validators and pre-save middleware in schema.
- Integrate authentication and session storage using MongoDB.

Collaborative Assignments

- Team challenges:
  - Refactor SQL-based app to MongoDB using Mongoose.
  - Build a **modular backend** with reusable model definitions.
  - Write a proposal on **NoSQL schema evolution strategies**.

Text and Reference Book

Full Stack Development with MongoDB and Mongoose" by Mithun Satheesh – *Packt Publishing*

Arithmetic and Reasoning

Program Name	**B. Tech CSE (Specialization in Cybersecurity)**			
Course Name:	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**

Arithmetic and Reasoning		**2-0-0**	**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:

Classroom Learning

Interactive Lectures

- Use **visual aids, number lines, Venn diagrams**, and **real-life analogies** (shopping bills, clocks, distance-time, puzzles).
- Explain core concepts such as **percentages, ratios, averages, profit & loss, number series, coding-decoding**, and **syllogisms**.

Concept Clarity

- Provide **step-by-step solutions** and mental math tricks for simplification.
- Use **logical flowcharts** and **tables** to break down reasoning problems (e.g., seating arrangements, blood relations, directions).
- Explain time-saving techniques like **elimination method** in MCQs.

Live Demos & Simulations

- Use interactive tools or board games to simulate problems:
    - Number puzzles, logical patterns, Sudoku-based inference.
    - Digital tools (Kahoot, Quizizz) for live problem-solving races.

Group Work & Case Studies

- Collaborative problem-solving on:
    - Quantitative aptitude word problems.
    - Logical deduction scenarios (detective-style riddles or real-life tasks).
    - Math relay games for practice with speed and accuracy.

Problem-Solving Practice

- Regular drills on:
    - Time-Speed-Distance, Worktime, Simple/Compound Interest, and Mixture problems.
    - Analytical Reasoning: Puzzles, inequalities, direction sense.

- Practice previous competitive exam questions (SSC, Bank, GRE, etc.).

Outside Classroom Learning

Hands-On Projects

- Create fun prototypes like:
    - A "Math Maze" Game involving logic and arithmetic progression.
    - A Daily Budget Tracker using ratio/percentage concepts.
    - Code a mini quiz app using logical reasoning questions.

Coding Assignments (Optional for Advanced Learners)

- Create **C/Python programs** for:
    - Calculating LCM/HCF, prime numbers, or digit-sum tricks.
    - Solving number series or permutation-combination patterns.

Collaborative Assignments

- Group tasks such as:
    - **Designing aptitude test papers for peers.**
    - **Creating video explainers of commonly misunderstood topics.**
    - **Developing logical board games with instructions and reasoning logic.**

Self-Learning Resources

- **Books**: Quantitative Aptitude by R.S. Aggarwal, Magical Book on Reasoning by B.S. Sijwali.
- **Apps**: Unacademy, Khan Academy, GradeUp, Oliveboard.
- **Websites**: Indiabix, GeeksforGeeks Aptitude, BYJU's.
- **YouTube**: Learners' Habitat, Dear Sir, Study Smart.

Textbooks/Web resources / MOOCs / Magazines/ Journals:

- **https://www.indiabix.com/online-test/aptitude-test/**
- **https://www.geeksforgeeks.org/aptitude-questions-and-answers/**
- **https://www.hitbullseye.com/**

COMPETITIVE CODING -III

Programme Name:	**B. Tech CSE (Specialization in Cybersecurity)**

Course Name: COMPETITIVE CODING -III	Course Code	L-T-P	Credits	Contact Hours
		2-0-0	NIL	30
Type of Course:	Audit/credit Course			
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.		

Text Books:

- *Elements of Programming Interviews (EPI)* – Adnan Aziz, Tsung-Hsien Lee, Amit Prakash.
- *Cracking the Coding Interview* – Gayle Laakmann McDowell Best for coding interview prep with 189 questions, concepts, and system design.

Summer Internship-II

Programme Name:	**B. Tech CSE (Specialization in Cybersecurity)**

Course Name: Summer Internship-II	Course Code	L-T-P	Credits	Contact Hours
	ETCCIN504	0-0-4	2	30
Type of Course: INT	Audit/credit Course			
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:

1. **Industry Internship (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. 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 Cybersecurity)**

Course Name 7: 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 Cybersecurity)**			
Course Name 7: 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	**Designing** 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 (7 hours)	**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**	**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.

K.R. MANGALAM UNIVERSITY

Digital Communication, Personal Branding & Influence

Program Name	**B. Tech CSE (Specialization in Cybersecurity)**			
Course Name 7:	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30

Digital Communication, Personal Branding & Influence				
Type of Course:	VAC- II			
Pre-requisite(s): None				

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:

Unit Number: 1	**Foundations of Digital Communication & Online Presence**	**No.of hours: 8**
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		
Unit Number: 2	**Personal Branding: Discover, Design, Deliver**	**No. of hours: 8**
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		
Unit Number: 3	**Designing Tech-Life Routines & Energy Management**	**No. of hours: 8**
Content: **Content Strategy, Storytelling & Influence** • 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** • Hands-on: Write a blog/article or record a 2-min personal pitch video		
Unit Number: 4	**Growing Digital Influence & Engagement**	**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 Cybersecurity)**			
Course Name 7:	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**

Purposeful Living and Ikigai: Designing a Meaningful Life	**VAC**	2-0-0	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	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**	**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**	**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**	**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 Cybersecurity)			
Course Name 7: 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): None				

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	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**	**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**	**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	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 Cybersecurity)**

Course Name 7: 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): None				

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	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**	**No. of hours: 8**
Content: • Empathy vs. sympathy; active listening skills • Compassion-based practices (loving-kindness meditation) • Emotional contagion and boundary management • Role-play: Difficult conversations with emotional awareness • Activity: "Empathy map" for a friend or classmate		
Unit Number: 4	**Designing a Mindful Life & Capstone Practice**	**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 Cybersecurity)**

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): None				

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	**Applying** 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**	**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**	**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	**Empathy, Compassion, and Relational Intelligence**	**No. of hours: 8**

Content:

- **Visual & Video Branding with AI Tools**
- 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**	**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:

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

K.R. MANGALAM UNIVERSITY

Ethical Hacking and Penetration Testing

Program Name	**B. Tech CSE (Specialization in Cybersecurity)**			
Course Name: Ethical Hacking and Penetration Testing	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCYPT602	3-0-2	4	45
Type of Course:	DSE			
Pre-requisite(s): Basic understanding of computer networks, operating systems, and command-line interfaces (Linux/Windows).				

Course Perspective. This course provides hands-on training in ethical hacking and penetration testing techniques used to assess and secure computer systems. Students will simulate cyberattacks in controlled lab environments and learn reconnaissance, vulnerability analysis, exploitation, and reporting methodologies aligned with EC-Council's CEH v12 framework. Emphasis is placed on real-world tools such as Nmap, Burp Suite, Metasploit, Wireshark, and Kali Linux.

