K.R. MANGALAM UNIVERSITY
THE COMPLETE WORLD OF EDUCATION

SCHOOL OF ENGINEERING AND TECHNOLOGY

Programme Handbook
(Programme Study and Evaluation Scheme)

Bachelor in Computer Applications (BCA)
With
Specialization in Cyber Security
[Honors/Honors with Research]
Programme Code: 227

UNDERGRADUATE PROGRAMME

As per National Education Policy 2020

(Multiple Entry and Exit in Academic Programmes)

(With effect from 2025-26 session)

Approved in the 38th Meeting of Academic Council

Held on 28 June 2025

Table of Contents

1. Preface

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

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

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

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

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

We are also dedicated to fostering a culture of innovation and entrepreneurship through our Entrepreneurship and Incubation Center and initiatives like 'MindBenders,' 'Hack-KRMU,' and participation in the 'Smart India Hackathon.' These programs are designed to inspire and prepare students to become forward-thinking leaders in the technology sector.

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

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

School of Engineering & Technology
K.R Mangalam University

2. NEP-2020: Important features integrated in the curriculum

K.R. Mangalam University has adopted the National Education Policy NEP-2020 to establish a holistic and multidisciplinary undergraduate education environment, aiming to equip our students for the demands of the 21st century. Following the guidelines of NEP-2020 regarding curriculum structure and duration of the undergraduate programme, we now offer a Four-Year Undergraduate Programme with multiple entry and exit points, along with re-entry options, and relevant certifications.

- **UG Certificate:** After completing 1 year (2 semesters with the required number of credits) of study, and an additional vocational course/internship of 4 credits during the summer vacation of the first year.
- **UG Diploma:** After completing 2 years (4 semesters with the required number of credits) of study, and an additional vocational course/internship of 4 credits during the summer vacation of the second year.
- **Bachelor's Degree:** After completing 3-year (6 semesters with the required number of credits) programme of study.
- **Bachelor's Degree (Honours):** 4-year Bachelor's Degree (Honours) with the required number of credits after eight semester's programme of study.
- Students who secure 75% marks and above in the first six semesters and wish to undertake research at the undergraduate level can choose a research stream in the fourth year. Upon completing a research project in their major area(s) of study in the 4th year, a student will be awarded **Bachelor's Degree (Honours with Research)**.

Advantage of pursuing 4-year Bachelor’s degree programme with Honours/Honours with Research is that the Master’s degree will be of one year duration. Also, a 4-year degree programme will facilitate admission to foreign universities.

S. No.	Broad Categories of Courses	Minimum Credit Requirement for Four Year UG Programme
1	Major (Core)	67
2	DSE	28
3	Open Elective	09
4	Ability Enhancement Course (AEC)	08
5	Skill Enhancement Course (SEC)	14
6	Value-Added Course (VAC)	06
7	Summer Internship	14
8	Project	14
9	Community Service	2
10	MOOC	4
9	Total	166

3. Categories of Courses

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

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

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

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

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

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

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

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

Categories:

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

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

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

4. 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

5. About The School

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

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

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

Industry integration is a key component of our educational model. We collaborate with renowned organizations such as IBM, Samatrix, Xebia, EC Council, and ImaginXP to provide students with practical, real-world experience through internships, projects, and workshops. These partnerships ensure that our students are well-prepared to meet industry demands. Additionally, we offer elective courses in areas such as AI, Cloud Computing, Cyber Security, and Full Stack Development, allowing students to tailor their learning experience to align with their career goals. We are also committed to fostering innovation and entrepreneurship. Our **Entrepreneurship and Incubation Center** and initiatives like 'MindBenders,' 'Hack-KRMU,' and participation in the **Smart India Hackathon** inspire students to develop forward-thinking solutions and entrepreneurial ventures.

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

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

8. 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.

9. Programme Outcomes (PO)

Engineering Graduates will be able to:

PO1. Core Competencies: Graduates will possess a strong foundation in computer science principles, 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 IT tools including prediction and modeling to complex computer science activities with an understanding of the limitations.

PO3. Societal and Environmental Responsibility

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

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

PO5. **Effective Communication and Team Collaboration**

Excel in both individual and team roles within diverse and multidisciplinary settings, while communicating complex computer science 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 contexts.

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

10. Programme Specific Outcomes (PSO)

PSO1: Understanding the core concepts, theories, tools, techniques, and methodologies of Artificial Intelligence and Data Science.

PSO2: Applying AI and Data Science principles to solve real-world problems and make data-driven decisions.

PSO3: Analysing data and AI methodologies to uncover insights and address challenges.

PSO4: Evaluating and optimizing AI models and data solutions for enhanced performance.

PSO5: Designing and developing innovative AI and Data Science solutions to tackle complex problems.

11. Career Avenues

Graduates of the BCA program have a wide array of career opportunities, including:

1. **Software Developer**: Design, code, and maintain software across various domains such as web, mobile, game, and enterprise applications.
2. **Systems Analyst**: Evaluate and enhance organizational computer systems, design new solutions, and optimize existing processes for improved efficiency.
3. **Data Scientist**: Analyze large datasets, apply statistical methods and machine learning, and develop predictive models to drive data-informed decisions.

4. **AI Engineer**: Develop and implement AI algorithms, including natural language processing, computer vision, and other intelligent systems.
5. **Cybersecurity Analyst**: Protect systems and data by identifying vulnerabilities, implementing security measures, and responding to security incidents.
6. **Network Engineer**: Design, deploy, and maintain reliable and secure network infrastructures, and troubleshoot network issues to ensure optimal performance.
7. **IT Project Manager**: Oversee technology projects from planning to execution, manage teams, coordinate resources, and ensure timely and budget-compliant delivery.
8. **Database Administrator**: Maintain database systems, ensure data integrity and security, and optimize performance for efficient data management.
9. **Quality Assurance Engineer**: Test and ensure the reliability and functionality of software applications by developing test plans, identifying issues, and collaborating with development teams.
10. **Research and Development**: Engage in cutting-edge research, explore emerging technologies, and contribute to innovation in academia, industry labs, or R&D departments.

12. Duration

BCA: 3 years/BCA Honors with Research: 4 Years (Full-Time)

13. Eligibility Criteria

Course Level/Duration/System:

Undergraduate / Three or Four years/6 or 8 Semesters with multiple entry and exit. The following option will be made available to the students joining BCA Research Program:

Multiple Exit Options to Students

Exit Option	Naming convention for the award Certification/Diploma/Degree
Exit after 1st year	▪ Undergraduate Certificate in Computer Application.
Exit after 2nd year	▪ Undergraduate Diploma in Computer Application
Exit after 3rd year	▪ Bachelor's in computer application (BCA) with specialization in AI & Data Science
Exit after 4th year	▪ Bachelor's in computer application with Honours: BCA(Honours) with specialization in AI & Data Science OR ▪ Bachelor in Computer Application Honours with Research: BCA (Honours with Research) with specialization in AI & Data Science

Minimum Eligibility Criteria:

Minimum eligibility criteria for opting the course in the fourth year will be as follows:

1. BCA (Honours with Research): BCA Degree

2. For BCA (Honours): BCA Degree

Note : The students who are eligible for BCA (Honours with Research) shall have choice to pursue istribution either BCA (Honours) or BCA (Honours with Research).

Semester wise credit Distribution

Program Name	I	II	III	IV	V	VI	VII	VIII	Total Credits
BCA (AI & DS)	22	22	24	24	22	22	16	14	166

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

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

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

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

Core Learning

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

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

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

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

Skill Development

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

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

Capstone and Exit Phase

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

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

Co-Curricular and Extra-Curricular Activities

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

Community Connect

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

Ethics and Professional Values

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

Career Counselling and Entrepreneurship

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

Course Registration

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

e. Student Support Services

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

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

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.

I. Evaluation Scheme:

Evaluation Components	Weightage
Internal Marks (Theory) **Continuous Assessment (40 Marks)** **Lab/Project/ Quizzes/ Assignments and Essays/ Presentations/ Participation/ Case Studies/ Reflective Journals**	40 Marks
Internal Marks (Theory) – Mid Term Exam	20 Marks
External Marks (Theory): - **End term Examination**	40 Marks
Total	100 Marks

g. Feedback and Continuous Improvement Mechanisms:

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

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

h. Academic Integrity and Ethics:

K.R. Mangalam University upholds the highest standards of academic integrity and ethics as a core value of its educational philosophy. The university implements a zero-tolerance policy towards academic misconduct, including plagiarism and other unethical practices. To ensure transparency and honesty in academic work, plagiarism detection software like Drillbit 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

Semester I

S.N	**Course_Code**	Category	Course Title	L	T	P	Credits
1	ETCCCA101	Major	Mathematics for Modern Computing Applications	4	0	0	4
2	ETCCWD102	Major	Foundations of Web Development	3	0	2	4
3	ETCCCPP103	Major	Problem Solving with Python	3	0	2	4
4	ETCCCS104	Major	Essentials of Computer Science and Career Skills	4	0	0	4
5	SEC	SEC	Data Visualization with Power BI	2	0	0	2
6	VAC	VAC	VAC-I (Environmental Studies)	2	0	0	2
7	SEC	SEC	Foundations of Data-Driven Decision Making	2	0	0	2
		TOTAL		**20**	**0**	**4**	**22**

Semester II

S.N		Category	Course Title	L	T	P	Credits
1	ETCCDS201	Major	Essentials of Data Structures	3	0	2	4
2	ETCCDM202	Major	Introduction to Discrete Mathematics	3	0	2	4
3	ETCCED203	Major	Interactive Front-End Development	3	0	2	4
4	ETCCCC204	Major	Cloud Computing	3	0	0	3
5	SEC	SEC	Introduction to Design Thinking and Prototyping	1	0	2	2
6	OEC	Open Elective	Open Elective-I	3	0	0	3
7	ETCCPR205	Proj	Minor Project-I	2	0	0	2
		TOTAL		**18**	**0**	**8**	**22**

Semester III

S. N		Category	Course Title	L	T	P	Cre dits
1	ETCCAD 301	Major	Analysis & Design of Algorithms	3	0	2	4
2	ETCCWD 302	Major	Back-End Web Development	3	0	2	4
3		DSE	Specializaton Course-I (Cybersecurity Essentials and Practices)	3	0	2	4
4	OEC	Open Elective	Open Elective-II	3	0	0	3
5	AEC	AEC	Verbal Ability	2	0	0	2
6	**ETCCIN 305**	INT	Summer Internship-I	0	0	4	2
7	SEC	SEC	Competitive Coding - I	2	0	0	2
8	CS	CS	Community Service	1	0	0	1
9	VAC	VAC	VAC-II	2	0	0	2
TOTAL				19	0	10	24