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

COs	Statements
CO 1	Performing reconnaissance and footprinting using ethical methods and tools.
CO 2	Identifying, exploiting, and documenting system vulnerabilities through penetration testing.
CO 3	Utilizing industry-standard tools (e.g., Metasploit, Wireshark, Burp Suite) in a lab environment.
CO 4	Illustrating vulnerability assessments, write structured pen test reports, and recommend mitigations.
CO5	Performing reconnaissance and footprinting using ethical methods and tools.

Course Outline:

Unit Number: 1	**Title: Reconnaissance, Footprinting & Scanning**	**No. of hours: 10**

Contents:

- Introduction to ethical hacking, legal and ethical aspects (Computer Misuse Act, GDPR)
- Phases of penetration testing: Recon, Scan, Exploit, Report
- Passive vs active reconnaissance: DNS, WHOIS, social media profiling
- Scanning techniques: TCP/UDP scanning, SYN scans, stealth scans
- Tools: Nmap, Netcat, Shodan, Maltego
- Identifying live hosts, OS fingerprinting, service enumeration

Hands-On:

- Use Nmap and Shodan to identify open ports and services on a target
- Map network architecture and detect live hosts
- Document a passive footprinting report on a test domain

Unit Number: 2	**Title: Vulnerability Analysis & Exploitation**	**No. of hours: 12**

Content:

- Vulnerability types: OS, web app, network, misconfiguration
- Exploitation basics: Buffer overflow, privilege escalation, remote code execution
- Tools: Nessus, Nikto, OpenVAS, ExploitDB
- Metasploit Framework: structure, payloads, modules
- Client-side attacks and privilege escalation
- Post-exploitation basics: password dumping, persistence, data exfiltration

Hands-On:

- Scan a vulnerable VM (e.g., Metasploitable) using Nessus/OpenVAS
- Exploit vulnerabilities using Metasploit
- Perform privilege escalation on a compromised machine

Unit Number: 3	**Title: Web Application & Wireless Network Exploitation**	**No. of hours: 11**

Contents:

- WASP Top 10 vulnerabilities: SQLi, XSS, CSRF, SSRF, IDOR, Broken Auth
- Manual and automated testing of web applications
- Wireless security: WEP/WPA/WPA2 cracking, rogue AP attacks
- Tools: Burp Suite, OWASP ZAP, SQLmap, Aircrack-ng, Wireshark
- Session hijacking, cookie poisoning, SSL stripping

Hands-On:

- Exploit a test web app using SQLmap and Burp Suite
- Conduct MITM and packet sniffing attacks on Wi-Fi in lab sandbox
- Crack WPA2 handshake in a Kali lab environment

Unit Number: 4	**Title: Penetration Testing Process & Reporting**	**No. of hours: 12**

Contents:

- Rules of engagement and scoping a penetration test
  - Writing a structured penetration testing plan
  - Risk rating and vulnerability classification (CVSS)
- Reporting methodology: executive vs technical summaries
- Mitigation recommendations and validation
- Introduction to bug bounty, red teaming, and certification tracks (CEH, OSCP)

Hands-On:

- Execute an end-to-end penetration test on a CTF-style target lab
- Prepare a full technical report with risk classifications and remediation
- Present pen test findings in a simulated stakeholder briefing

Learning Experiences

Inside Classroom

- **Hands-on Lab Sessions:** Identify and fix common web vulnerabilities such as SQL injection, XSS, CSRF, and broken authentication using tools like OWASP Juice Shop and DVWA.
- **Secure Coding Practices:** Practice writing secure code in languages like Java, Python, or PHP to mitigate security flaws.
- **Live Vulnerability Demos:** Demonstrate exploitation techniques and defenses in real-time using Burp Suite and browser dev tools.

- **Code Review Activities:** Analyze sample code for security flaws and recommend fixes in peer-review style sessions.
- **Case Study Discussions:** Investigate real-world application breaches (e.g., Equifax, Facebook) to understand causes and mitigations.

Outside Classroom

- **OWASP Top 10 Exploration:** Independent research and presentation tasks based on each OWASP Top 10 security risk.
- **Bug Bounty Simulation:** Participate in simulated bug hunting contests to find and report vulnerabilities in test environments.
- **Security Tool Exploration:** Use static and dynamic analysis tools like SonarQube, Fortify, and ZAP through guided tutorials.
- **Open Source Contribution:** Contribute to secure coding or documentation in GitHub projects promoting application security.
- **Industry Talks & Webinars:** Attend expert-led sessions on DevSecOps, threat modeling, and secure software development lifecycle (SSDLC).

Textbooks:

1. Dafydd Stuttard & Marcus Pinto, The Web Application Hacker's Handbook, Wiley, 2011.
2. Tanya Janca, Alice and Bob Learn Application Security, Wiley, 2020.

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab 1 (Unit 1)** Title: Footprinting and Reconnaissance of a Simulated Target **Objectives**: • Use DNSdumpster, Nmap, and Maltego • Perform port scanning, OS detection • Document network map and footprinting data	CO 1

2	**Lab 2 (Unit 2)** . Title: Vulnerability Scanning and Exploitation **Sub-Objectives:** • Run Nessus/OpenVAS on vulnerable VM • Exploit known CVEs using Metasploit • Simulate privilege escalation on Linux system	CO 2
3	**Lab (Unit 3)** Web App Penetration on DVWA **Sub-Objectives:** • Perform SQL injection, XSS, and CSRF attacks • Use Burp Suite for session analysis and fuzzing • Document vulnerabilities and test fix validation	CO 3
4	**Lab 4 (Unit 4)** Title: Full Penetration Test Simulation **Sub-Objectives:** • Develop pen test plan, perform scans, exploit, and generate report • Rate risks using CVSS • Present findings as if to a security team	CO 3
5	**Capstone Assignment** Title: Realistic End-to-End Penetration Test on a Virtual Corporate Network **Sub-Objectives:** • Perform reconnaissance, vulnerability scanning, exploitation, post-exploitation • Combine tools: Nmap, Metasploit, Wireshark, SQLmap, Aircrack-ng • Prepare a professional-grade pen test report with screenshots, risk ratings, and mitigations.	CO 4

Cloud Security

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
Course Name: Cloud Security	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCYCS605	3-0-2	4	45
Type of Course:	DSE			
Pre-requisite(s): Fundamental knowledge of cloud computing concepts, networking, and virtualization technologies.				

Course Perspective. This course equips students with the practical knowledge and skills to secure cloud environments across platforms such as AWS, Azure, and GCP. It covers cloud security architecture, IAM, data protection, threat detection, compliance frameworks, and secure deployment strategies using industry tools. Through hands-on labs and projects, students will simulate and remediate real-world cloud security issues.

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

COs	Statements
CO 1	**Understanding** cloud computing models and shared responsibility in security.
CO 2	**Implementing** IAM, encryption, and network security across cloud services.
CO 3	**Detecting** misconfigurations and respond to threats in cloud infrastructure.
CO 4	**Designing** and **auditing** secure cloud architectures for enterprise deployments.

Course Outline:

Unit Number: 1	Title: Cloud Architecture & Security Fundamentals	No. of hours: 12

Contents:

- Cloud deployment models: Public, Private, Hybrid, Multi-cloud
- Service models: IaaS, PaaS, SaaS and their security implications
- Shared Responsibility Model (AWS, Azure, GCP)
- Overview of threats in the cloud: misconfiguration, data loss, privilege escalation
- Introduction to Cloud Security Frameworks: NIST 800-53, ISO 27017, CIS Benchmarks
- DevSecOps and CI/CD overview in cloud contexts

Hands-On:

- Create a basic secure AWS/Azure environment using free-tier services
- Explore shared responsibility via real cloud documentation and cases
- Analyze common breach reports (e.g., Capital One, Uber) to identify architectural flaws

Unit Number: 2	Title: Identity, Access & Data Security in Cloud	No. of hours: 10

Content:

- Identity and Access Management (IAM): Users, Groups, Roles, Policies
- Federation and SSO: SAML, OIDC, MFA
- Encryption techniques: Data at rest, in transit, and envelope encryption
- Cloud-native Key Management Services (KMS)
- Fine-grained access controls (RBAC, ABAC)
- IAM misconfigurations and privilege escalation prevention

Hands-On:

- Create IAM policies and test least privilege access
- Configure encryption of storage services (e.g., S3/GCS/Blob)
- Rotate KMS keys and audit encryption logs

Unit Number: 3	Title: Infrastructure & Network Security in Cloud	No. of hours: 11

Contents:

- Securing virtual machines and containers: hardening, patching, image scanning

- Virtual networks, VPCs, subnets, and routing
- Firewalls, security groups, and NACLS
- Cloud-native monitoring and logging: AWS CloudTrail, Azure Monitor, GCP Audit Logs
- Securing storage (S3/GCS) with bucket policies, lifecycle rules, versioning
- Infrastructure-as-Code (IaC) scanning and secure templates

Hands-On:

- Harden a cloud VM (e.g., Ubuntu on AWS EC2)
- Set up a secure VPC with subnets, firewall rules, and restricted inbound/outbound access
- Scan Terraform/Azure ARM templates with tools like tfsec or Checkov

Unit Number: 4	**Title: Threat Detection, Compliance & Secure Cloud Architecture**	**No. of hours: 12**

Contents:

- Threat detection and response tools: GuardDuty, Azure Defender, Chronicle, CloudSOC
- Cloud incident response: playbooks, response automation
- Compliance standards: GDPR, HIPAA, PCI-DSS, SOC 2 in cloud
- Cloud Security Posture Management (CSPM)
- Zero Trust security model in cloud
- Designing secure architecture patterns for real-world use cases

Hands-On:

- Simulate IAM brute force detection with AWS GuardDuty
- Perform a compliance audit using AWS Config or Azure Policy
- Design a secure architecture for a 3-tier web application

Learning Experiences

Inside Classroom

- **Hands-on Lab Sessions:** Configure identity and access management (IAM), security groups, and encryption settings on platforms like AWS, Azure, or GCP.
- **Cloud Threat Simulations:** Simulate cloud-specific attacks such as misconfigured buckets, privilege escalation, and API abuse.
- **Live Demonstrations:** Demonstrate the use of cloud security tools like AWS Shield, Azure Security Center, and CloudTrail.

- **Security Architecture Design:** Design secure cloud architectures considering multi-tenancy, data isolation, and regulatory compliance.
- **Case Study Discussions:** Analyze cloud data breaches and assess what security controls failed or were missing.

Outside Classroom

- **Cloud Security Labs:** Complete guided labs on platforms like AWS Educate, Google Cloud Skills Boost, or Microsoft Learn.
- **Policy Drafting Activities:** Create cloud usage and security policies for hypothetical organizations, focusing on data protection and compliance.
- **Webinars by Cloud Security Experts:** Attend sessions covering topics such as zero trust, shared responsibility, and cloud-native security tools.
- **Cloud Certifications Practice:** Engage in self-paced learning for certifications like AWS Certified Security – Specialty or Microsoft SC-900.
- **Mini Projects:** Implement end-to-end secure cloud applications with logging, monitoring, and threat detection.

Textbooks:

1. Tim Mather, Subra Kumaraswamy, and Shahed Latif. Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance. O'Reilly Media.
2. Stuart Scott. AWS Certified Security – Specialty Exam Guide. Packt Publishing.
3. Microsoft Azure Security Documentation. Microsoft Press (online)

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab 1 (Unit 1)** Title: Shared Responsibility and Threat modelling in Cloud **Sub-Objectives:** • Set up cloud environment (AWS/Azure) • Map security responsibilities for a given application	CO 1

	• Perform threat modeling using STRIDE on a cloud-based app	
2	**Lab 2 (Unit 2)** . Title: IAM and Encryption Controls for Cloud Storage **Sub-Objectives:** • Create IAM policies with granular permissions • Encrypt cloud storage (e.g., S3 bucket or Azure Blob) • Configure key rotation and audit access logs	CO 2
3	**Lab 3 (Unit 3)** Title: Secure Deployment of Compute and Network Infrastructure **Sub-Objectives**: • Launch a hardened EC2/GCE VM • Implement secure firewall/security group configuration • Scan IaC scripts for misconfigurations	CO 3
4	**Lab 4 (Unit 4)** Title: Threat Detection and Compliance Automation **Sub-Objectives:** • Enable and test GuardDuty or Azure Security Center • Simulate suspicious activities (e.g., port scanning or API misuse) • Run a compliance audit and generate a report	CO 4
5	**Capstone Assignment** Title: Designing and Auditing a Secure Cloud Architecture for a FinTech Application **Sub-Objectives:** • Architect a secure 3-tier cloud app using IaaS and PaaS • Implement IAM, encryption, monitoring, and CSPM controls • Simulate threats, perform audits, and document risk mitigation	CO 4

Programme Name:	**B. Tech CSE (Specialization in Cybersecurity)**			
Course Name: **Next-Gen Software Engineering with Agile Frameworks**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCAF601	3-0-2	4	45
Type of Course:	Major			
Pre-requisite(s), if any: Solid programming and computer science foundation, understanding of software development tools, and knowledge of Agile principles and team collaboration frameworks like Scrum or Kanban.				

Course Perspective: This course provides a hands-on approach to modern Software Engineering principles, focusing on Agile methodologies, DevOps, and secure software development. It covers fundamental software engineering concepts, software project management, software development models, and modern industry practices. Students will learn practical aspects of software development, including requirement analysis, Agile project management, software architecture, continuous integration/deployment (CI/CD), containerization, automated testing, and security best practices. The course emphasizes industry-relevant tools such as GitHub Actions, Jenkins, Docker, and Kubernetes.