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

Semester IV

		Category	Course Title	L	T	P	Credits
1	ETCCPJ401	Major	Essentials of Object Oriented programming with Java	3	0	2	4
2	ETCCMS 402	Major	Fundamentals of Database Management Systems	3	0	2	4
3		DSE	Specializaton Course-II (Network Defense and Security Protocols)	3	0	2	4
4	OEC	Open Elective	Open Elective-III	3	0	0	3
5	AEC	AEC	Communication & Personality Development	2	0	0	2
6	**ETCCPR 405**	Proj	Minor Project-II	0	0	4	2
7	SEC	SEC	Competitive Coding - II	2	0	0	2
8	CS	CS	Club/Society	1	0	0	1
9	VAC	VAC	VAC-III	2	0	0	2
		TOTAL		19	0	10	24

Semester V

S.N		Category	Course Title	L	T	P	Credits
1	ETCCOS501	Major	Foundations of Operating Systems	3	0	2	4
2	ETCCDE502	Major	System Design Essentials	4	0	0	4
3		DSE	Specilaization Course-III (Applied Cryptography for Cybersecurity)	3	0	2	4
4		DSE	Specilaization Course-IV (Building Secure Applications)	3	0	2	4
5	AEC	AEC	Arithmetic and Reasoning Skills	2	0	0	2
6	**ETCCIN504**	INT	Summer Internship-II	0	0	4	2
7		SEC	Competitive Coding - III	2	0	0	2
TOTAL				**17**	**0**	**10**	**22**

Semester VI

		Category	Course Title	L	T	P	Credits
1	ETCCCN601	Major	Introduction to Computer Networks	3	0	2	4
2	ETCCEE602	Major	Agile Software Engineering Essentials	3	0	2	4
3		DSE	Specilaization Course- V (Ethical Hacking)	3	0	2	4
4		DSE	Specilaization Course- VI (Securing Cloud Infrastructures and Services)	3	0	2	4
5	AEC	AEC	Comprehensive Placement Preparation	2	0	0	**2**
6	**ETCCPR604**	Proj	Minor Project-III	2	0	0	2
7		SEC	Competitive Coding - IV	2	0	0	2
		TOTAL		**18**	**0**	**8**	**22**

Semester VII

S.N		Category	Course Title -**BCA (Honours)**	L	T	P	Credits
1	ETCCGA703	Major	Applied Generative AI	3	0	2	4
2		DSE	Specilaization Course- VII (Incident Response and Cyber Investigation)	3	0	2	4
3	MOOC	MOOC	MOOC in the relevant domain of Specilaization (Swayam/ NPTEL/AICTE's ELIS)	2	0	0	2
4	ETCCPR701	PROJ	Major Project -I	0	0	8	4
5	**ETCCIN702**	INT	Summer Internship-III	0	0	4	2
			TOTAL	**8**	**0**	**16**	**16**

OR

Sr. No.	**Category**	**Course Title- BCA (Honours with Research)**	**L**	**T**	**P**	**Credits**
1	Major	Applied Research Methodologies in Computer Science	3	0	2	4
2	DSE	Specilaization Course- VII	3	0	2	4
3	MOOC	MOOC in the relevant domain of Specilaization (Swayam/ NPTEL/AICTE's ELIS)	2	0	0	2
4	PROJ	Major Project -I	0	0	8	4

5	INT	Summer Internship-III	0	0	4	2
Total			**8**	**0**	**16**	**16**

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

SN	Category	Course Code	Course Title	L	T	P	C
1	DSE-3		Discipline Specific Elective -II	4	-	-	4
2	DSE-4		Discipline Specific Elective -III	4	-	-	4
2	DSE-5		Discipline Specific Elective -IV	4	-	-	4
3	DSE-6		Discipline Specific Elective -II Lab	-	-	2	1
4	DSE-7		Discipline Specific Elective III Lab	-	-	2	1

5	DSE-8		Discipline Specific Elective IV Lab	-	-	2	1
6	INT-3	ENSI451	Summer Internship-III	2	-	-	2
7	MOOC-2		MOOC in the domain of AI & Data Science (Swayam/ NPTEL/AICTE’s ELIS)	-	-	-	2
	TOTAL			14	0	6	19

Discipline Specific Elective - II (Cloud Computing)

(i)	DSE	ENSP401	Computational Services in The Cloud	4	-	1	4
	DSE	ENSP451	Computational Services in The Cloud Lab	-	-	2	1
(ii)	DSE	ENSP403	Microsoft Azure Cloud Fundamentals	4	-	1	4
	DSE	ENSP453	Microsoft Azure Cloud Fundamentals Lab	-	-	2	1
(iii)	DSE	ENSP405	Storage and Databases on Cloud	4	-	1	4
	DSE	ENSP455	Storage and Databases on Cloud Lab	-	-	2	1
(iv)	DSE	ENSP407	Application Development and DevOps on Cloud	4	-	1	4
	DSE	ENSP457	Application Development and DevOps on Cloud Lab	-	-	2	1