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

COs	Statements
CO1	**Understanding** fundamental software engineering principles, the software development life cycle (SDLC), and the role of project management in large-scale software systems.
CO2	**Applying** Agile methodologies such as Scrum and Kanban to plan, manage, and execute real-world software development projects.
CO3	**Designing**, **implementing**, and evaluate secure and maintainable software systems using industry-standard coding and architecture practices.
CO4	**Utilizing** automated testing strategies including unit, integration, and functional testing to ensure software reliability and quality.

Course Outline:

Unit Number: 1	**Software Engineering Foundations & SDLC**	**No. of hours: 15**

Content:

- Introduction to Software Engineering: Principles and Practices
- Software Development Life Cycle (SDLC) Models: Waterfall, Iterative, Spiral, V-Model
- Requirements Engineering & Use Case Modeling
- Software Design Principles: SOLID, Coupling, Cohesion, Modularity
- Introduction to Estimation: Function Points, Story Points
- Introduction to Project Management Tools: Gantt Charts, Milestone Planning

Real-World Scenario:

- A startup is preparing to build its first SaaS product. You are part of the core engineering team responsible for defining requirements, selecting the development model, and preparing a timeline. You must analyze trade-offs between different SDLC models, estimate project effort, and document key design principles.

Unit Number: 2	**Agile Methodologies & Project Workflows**	**No. of hours: 12**

Content:

- Agile Manifesto and Core Principles
- Scrum Framework: Roles, Ceremonies, Artifacts
- Kanban vs Scrum: When and How to Use
- User Stories, Acceptance Criteria, Definition of Done
- Agile Estimation Techniques: Planning Poker, Velocity
- Tools: Trello, GitHub Projects, Jira (basics)

Real-World Scenario:

You join a fintech company that is shifting to Agile development. As part of the onboarding project, you are assigned to a Scrum team. You must write epics and

user stories for a new feature, participate in sprint planning, and help your team track velocity to meet the product roadmap goals.

Unit Number: 3	Software Design, Architecture & Secure Development	No. of hours: 10

Content:

- Layered Architecture, MVC Pattern
- Introduction to Software Refactoring and Clean Code Principles
- Common Vulnerabilities (SQL Injection, XSS, Hardcoded Secrets)
- Secure Coding Practices: Input Validation, Authentication, Error Handling
- Threat Modelling and Risk Assessment
- Managing Sensitive Data (Storage, Encryption Basics)

Real-World Scenario:

- You are tasked with reviewing an e-commerce application after a security breach. You must identify insecure coding practices, implement input validation and secure login, and refactor business logic using clean code principles. Your report will be used to brief the CTO on compliance and future risk mitigation.

Unit Number: 4	Testing, Software Quality & Assurance	No. of hours: 08

Content:

- **Topics Covered:**
- Introduction to Software Testing: Levels and Types
- Software Quality Attributes: Reliability, Maintainability, Usability, Portability, Efficiency
- Test-Driven Development (TDD) Concepts
- Unit Testing and Integration Testing: JUnit, PyTest, Mocha
- Functional Testing: Selenium, Postman
- Code Coverage, Test Reports, and Quality Metrics
- Static and Dynamic Code Analysis
- Role of QA in the SDLC, Verification vs. Validation, Defect Lifecycle

Real-World Scenario:

- You are working for a Healthtech company developing a medical records platform. Before launch, your QA team must ensure the software meets industry standards for quality, including performance, security, and maintainability. Your team is responsible for writing automated test cases, conducting regression testing, and generating quality metrics reports for regulatory approval.

Classroom Learning Experience

Inside Classroom Learning:

- **Agile Framework Workshops:** Core Agile methodologies (Scrum, Kanban, XP) are introduced through interactive role-playing sessions, where students enact Sprint Planning, Daily Scrums, Reviews, and Retrospectives using real product backlog examples.
- **User Story Development and Estimation:** Students learn to write effective user stories with INVEST criteria and practice estimation using Planning Poker and T-shirt sizing, guided by real-world case studies.
- **Version Control and Collaboration Labs:** Git basics and branching strategies (feature, develop, release) are taught using GitHub/GitLab in collaborative exercises simulating team-based development workflows.
- **Software Design Principles in Action:** Concepts like SOLID, DRY, and KISS are explained using group-based code reviews and refactoring tasks in object-oriented languages.
- **Unit Testing and TDD Sessions:** Students write unit tests using frameworks like JUnit or PyTest, applying Test-Driven Development (TDD) to build small modules, with peer evaluation and coverage analysis.

Outside Classroom Learning Experience

- **Agile Project Simulations:** Teams build functional software over iterative sprints, using tools like Jira or Trello to manage user stories, backlogs, sprint goals, and burndown charts. Each sprint ends with a demo and retrospective.
- **Continuous Integration & Deployment (CI/CD) Labs:** Students set up pipelines using GitHub Actions or Jenkins, integrating automatic testing, code linting, and deployment to sandbox environments (e.g., Heroku or Netlify).
- **Code Collaboration and Merge Conflict Resolution:** Real-time pair programming sessions using VS Code Live Share or Git merge conflict scenarios simulate real-world developer collaboration challenges.
- **Customer Feedback Integration:** Students act on simulated customer feedback during sprint reviews, adjusting product backlog and sprint planning accordingly to improve responsiveness and adaptability.
- **DevOps and Agile Integration Activities:** Learners explore how Agile connects with DevOps by experimenting with containerization (Docker), environment configuration, and monitoring basics to support rapid delivery and feedback loops.

Text and Reference Book

- Sommerville, I. (2022). *Software Engineering* (11th ed.). Pearson.
- Pressman, R. S. (2019). *Software Engineering: A Practitioner's Approach* (9th ed.). McGraw-Hill.
- Feathers, M. (2004). *Working Effectively with Legacy Code.* Prentice Hall.

List of Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab Task 1: Software Requirements & SDLC Modeling** (Aligned with Module 1 – Software Engineering Foundations & SDLC) **Real-World Scenario:**	CO1

	You've joined a new SaaS startup that's building a project management tool for remote teams. Your task is to document requirements, plan the development timeline, and model the software design flow using SDLC principles. **Objectives:** • Define functional and non-functional requirements for the application. • Create use case diagrams and system models using UML. • Design a project timeline using Gantt charts or milestone charts. • Document SDLC model selection (Agile/Iterative) with justification. **Tools:** Draw.io / Lucidchart, MS Project / GanttProject, Word or Markdown for documentation.	
2	**Lab Task 2: Agile Planning & Sprint Simulation** (Aligned with Module 2 – Agile Methodologies & Project Workflows) **Real-World Scenario:** You're part of a Scrum team working on a personal finance tracker app. You'll simulate a full sprint cycle, from user story writing to sprint review, using Agile tools. **Objectives:** • Write Epics and break them down into User Stories with acceptance criteria. • Estimate stories using Planning Poker and calculate team velocity. • Conduct Sprint Planning and simulate a Daily Stand-up. • Use Jira / Trello to create and manage the product backlog. **Tools:** Jira / Trello / GitHub Projects, Google Docs / Miro for planning.	CO2

3	**Lab Task 3: Secure Code Refactoring & Vulnerability Detection** (Aligned with Module 3 – Software Design, Architecture & Secure Development) **Real-World Scenario:** You're assigned to fix a bug-ridden login module in an existing app that fails secure login and exposes session information. You need to clean the codebase, patch security gaps, and improve maintainability. **Objectives:** • Refactor an insecure or poorly structured codebase using Clean Code principles. • Implement form input validation and error handling. • Mitigate OWASP Top 10 issues like SQL Injection/XSS in provided snippets. • Write documentation outlining applied security best practices. **Tools:** Visual Studio Code / IntelliJ / PyCharm, ESLint / SonarLint / Bandit.	CO3, CO4
4	**Lab Task 4: Software Testing & Quality Analysis (Aligned with Module 4 – Testing, Software Quality & Assurance)** **Real-World Scenario:** You're working in a healthcare app QA team. You must write unit tests for the appointment module and automate functional tests for its REST API endpoints while ensuring the app meets key quality metrics. **Objectives:** • Write unit and integration tests using a testing framework (e.g., JUnit, Mocha, PyTest). • Automate functional tests using Selenium or Postman. • Measure code coverage using tools like Istanbul or Coverage.py.	CO4

	• Evaluate software quality using metrics such as maintainability index and defect density. **Tools:** JUnit / Mocha / PyTest, Postman / Selenium, Code Coverage Tools.	
5	**Capstone Project:** Agile-Driven Development & QA for a Mini Product **Rel-World Scenario:** You will design, plan, and quality-assure a real-world mini product (e.g., Event Manager App, Helpdesk Ticketing System, College Placement Portal) following Agile practices from sprint planning to secure development and automated testing. **Objectives:** • Define requirements and create a product backlog with user stories. • Conduct 1–2 Agile sprints with sprint reviews and retrospectives. • Implement secure, modular code following industry design practices. • Write unit, integration, and API tests, and generate test coverage reports. • Present a working demo, sprint summary, and QA report. **Tools:** GitHub, Jira/Trello, Visual Studio Code, Postman, PyTest/JUnit, CI tool (optional).	CO4

Computer Networks

Programme Name:	**B. Tech CSE (Specialization in Cybersecurity)**			
Course Name: Computer Networks	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCCN603	3-0-2	4	45
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	**Analysing** 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: 10

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: 12

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: 12

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: 11**

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:

- **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.

- **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.
- **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.
- **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

- **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.
- **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.
- **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.
- **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.
- **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

- Kurose, J. F., & Ross, K. W. – *Computer Networking: A Top-Down Approach*, Pearson.
- Forouzan, B. A. – *Data Communications and Networking*, McGraw-Hill.
- 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 • 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)	CO1

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	CO4

	network for departments like Admin, Library, and Labs. The network must support dynamic IP, secure HTTP access, proper routing, and subnetting across departments. **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 Cybersecurity)**			
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: None				

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 Cybersecurity)**			
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

K.R. MANGALAM UNIVERSITY

Value Added Course (VAC)-III

Product Deployment & Growth Hacking

Program Name	**B. Tech CSE (Specialization in Cybersecurity)**			
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): None				

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 Cybersecurity)**			
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): None				

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 Cybersecurity)**			
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): None				

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

K.R. MANGALAM UNIVERSITY

Digital Entrepreneurship: Monetizing Skills in the Creator Economy

Program Name	**B. Tech CSE (Specialization in Cybersecurity)**			
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): None				

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

K.R. MANGALAM UNIVERSITY

Design Thinking for Social Innovation

Program Name	**B. Tech CSE (Specialization in Cybersecurity)**			
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

K.R. MANGALAM UNIVERSITY

No-Code & Low-Code Product Development

Program Name	**B. Tech CSE (Specialization in Cybersecurity)**			
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 Cybersecurity)**			
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): None				

Course Perspective. This course introduces students to the foundational frameworks of Indian logic (Nyāya) 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

K.R. MANGALAM UNIVERSITY

SEMESTER: VII

Digital Forensics and Incident Response

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
Course Name: **Digital Forensics and Incident Response**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCYIS703	3-0-2	4	45
Type of Course:	DSE			
Pre-requisite(s): Basic knowledge of operating systems, computer networks, and file systems, along with familiarity with cybersecurity fundamentals.				

Course Perspective. This course introduces students to core principles, tools, and hands-on techniques in digital forensics and incident response. Students will learn to collect, analyze, and preserve digital evidence from compromised systems and networks. The course blends investigative forensics, malware analysis, and incident response workflows using industry-standard tools like Autopsy, FTK Imager, Wireshark, Volatility, and Splunk. Students will simulate forensic cases involving data breaches, insider threats, malware, and ransomware to build real-world readiness.

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

COs	Statements
CO 1	Understand legal, ethical, and procedural foundations of digital forensics.
CO 2	Performing forensic acquisition, preservation, and analysis of digital media.
CO 3	Detecting and investigating security incidents across networks, hosts, and memory.
CO 4	Applying industry tools and frameworks (e.g., MITRE ATT&CK, NIST IR lifecycle) in simulated incident response scenarios.

Course Outline:

Unit Number: 1	Title Introduction to Digital Forensics & Incident Response	No. of hours: 12

Contents:

- Overview of digital forensics and incident response
- Types of forensics: disk, memory, network, mobile, cloud
- Chain of custody and legal considerations (Indian IT Act, GDPR, ECPA)
- NIST Incident Response Lifecycle: Preparation, Detection, Containment, Eradication, Recovery, Lessons Learned
- Incident types: malware outbreak, unauthorized access, insider threat, DDoS
- Introduction to forensic imaging & write blockers

Hands-On:

- Create a forensic lab VM (Kali, SIFT, or REMnux)
- Use FTK Imager to acquire and hash forensic images
- Document evidence handling logs and case notes

Unit Number: 2	Title: Disk & File System Forensics	No. of hours: 12

Content:

- File systems overview: NTFS, FAT32, ext4
- Windows artifacts: Registry, prefetch, log files, recycle bin, shellbags
- File carving and metadata analysis
- Timestamps and MAC times analysis
- Hidden partitions and anti-forensics
- Tools: Autopsy, Sleuth Kit, FTK Imager, X-Ways (demo)

Hands-On:

- Analyze a Windows image using Autopsy
- Recover deleted files and extract timeline
- Investigate browser history, USB usage, user logins

Unit Number: 3	Title: Memory & Network Forensics	No. of hours: 12

Contents:

- Memory acquisition and volatility of artifacts
- Malware behavior, processes, DLL injection, hooks
- Introduction to Volatility framework: pslist, netscan, malfind
- Network evidence: PCAP analysis, C2 traffic, DNS tunneling, anomalies
- Wireshark filters, protocol dissection
- Detecting beaconing, password sniffing, MITM