Discipline Specific Elective - III (Full Stack Development)

(i)	DSE	ENSP409	Mobile Application Development using iOS	4	-	-	4
	DSE	ENSP459	Mobile Application Development using iOS Lab	-	-	2	1
	DSE	ENSP411	DevOps & Automation	4	-	-	4

(ii)	DSE	ENSP461	DevOps & Automation Lab	-	-	2	1
(iii)	DSE	ENSP413	.Net FRAMEWORK	4	-	-	4
	DSE	ENSP463	.Net FRAMEWORK Lab	-	-	2	1
(iv)	DSE	ENSP415	New Age Programming languages	4	0	0	4
	DSE	ENSP465	New Age Programming languages Lab	0	0	2	1

Discipline Specific Elective IV(Cyber Security)

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

Subjects through Industry Support

Sem	Type	Code	Subject	L	T	P	C	

I	DSE	ENSP101	Clean Coding with Python	4	0	0	4	IBM
I	DSE	ENSP151	Clean Coding with Python Lab	0	0	2	1	IBM
II	DSE	ENSP102	Overview of AI, Data Science, Ethics and Foundation of Data Analysis	4	0	0	4	Samatrix
II	DSE	ENSP152	Overview of AI, Data Science, Ethics and Foundation of Data Analysis Lab	0	0	2	1	Samatrix
III	DSE	ENSP205	Probabilistic Modelling and Reasoning	-	-	4	2	Samatrix
IV	DSE	ENSP212	Foundation of Machine Learning	4	-	-	4	Samatrix
IV	DSE	ENSP262	Foundation of Machine Learning lab	-	-	2	1	Samatrix
IV	DSE	ENSP359	Big Data Analysis with Scala and Spark Lab	-	-	4	2	IBM
V	DSE	ENSP302	Natural Language Processing	4	-	-	4	Samatrix
V	DSE	ENSP352	Natural Language Processing Lab	-	-	2	1	Samatrix
V	SEC	SEC040	Data Science - Tools and Techniques Lab	0	0	4	2	Samatrix
VI	DSE	ENCA306	Basics of Neural Networks and Deep Learning	4	-	-	4	Samatrix
VI	DSE	ENCA354	Neural Networks and Deep Learning Lab	-	-	2	1	Samatrix
VI	PROJ	ENSI451	Project & Case Studies by Samatrix	-	-	-	2	Samatrix
				20	0	22	33	

Semester VIII

SN	Code	Category	Course Title **BCA (Honours)**	L	T	P	Credits
1	**ETCCIN801**	INT	Industry Internship	0	0	0	8
2	**ETCCPR802**	PROJ	Major Project-II (**Industry Track)**	0	0	0	4
3	**MOOC**	MOOC	MOOC in the relevant domain of Specilaization (Swayam/ NPTEL/AICTE's ELIS)	0	0	0	2
		TOTAL		**0**	**0**	**0**	**14**
		OR					
		Category	Course Title - **BCA (Honours with Research)**	L	T	P	Credits
1	**ETCCIN803**	INT	Research Internship	0	0	0	8
2	**ETCCPR804**	PROJ	Major Project-II (**Research Track)**	0	0	0	4
3	**MOOC**	MOOC	MOOC in the relevant domain of Specilaization (Swayam/ NPTEL/AICTE's ELIS)	0	0	0	2
		TOTAL		**0**	**0**	**0**	**14**

Semester	**I**	**II**	**III**	**IV**	**V**	**VI**	**VII**	**VIII**	**Total**
BCA (AI & DS) Honors with Research	22	22	24	24	22	22	16	14	166

TOTAL CREDITS=166 for Honors/Honors with Research

Syllabus

Semester: 1

Mathematics for Modern Computing Applications

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: **Mathematics for Modern Computing Applications**	**Course Code**	**L-T-P**	**Credits**
	ETCCCA101	4-0-0	4
Type of Course:	Major		
Pre-requisite(s): NA			

Course Perspective:

This course introduces essential mathematics used in modern computing, designed especially for BCA students. It focuses on applying basic math concepts in areas like programming, data analysis, web development, and artificial intelligence. Through simple Python-based exercises and real-life examples, students will learn how math powers today’s digital world.

Course Outcomes (COs):
After completing this course, students will be able to:

COs	**Statements**
CO 1	**Understand** core mathematical topics like matrices, statistics, calculus, and numerical methods relevant to computing.
CO 2	**Use** Python programming to implement and solve basic mathematical problems from real-world IT applications.
CO 3	**Analyze** and apply mathematical concepts to real-life scenarios in AI, web development, and data processing.
CO 4	**Develop** computational thinking, logical reasoning, and problem-solving ability through code and simulations.