Hands-On:

- Dump RAM using DumpIt or winpmem
- Analyze memory with Volatility to detect malware
- Analyze PCAP of malware communication using Wireshark

Unit Number: 4	Title: Malware Analysis & Incident Response Workflow	No. of hours: 9

Contents:

- Static and dynamic malware analysis (intro level)
- Sandboxing tools: Any.Run, Cuckoo (demo walkthroughs)
- IOC (Indicator of Compromise) extraction
- Incident triage: MITRE ATT&CK mapping
- IR documentation and reporting format
- Threat intelligence basics (VirusTotal, Hybrid Analysis, AbuseIPDB)

Hands-On:

- Analyze ransomware behavior from sandbox logs
- Map attack to MITRE ATT&CK using IOC evidence
- Build a full incident response report with containment and recovery plan

Learning Experiences
Inside Classroom

- **Hands-on Lab Sessions:** Perform forensic imaging, metadata extraction, and file recovery using tools like Autopsy, FTK Imager, and Sleuth Kit.
- **Incident Simulation Exercises:** Analyze simulated cyber incidents such as malware infections or unauthorized access, and document response steps.
- **Log File Analysis:** Interpret system, application, and network logs to trace attacker activities and identify Indicators of Compromise (IoCs).
- **Chain of Custody Practice:** Learn proper procedures for evidence handling and documentation to maintain legal admissibility.
- **Case Study Discussions:** Examine real-world forensic investigations and major breach incidents to understand investigative techniques and legal considerations.

Outside Classroom

- **Open-Source Tool Exploration:** Practice using popular forensic and incident response tools like Volatility, Wireshark, and X-Ways in sandbox environments.

- **CTF and Forensic Challenges:** Participate in digital forensics challenges to analyze disk images, memory dumps, or PCAP files.
- **Report Writing Exercises:** Draft forensic investigation reports with proper structure, timeline reconstruction, and conclusions.
- **Guest Lectures/Webinars:** Attend talks by cybercrime investigators or forensic analysts on current practices and trends.
- **Mock IR Drills:** Engage in simulated tabletop exercises to build decision-making and response coordination skills.

Textbooks:

1. Bill Nelson, Amelia Phillips, Christopher Steuart. Guide to Computer Forensics and Investigations (6th Edition). Cengage.
2. Jason Luttgens, Matthew Pepe, Kevin Mandia. Incident Response & Computer Forensics (3rd Edition). McGraw Hill.
3. Sherri Davidoff, Jonathan Ham. Network Forensics: Tracking Hackers Through Cyberspace. Pearson.

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Lab 1 (Unit 1)** Title: Imaging and Preservation of Digital Evidence **Sub-Objectives:** • Create forensic image of USB or VM disk using FTK Imager • Generate MD5/SHA256 hash of acquired image • Log all actions in a chain of custody worksheet	CO 1
2	**Lab 2 (Unit 2)** Title: Analyzing Artifacts from a Suspicious User Profile **Sub-Objectives:** • Extract registry hives, browser history, and system logs • Use Autopsy to create timeline of user activity • Detect deleted files, executed apps, USB insertions	CO 2

3	**Lab 3 (Unit 3)** Title: Detecting Malware in Memory & Network Logs **Sub-Objectives:** • Capture live memory from compromised VM • Use Volatility to analyze suspicious processes • Identify malware communication in PCAP logs using Wireshark	CO 3
4	**Lab 4 (Unit 4)** Title: Incident Response Drill – Web Server Breach **Sub-Objectives:** • Triage logs and system images from a breached web server • Detect backdoors and persistent malware • Prepare IR plan and suggest recovery strategy	CO 4
5	**Capstone Assignment** Title: End-to-End Forensic Investigation of a Corporate Data breach **Sub-Objectives:** • Investigate a simulated case involving insider data theft + malware • Perform disk, memory, and network forensics across 2 compromised hosts • Correlate logs, identify attacker tools, map MITRE TTPs • Deliver full forensic report with timeline, evidence screenshots, and mitigation plan	CO 4

Generative AI Tools & Techniques

Program Name	B. Tech CSE (Specialization in Cybersecurity)			
Course Name: Generative AI Tools & Techniques	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		3-0-2	4	45
Type of Course:	DSE			
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 deploy 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

- **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	No. of hours: 12
• **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 Applications**	**No. of hours: 13**
• **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 into Applications**	**No. of hours: 10**
• **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.		

Learning Experience:

Inside Classroom Learning

- **Prompt Engineering Labs**: Learn how to craft zero-shot, few-shot, and chain-of-thought prompts for LLMs like GPT-4, Gemini, and Claude. Use prompt tuning techniques for consistent, reliable outputs.
- **Model Playground Sessions**: Work hands-on with tools like ChatGPT, Stable Diffusion, Midjourney, and RunwayML to generate text, code, art, and video. Explore inputs/outputs, parameters, and limitations.

- **Custom Model Workflows**: Fine-tune open-source models (e.g., LLaMA, Mistral, SDXL) using LoRA/PEFT on platforms like Hugging Face + Google Colab/Kaggle notebooks.
- **Toolchain Deep Dives**: Learn to integrate GenAI in apps using LangChain, LlamaIndex, and Hugging Face Transformers. Build simple agents, chatbots, and content pipelines.
- **Ethics & Safety Debates**: Explore the dark side—bias, hallucinations, prompt injection, copyright abuse—and debate real-world cases with legal/ethical implications.
- **Real-Time Demos**: Build live apps like resume writers, meme generators, blog summarizers, and AI tutors using Next.js, Vercel, and OpenAI APIs.

Outside Classroom

- **Model Comparison Projects**: Evaluate outputs from GPT-4, Claude 3, Gemini, and open-source models on tasks like summarization, translation, or reasoning.
- **Sandbox Experiments**: Spin up your own environments to test GenAI tools like KoboldAI, Fooocus, and AudioCraft locally or on cloud GPUs.
- **Hackathons & Prompt Battles**: Compete in team-based rapid prototyping events—build a GenAI product in 48 hours or crack a creative challenge with best prompt wins.
- **Guest Sessions with Creators**: Join webinars and AMAs with GenAI researchers, startup founders, and tool builders for behind-the-scenes perspectives.
- **Portfolio Project Dev**: Build and publish at least one full-stack GenAI-powered product (e.g., voice-enabled chatbot, AI content assistant) on GitHub + deploy it.
- **Self-Led Critique Rounds**: Peer-review outputs based on originality, coherence, accuracy, and bias—develop an eye for GenAI quality and red flags.

Textbooks:

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

Reference Books:

- Mary L. Boas, Mathematical Methods in the Physical Sciences, 3rd Edition, Wiley, 2005.
- Gilbert Strang, Introduction to Linear Algebra, 5th Edition, Wellesley-Cambridge Press, 2016.
- Ray Wylie C and Louis C Barret, Advanced Engineering Mathematics, Tata Mc-Graw-Hill, Sixth Edition.