Course Outline:

<table>
<tr><td><b>Unit Number: 1</b></td><td><b>Title: Linear Algebra Basics in Computing</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
Introduction to vectors and matrices
• Solving simple systems of equations
• Real-life applications of matrices in computing
• Matrix operations using Python (NumPy)
Real-World Scenario:
Use matrices to manage a basic inventory system or to represent digital images (image as matrix pixels).</td></tr>
<tr><td><b>Unit Number: 2</b></td><td><b>Title: Probability & Statistics for Data</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
• Introduction to probability
• Common distributions (e.g., normal, binomial)
• Mean, median, mode, and standard deviation
• Basics of hypothesis testing
Real-World Scenario:
Use probability to analyze customer buying behavior and statistics to summarize survey data in an app.</td></tr>
<tr><td><b>Unit Number: 3</b></td><td><b>Title: Introduction to Calculus & Optimization</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
Basic idea of derivatives and integrals
• Simple optimization using graphs
• Role of calculus in system performance
• Gradient descent overview
Real-World Scenario:
Use calculus to improve the loading speed of a webpage or optimize product pricing in an app.</td></tr>
<tr><td><b>Unit Number: 4</b></td><td><b>Title: Basics of Numerical Methods</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
Finding roots of equations (Bisection, Newton-Raphson)
• Introduction to numerical integration
• Simple error checking
• Problem-solving with Python
Real-World Scenario:
Solve a temperature prediction problem using interpolation or simulate real-time values in a game.</td></tr>
</table>

Classroom Learning Experience

Inside Classroom Learning:

1. **Concept-driven Lectures:** Concepts such as matrix operations, statistics, and calculus will be introduced using examples from digital technologies.
2. **Coding Walkthroughs:** Real-time demonstrations of Python programs solving matrix or statistical problems.
3. **Interactive Problem Solving:** Students will solve statistical and optimization problems collaboratively on whiteboards.
4. **Visualization Sessions:** Graphical interpretation of optimization and calculus topics using Python libraries (e.g., Matplotlib).
5. **Mini Case Studies:** Short scenario-based discussions on how math is used in AI, gaming, or e-commerce.

Outside Classroom Learning:

1. **Take-Home Programming Exercises:** Python-based problems on matrix algebra, hypothesis testing, or numerical methods.
2. **Project-Based Assignments:** Apply linear algebra or probability to real datasets (e.g., build a GPA calculator or simulate user activity).
3. **Online Simulations and Tutorials:** Use platforms like Desmos, GeoGebra, or Khan Academy for visual learning.
4. **Peer Discussions & Code Reviews:** Forums and GitHub-based code-sharing and collaborative debugging.
5. **Reflection Worksheets:** Weekly worksheets to connect mathematical concepts to IT tools or applications.

Textbooks:

1. Deisenroth, M. P., Faisal, A. A., & Ong, C. S. (2020). Mathematics for Machine Learning. Cambridge University Press.
2. Bruce, P., & Bruce, A. (2017). Practical Statistics for Data Scientists: 50 Essential Concepts. O'Reilly Media.

Foundation of Web Development

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: **Foundation of Web Development**	**Course Code**	**L-T-P**	**Credits**
	ETCCWD102	3-0-2	4
Type of Course:	Major Course		
Pre-requisite(s), if any: Basic computer skills and familiarity with using a browser and editor			

Course Description:

This foundation course introduces BCA students to the core building blocks of front-end web development. It focuses on HTML for structure, CSS for styling, and basic JavaScript for interactivity. By working on small, guided projects, students will gain practical experience in designing clean, responsive web pages and applications. The course emphasizes writing structured code, developing problem-solving skills, and understanding the role of each technology in the frontend stack.

Course Outcomes (COs)

Upon completion of this course, students will be able to:

COs	Statements
CO 1	**Design** and structure accessible web pages using semantic HTML5.
CO 2	**Apply** styling, layout techniques, and responsive design using CSS.
CO 3	**Use** JavaScript for adding basic functionality to web pages.
CO 4	**Build** a complete front-end website using HTML, CSS, and JavaScript.
CO 5	**Apply** learned concepts to develop a capstone mini-website project.

Detailed Syllabus

Unit Number: 1	**Title HTML Basics**	**No. of hours: 06**
Content: • Introduction to HTML, structure of a webpage • Basic tags: headings, paragraphs, images, links, lists • Tables and Forms: elements and attributes • Semantic HTML: header, footer, section, article, nav • Using Visual Studio Code and Live Server • *Mini Project:* Personal profile webpage with sections and form		
Unit Number: 2	**Title CSS Fundamentals**	**No. of hours: 10**
Content: • CSS syntax: selectors, properties, values • Styling elements: colors, fonts, background, border • Box Model: padding, margin, border • Flexbox for layout and alignment • Responsive design with media queries • Simple transitions and hover effects • *Mini Project:* Portfolio layout with responsive header and sections		
Unit Number: 3	**Title: JavaScript Essentials**	**No. of hours: 10**
Content: • Introduction to JavaScript and how to embed it • Variables, data types, operators • Control flow: if, else, loops • Functions, scope, and simple events • Arrays and basic manipulations • *Mini Project:* Interactive quiz using prompts and alerts		
Unit Number: 4	**Title: Final Project Development**	**No. of hours: 04**
Content: • Build a complete multi-section responsive web page • Integrate structure (HTML), styling (CSS), and simple logic (JavaScript) • Include contact forms, images, navigation, and minor interactivity • *Project Themes:* Portfolio / Event Page / Product Showcase		

Lab Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Simple Web Page with HTML (Unit 1)** **Objective:** Learn HTML structure and essential tags **Sub-objectives:** - Create a personal info page - Add About, Hobbies, and Contact sections - Use headings, lists, images, and links - Create a basic contact form	CO 1
2	**Styled Portfolio Page (Unit 2)** **Objective:** Style and layout using CSS **Sub-objectives:** - Apply custom colors, fonts, and spacing - Use Flexbox to organize header, main, and footer - Add hover effects on navigation or buttons - Make layout responsive with media queries	CO 2
3	**JavaScript Quiz App (Unit 3)** **Objective:** Write logic with JavaScript **Sub-objectives:** - Use prompt() to ask questions - Store questions in an array - Use loops to cycle through and score answers - Display total score at the end	CO 3
4	**Final Responsive Website (Unit 4)** **Objective:** Combine all technologies in one site **Sub-objectives:** - Build a one-page website with semantic HTML - Layout using Flexbox and make it responsive with media queries - Add basic JavaScript interactivity (form validation, light/dark mode)	CO 4
5	**Capstone: Mini Project Website** **Objective:** Final hands-on project demonstrating all skills **Choose One Theme:** - Personal Portfolio- Online Bookstore - College Fest/Event Page - Food Menu Display - Local Service Listing	CO 5

Requirements:

- At least 3 sections: Home, Services/Projects, Contact
- Semantic HTML5 and clean CSS3 layout
- Mobile responsiveness via Flexbox and media queries
- JavaScript-based interactivity (form validation, quiz, toggle mode, etc.)

- Proper folder structure and comments

Deliverable:

- Fully working site
- Submit all files (HTML, CSS, JS) in a zip folder
- Must be mobile-responsive and tested on multiple screen sizes

Classroom Learning Experience

Inside Classroom Learning:

- **Interactive Demonstrations:** Step-by-step walkthroughs of HTML, CSS, and JavaScript concepts using VS Code and browser.
- **Live Coding Sessions:** Real-time creation of web pages and debugging of code in front of students.
- **Design Thinking Exercises:** Students sketch layouts on paper before implementing them.
- **Mini-Reviews:** Code walkthroughs to explain semantic structure and styling choices.
- **Collaborative Peer Review:** Students exchange and review code to learn from each other's practices.
- **Short Quizzes & Code Fixes:** In-class problem-solving focused on layout issues or JS errors.

Outside Classroom Learning:

- **Hands-on Assignments:** Build and submit HTML/CSS/JS projects.
- **Self-Guided Practice:** Use platforms like CodePen, JSFiddle, or GitHub Pages for experimentation.
- **Discussion Forums:** Use LMS discussion boards to ask queries, share code, or debug collaboratively.
- **Online Resources:** Access free courses from Mozilla Developer Network (MDN), W3Schools, and YouTube tutorials.
- **Capstone Preparation:** Peer brainstorming and testing before final submission for feedback.

Textbooks:

- 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.
- HTML & CSS: Visual QuickStart Guide – 9th Ed., Elizabeth Castro, Bruce Hyslop, Peachpit Press (Pearson), 2021, 978-0136581444.

Problem Solving with Python

Program Name	**B. Tech (Computer Science and Engineering)**		
Course Name: Problem Solving with Python	**Course Code**	**L-T-P**	**Credits**
	ETCCCPP103	3-0-2	4
Type of Course:	Major Course		
Contact Hours	45 hrs		
Version			
Pre-requisite(s), if any: None			

Course Perspective: This course introduces students to fundamental programming concepts using Python. It aims to build basic skills in computer programming, data handling, and problem solving through real-life examples.

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

COs	Statements
CO 1	**Understand** the fundamentals of Python, including syntax, variables, data types, and operators, and apply them in basic programming tasks
CO 2	**Apply** control flow structures and functions to solve problems using Python's built-in data structures like lists, dictionaries, sets, and tuples.
CO 3	**Implement** object-oriented programming concepts and manage file handling and exception handling to develop robust Python applications.
CO 4	**Analyze** and **visualize** data using Python libraries, and explore the basics of web development and client-server architecture.

Course Outline:

Unit Number: 1	**Title: Getting started with Python**	**No. of hours: 10**
Content:		

<table>
<tr><td colspan="3"><b>Introduction to Python Programming:</b> History of Python, Python Features, Local Environment Setup, setting up Python environment: Installing Python, IDEs (e.g., VSCode, Anaconda, PyCharm);
<b>Python Programming Basics:</b> Python Syntax, Keywords, Understanding Variables, numbers, and data types.
<b>Operators:</b> Arithmetic, Assignment, Comparison, Logical, Identity, Membership, Bitwise.
<b>Python String:</b> Manipulating strings, Modify Strings, String Concatenation, Format - Strings, Escape Characters, Inbuilt method of Strings.</td></tr>
<tr><td><b>Unit Number: 2</b></td><td><b>Title: Control Flow and Data Structures in Python</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
<b>Conditional statements:</b> if, elif, else; Loops: for loop, while loop, nested loops; Control flow statements: break, continue.
<b>Functions:</b> Defining functions, parameters, function calls, return statement; Scope and lifetime of variables. Recursive and Lambda Functions.
<b>Basics of Python Data Structure:</b> Mutable: List, Dictionary, Set, Immutable Types: Numbers, String, tuple.
<b>Lists:</b> Operations, methods, slicing; Tuples and sets: Properties, operations; Dictionaries: Creating, accessing, modifying.</td></tr>
<tr><td><b>Unit Number: 3</b></td><td><b>Title: Object-Oriented Programming and File Handling in Python</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
<b>OOPs Concept:</b> Introduction to object-oriented programming (OOP), Abstraction, encapsulation, Polymorphism;
<b>Classes and objects:</b> Defining classes, creating objects; Constructors in Python, Parameterized and Non-parameterized.
<b>Inheritance, and polymorphism:</b> Types, Method overriding and overloading; Special methods (dunder methods): __init__, ____str__, __repr_____.
<b>File handling:</b> Opening, reading, writing, and closing files;
<b>Exception handling</b>: try, except, finally blocks.</td></tr>
<tr><td><b>Unit Number: 4</b></td><td><b>Title: Data handling and web development in python</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
<b>Data visualization with matplotlib:</b> line plot, multiple subplots in one figure, histograms, bar charts, pie charts, scatter plots
<b>Handling data with pandas:</b> series, dataframes, read and write csv file, operations using dataframe
<b>Numpy arrays:</b> numpy - datatype, array operations, statistical functions.</td></tr>
</table>