Lab Experiments

Lab Task 1: Exploring Pre-trained LLMs & Basic Text Generation

Real-World Scenario:

You've joined an AI innovation lab and are assigned to test various LLMs for customer query automation. Your task is to load open-source models and generate coherent responses based on simple inputs.

Objectives:

- Set up Hugging Face Transformers in Google Colab or VS Code
- Load a pre-trained model (e.g., Google Gemma, Falcon) for text generation
- Process and tokenize input/output sequences
- Compare the outputs from different models using the same prompt

Tools: Google Colab, Hugging Face Transformers, PyTorch or TensorFlow

Lab Task 2: Prompt Engineering for Multi-Modal Tasks

Real-World Scenario:

You work for a content automation startup. Your role is to experiment with prompting strategies to generate marketing copy, summarize product reviews, and offer coding help.

Objectives:

- Write and compare zero-shot, few-shot, and CoT prompts for text summarization
- Build prompts for Python and SQL code generation
- Design prompts for creative writing (e.g., product descriptions, stories)
- Compare prompt performance in OpenAI Playground and Hugging Face Inference API
- **Tools:** OpenAI Playground, Hugging Face, LangChain (prompt templates), Google Colab

Lab Task 3: Open-Source LLM Applications – Chatbot & Code Generator

Real-World Scenario:

You are developing a lightweight AI assistant for customer service using a local LLM and building a tool that auto-generates basic code snippets from user instructions.

Objectives:

- Run open LLMs like LLaMA 3, Mistral, or Falcon locally or on Hugging Face
- Build a simple Q&A chatbot using Hugging Face Pipeline or LangChain
- Use an LLM to generate SQL/Python/Java code from natural language
- Implement basic error handling for input-output quality

Tools: LangChain, Hugging Face, LlamaIndex, Google Colab

Lab Task 4: End-to-End LLM App Deployment

Real-World Scenario:

As part of a hackathon team, you're tasked with building and deploying an AI app (summarizer, chatbot, or writing assistant) accessible via web.

Objectives:

- Use LangChain or Gradio to chain model outputs and create a complete workflow
- Design and test front-end interaction with an LLM backend (Gradio/Streamlit)
- Deploy the app using Hugging Face Spaces or Streamlit Cloud
- Document model usage, prompt tuning, and API endpoints

Tools: Gradio, Streamlit, Hugging Face Spaces, LangChain, FastAPI (optional)

Capstone Project: Build & Deploy a Custom AI-Powered Tool with LLMs

Real-World Scenario:

You are building a real-world AI-powered assistant for a startup. This tool should accept natural language input, respond intelligently, and optionally integrate with external data (e.g., documents or APIs).

Objectives:

- Design an AI product concept (e.g., InterviewBot, Legal Summarizer, Startup Pitch Generator)
- Choose and fine-tune an open-source LLM or use prompt optimization
- Develop the frontend using Gradio or Streamlit with backend logic in LangChain

- Deploy the solution publicly and demonstrate its usage and limitations

Tools: Hugging Face, LangChain, Google Colab, Gradio, Streamlit, OpenAI API (optional)

Major Project- I

Program Name:	**B. Tech CSE (Specialization in Cybersecurity)**

Course Name:	Course Code	L-T-P	Credits
Summer Internship-Major Project -I	**ETCCPR 701**	0-0-8	4
Type of Course:	**Project**		

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 **2-4 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.

5.3 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 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**

5.4 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 assigned faculty mentor and/or project review committee.

5.5 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. Evaluation Metrices

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**	**20**

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**

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.

Summer Internship-III

Program Name:	**B. Tech CSE (Specialization in Cybersecurity)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Summer Internship-III	**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 2nd semester and before the commencement of the 3rd semester.

Internship Options
Students can choose from the following options:

1. **Industry Internship (Offline):**
    a. Students must produce a joining letter at the start and a relieving letter upon completion.
2. **Government/Research Institution Internship:**
    a. Students can engage in a research internship with premier government or research organizations such as IITs, IISc, ISRO, DRDO, CSIR, NPL, etc.
3. **On-Campus Bootcamp/ Industry Internship Programs:**
    a. The university will offer on-campus internships in collaboration with industry partners.

4. **Deliverables and Documentation:**

5. 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 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(Assessments)

Program Name:	B. Tech CSE (Specialization in Cybersecurity)

Course Name:	Course Code	L-T-P	Credits
Summer Internship- IV (Assesment)	**ETCCIN702**	0-0-16	8
Type of Course:	INT		

Preface:

The **B.Tech Final Semester Full-Time Project Work** is a culmination of the academic journey for engineering students at the School of Engineering & Technology, K.R. Mangalam University. This detailed Standard Operating Procedure (SOP) is designed to guide students through their project, ensuring a comprehensive, practical, and outcome-driven approach that aligns with the principles of the **National Education Policy (NEP) 2020**.

The SOP provides a framework for students to choose from three types of projects—**Industrial Projects**, **Research & Development (R&D) Projects**, and **Start-up Projects**. It emphasizes experiential learning, real-world problem-solving, and interdisciplinary collaboration, reflecting NEP 2020's focus on holistic development, innovation, and entrepreneurship. Students will work under the mentorship of both internal faculty and external experts, ensuring they are equipped with the skills and knowledge required to excel in industry, research, or entrepreneurship.

This document outlines each stage of the project work, from proposal submission to final evaluation, and offers clear guidelines for successful completion. By adhering to this SOP, students will not only demonstrate their technical proficiency but also contribute meaningfully to industry, academia, and society.

Standard Operating Procedure (SOP) for B.Tech Final Semester Full-Time Project Work

1. Introduction

The **B.Tech Final Semester Full-Time Project Work** is an essential academic requirement aimed at providing students with the opportunity to apply theoretical knowledge to practical challenges. The project is designed to foster critical thinking, problem-solving, innovation, and research-oriented learning, with a focus on real-world industrial, research, and entrepreneurial domains. Students may choose from:

- **Industrial Project**: Solving real industrial problems in collaboration with an industry partner.
- **Research & Development (R&D) Project**: Contributing to academic and applied research, with external guidance from academic/research institutions.
- **Start-up Project**: Developing and launching innovative start-up ideas with entrepreneurial mentors.

The SOP ensures that the project aligns with **NEP 2020 guidelines**, emphasizing interdisciplinary, practical, and outcome-based learning.

2. Objectives

The primary objectives of the full-time project are:

- **Application of Theoretical Knowledge**: Enabling students to apply their academic learning to practical problems.
- **Holistic Development**: Promoting interdisciplinary learning, critical thinking, creativity, and problem-solving.
- **Research and Innovation**: Encouraging innovative solutions, leading to publications, patents, or prototypes.
- **Industry Collaboration**: Fostering partnerships with industries for real-world problem-solving.
- **Entrepreneurship Development**: Developing entrepreneurial skills and creating viable start-ups.
- **Global Competency**: Ensuring students develop the skills required to excel in global environments through research, innovation, and collaboration.