LAB Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Basics of Python Programming** **Real-World Scenario:** A student assistant is asked to build small tools like a simple calculator, name formatter, or message generator to help in college events. **Sub-objectives:** - Set up Python environment using IDLE or Thonny - Write basic programs using variables, input, and output - Perform string operations like name formatting and character counts - Use arithmetic and logical operators to create a mark -sheet calculator or greeting card generator **Tools:** Python (IDLE or Thonny), Online Python Editors	CO 1
2	**Conditional Statements and Collections** **Real-World Scenario:** A small retail shop needs to maintain a list of products and generate bills with discount logic using Python. **Sub-objectives:** - Write Python programs using if-else and loops for decision making - Use lists, dictionaries, and sets to manage product data and apply offers - Create user-defined functions for order input and billing - Demonstrate tuple usage for category labeling (e.g., stationery, electronics) **Tools:** Python (IDLE), Visual Studio Code	CO 2
3	**Object-Oriented Programming & File Management** **Real-World Scenario:** Design a "College Result System" that stores student marks and displays summary reports using OOP and files. **Sub-objectives:**- Create classes like Student, Subject, and MarksSheet - Use constructors, methods, and basic inheritance to reuse code - Store student records in text or CSV files and retrieve them - Apply exception handling for invalid inputs (e.g., non-numeric marks) **Tools:** Python (OOP), .txt/.csv files	CO 3
4	**Data Handling and Visualization** **Real-World Scenario:** An intern at a local NGO wants to analyze volunteer attendance and event participation data visually. **Sub-objectives:** - Use pandas to read and analyze CSV files (e.g., attendance logs) - Create basic charts like bar and line plots using matplotlib - Use NumPy to perform simple calculations like average attendance - Save graphs and cleaned data back to CSV **Tools:** pandas, NumPy, matplotlib, Jupyter Notebook or Google Colab	CO 4

5	**Capstone Project: Student Performance Dashboard** **Real-World Scenario:** Develop a small system to import student exam results, calculate performance, and present charts for report generation. **Sub-objectives:** - Read and clean marks data using pandas - Use OOP to modularize student data handling - Generate bar charts and pie charts to represent subject -wise performance - Export summary reports with charts and formatted data to disk **Tools:** pandas, NumPy, matplotlib, Python (OOP)	CO 5

Classroom Learning Experience

Inside Classroom Learning:

- **Live Coding Demonstrations:** Instructors write and explain Python programs live (e.g., calculator, result system) to build familiarity with syntax and logic flow.
- **Concept Reinforcement through Practice:** Each topic (conditions, loops, collections, classes) is followed by 15–20 minute coding tasks to immediately apply what's learned.
- **Error Debugging Walkthroughs:** Teachers guide students through common coding mistakes, encouraging peer-assisted debugging.
- **Interactive Whiteboard Sessions:** Key concepts like flowcharts, class diagrams, and logic structures are illustrated on the board for visual learners.
- **Real-World Scenario Mapping:** Instructors relate topics (like file handling or data visualization) to relatable use cases like student grading systems or attendance charts.
- **Tool Familiarization Labs:** Regular hands-on sessions with IDLE, VS Code, Jupyter Notebook, and Google Colab to build comfort with real-world tools.

Outside Classroom Learning:

- **Practice Worksheets & Mini-Assignments:** Students practice concepts independently through structured assignments like building a mark-sheet calculator or data summarizer.
- **Peer Code Review Sessions:** Weekly peer discussions or Git-based reviews to enhance code readability, reusability, and adherence to best practices.
- **Video Tutorials & Online Quizzes:** Supplemental self-paced content via platforms like W3Schools, Python Tutor, and HackerRank to reinforce learning.
- **Project Planning Workshops:** Students brainstorm and plan their capstone project themes (e.g., dashboards or billing systems) under faculty mentorship.
- **Portfolio Development:** Each student documents lab work and mini-projects in a GitHub or offline portfolio to showcase progress and build confidence.

ESSENTIALS OF COMPUTER SCIENCE AND CAREER SKILLS

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: **Essentials of Computer Science and Career Skills**	**Course Code**	**L-T-P**	**Credits**
	ETCCCS104	4-0-0	4
Type of Course:	Major		
Pre-requisite(s), if any: NA			

Course Description

This course lays the groundwork for BCA students in computing fundamentals and practical digital skills. Students will explore how computers work, how to use essential digital tools safely, and how to approach problems logically using computational thinking. The course also focuses on professional skills development like using collaboration tools, improving tech literacy, and preparing for a career in IT.