3. Types of Projects

a) Industrial Project

Students working on **Industrial Projects** will:

- Collaborate with an industry partner.
- Identify specific, real-world challenges faced by the company.
- Propose and implement a solution that provides value to the industry.
- Develop a final product or prototype that can be implemented in the industrial setting.

Project Proposal:

- Problem Statement and Objectives: Identify the industrial problem and outline the objectives.
- Proposed Solution: Present a detailed methodology for solving the problem.
- Deliverables: Define tangible deliverables, including prototypes, software, or hardware.
- Expected Impact: Outline the expected impact on the industry.

Evaluation Criteria:

- Practical implementation and solution viability (40%)
- Project innovation (20%)
- Industrial applicability and impact (20%)
- Final presentation and report quality (20%)

b) Research & Development (R&D) Project

The **R&D Project** focuses on creating innovative research outcomes through collaborations with academic or research institutions. This can result in publications, research reports, or new discoveries.

Project Proposal:

- Literature Review: Detailed research on existing work related to the chosen topic.
- Hypothesis/Research Questions: Define the specific research problem or question.
- Methodology: Include data collection, experimental design, and analysis techniques.
- Research Timeline: Step-by-step phases of research with milestones.

External Mentor: Collaboration with an **external academic expert** is mandatory for research projects. The external mentor must be a research professional with expertise in the specific field of study.
Internal Mentor: Each student will also be assigned an **internal faculty member** who will supervise the project. The internal mentor will ensure that the research meets academic standards and deadlines.
Evaluation Criteria:

- Quality of Research and Novelty (30%)
- Research Methodology (25%)
- Contributions to the field (20%)
- Final Report, Presentation, and Publication (25%)

c) Start-up Project
The **Start-up Project** involves developing a business model or creating a start-up venture. Students work on a product/service idea that addresses a significant market need or societal problem.
Project Proposal:

- Start-up Idea: Explain the business or product idea.
- Market Research: Detailed research on the market, target customers, competitors, and potential revenue streams.
- Business Plan: Define the steps needed to take the idea to market, including funding, development phases, marketing, and operational plans.
- Product Prototype: If applicable, develop a working prototype.

Mentorship:

- **External Mentor**: An industry/start-up expert will guide the student in refining the idea, business model, and market strategy.
- **Internal Faculty Mentor**: An internal mentor will provide academic guidance and ensure the start-up idea is feasible and innovative.

Evaluation Criteria:

- Start-up viability and market potential (30%)
- Product or service innovation (30%)
- Prototype/Business Model Development (20%)
- Final Pitch/Presentation and Start-up Plan (20%)

4. Roles and Responsibilities
a) Student's Responsibilities:

- Select a suitable project topic based on interests (industrial, R&D, or start-up).
- Draft and submit a detailed proposal with objectives, methodology, timelines, and deliverables.
- Coordinate with both external and internal mentors regularly for feedback and guidance.
- Maintain a weekly progress report for both mentors.
- Submit a final comprehensive report and present the project.

b) Internal Supervisor:

- Guide the student throughout the project.
- Provide academic input and ensure that the project aligns with the program outcomes.
- Conduct progress reviews and ensure timelines are adhered to.

- Evaluate the project at the mid-term and final stages.

c) External Mentor:

- Offer specialized industrial, research, or entrepreneurial guidance.
- Provide real-world problem insights for industrial and start-up projects.
- Ensure the project is relevant to the chosen industry, research domain, or start-up ecosystem.
- Participate in the final evaluation of the project.

5. Project Phases

Phase 1: Proposal Submission and Approval

- Students will submit a project proposal during the first two weeks of the final semester.
- The proposal must include the problem statement, objectives, literature review (for R&D projects), methodology, and expected outcomes.
- The proposal is subject to review and approval by the internal supervisor and external mentor.

Phase 2: Planning and Resource Allocation

- Once approved, the student will develop a project plan that includes:
  - **Project Milestones**: Break down the project into smaller tasks with defined milestones.
  - **Resource Requirements**: Identify any software, hardware, lab resources, or tools required for the project.
  - **Team Roles**: For group projects, define the roles of each team member.
  - **Risk Assessment**: Highlight potential risks and the corresponding mitigation strategies.

Phase 3: Mid-term Review

- A mid-term review will be conducted halfway through the project to assess progress.
- Students will present their work to a committee consisting of the internal supervisor, external mentor, and department head.
- The review will assess the progress against the timeline and suggest course corrections if needed.

Phase 4: Final Execution and Evaluation

- **Industrial Projects**: Students must submit a prototype or industrial report, demonstrating the solution's applicability to the industry.
- **R&D Projects**: Students must submit a final research report or publish findings in academic journals.
- **Start-up Projects**: Students must present a business plan, along with a working prototype, market analysis, and revenue model.

Phase 5: Final Report Submission and Presentation

- **Final Report**: The project report should contain a title page, abstract, introduction, problem statement, objectives, methodology, results, discussion, conclusions, future scope, references, and appendices.
- **Presentation**: Students will deliver a final presentation to a panel of evaluators, showcasing their work, findings, or product.
- **Evaluation**: Based on the final report and presentation, students will be awarded marks in accordance with the evaluation rubrics.

6. Collaboration and Mentorship

For **Research Projects**, the mentorship will involve both:

- **External Mentor**: An academic expert outside the institution, preferably from a reputed university or research institute.
- **Internal Mentor**: A faculty member from the student's department to provide academic and administrative guidance.

For **Industrial Projects**:

- External mentorship will come from industry professionals, preferably from the partnering company.

For **Start-up Projects**:

- External mentorship will involve experienced entrepreneurs, start-up founders, or investors.

Mentors will:

- Provide critical inputs on the technical, business, or research aspects of the project.
- Offer feedback and advice during each phase of the project.

7. 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.

K.R. MANGALAM UNIVERSITY

Major Project- II

Program Name:	B. Tech CSE (Specialization in Cybersecurity)		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Summer Internship- Major Project -II	**ETCCPR802**	0-0-8	4
Type of Course:	**Project**		

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 **2-4 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.

5.3 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 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**

5.4 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:

3. **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.

4. **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 assigned faculty mentor and/or project review committee.

5.5 Faculty Allocation Process

3. **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.
4. **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

8. **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.
9. **Task Monitoring:**
   Assign tasks with clear deadlines and **track the progress and completion status** for each team through the project cycle.
10. **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.

11. **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**.
12. **Justification for Meeting Cancellation:**
    Any cancellation of scheduled team meetings by the mentor must be supported with a **valid and documented reason**.
13. **GitHub Activity Monitoring:**
    Regularly monitor each team's **GitHub repository** to verify individual contributions and maintain **performance records** for all team members.

14. Evaluation Metrices

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**	**20**

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**

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.