Course Outcomes (COs)

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

COs	Statements
CO 1	Explain the core concepts of computers, software, and operating systems.
CO 2	Use digital tools and apply basic cybersecurity practices.
CO 3	Apply logical thinking and flowcharts to solve simple problems.
CO 4	Build soft skills like digital resume creation and career goal

Course Outline

Unit Number: 1	**Title:** ***Computer Fundamentals & Problem Solving***	**No. of hours: 10**
Content: • History and evolution of computing and hardware generations • Components: input, output, storage, CPU, memory • Number systems: binary, decimal, hexadecimal • Encoding schemes: ASCII, Unicode		

- Software types: OS, applications, utilities
- Algorithm basics and IPO model
- Computational thinking: abstraction, decomposition, patterns, algorithm design
- Flowcharts and pseudocode basics
- Responsible tech usage & ethics

Unit Number: 2	**Title:** ***Linux Essentials & Open Source***	**No. of hours: 20**

Content:

- Why Linux? Real-world applications
- Kernel, shell, filesystem overview
- WSL or VirtualBox setup
- Basic terminal commands: `ls`, `cd`, `cp`, `mv`, `rm`, `mkdir`
- File permissions: `chmod`, `chown`
- Monitoring: `top`, `ps`, `kill`
- Networking: `ping`, `wget`
- Write and run shell scripts
- Open-source values & licenses (MIT, GPL)

Unit Number: 3	**Title:** ***Tech Domains & Career Awareness***	**No. of hours: 10**

Content:

- Domains: Web, App Dev, AI, Cybersecurity
- Indian & global tech case studies
- Student project: explore a domain + job roles
- Present using blog, infographic, or video
- Map subjects to careers (e.g., Linux → DevOps)
- Interdisciplinary tech in health/agriculture
- Indian startup & student innovation showcase

Unit Number: 4	**Title: Tools for Coding & Collaboration**	**No. of hours: 10**

Content:

- Using VS Code & Jupyter
- Git basics: add, commit, push, clone
- GitHub repositories + README, .gitignore
- Markdown documentation
- Learning tools: W3Schools, HackerRank, Kaggle
- Digital notes: Notion, Obsidian
- Team tools: Slack, Discord, Trello, MS Teams

Unit Number: 5	**Title:** ***Career Skills & Future Readiness***	**No. of hours: 10**

Content:

- SMART goal setting
- Certifications: Google, Cisco, AWS

- Build a digital portfolio: LinkedIn, GitHub
- Join coding events, internships, OSS programs
- Hackathons: Smart India Hackathon, GSoC
- Boost soft skills: communication, time mgmt., mindset

Classroom Learning Experience

Inside Classroom Learning:

- **Concept-Driven Lectures:** Each unit begins with an interactive session using whiteboard, slides, and real-life analogies (e.g., CPU = brain, storage = memory).
- **Lab-Integrated Sessions:** Hands-on lab time is integrated with theory – e.g., students write shell scripts during Linux lectures.
- **Peer Explainers:** Students present tech domains or Git workflows to the class in brief peer-led segments.
- **Live Demonstrations:** Faculty use terminal, GitHub, or Jupyter live to show tasks like script writing, version control, and markdown.
- **Digital Productivity Challenges:** Mini contests using Notion, Trello, or Slack to reinforce collaboration skills.

Outside Classroom Learning:

- **Mini Assignments:** Build a personal landing page, publish a README on GitHub, or create a project infographic.
- **Online Practice Platforms:** Encourage self-learning on platforms like HackerRank (coding), GitHub (versioning), and Kaggle (datasets).
- **Peer Reviews:** Share shell scripts, markdowns, or GitHub repos with peers for review and improvement.
- **Career Planning Reflections:** Use worksheets to map tech domains to certifications, internships, and portfolios.
- **Community Engagement:** Join Discord communities, GitHub Issues, or open-source Telegram groups to explore collaborative problem-solving.

Textbooks:

1. Brookshear, J. Glenn, Brylow, Dennis. *Computer Science: An Overview* (13th Edition). Pearson Education, 2022. ISBN: 9780137505954.

Data Visualization using Power BI

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: **Data Visualization with Power BI**	**Course Code**	**L-T-P**	**Credits**
	SEC037	2-0-0	2
Type of Course:	SEC-1		
Pre-requisite(s), if any: Basic knowledge of Excel & data numbers			

Course Perspective:
This course introduces students to the principles of data visualization using Microsoft Power BI. It equips learners with the ability to transform raw data into meaningful insights through interactive dashboards and reports. By the end of the course, students will be able to apply visualization techniques, use real-time data sources, and create compelling business intelligence solutions suitable for various industries.

Course Outcomes (COs)

COs	Statements
CO 1	**Understand** the fundamentals of data visualization and business intelligence.
CO 2	**Develop** interactive dashboards and reports using Power BI.
CO 3	**Connect** and transform various data sources for analytical use.
CO 4	**Apply** Power BI to real-world scenarios for decision-making and storytelling.

Course Outline

Unit Number: 1	**Title:** ***Introduction to Data Visualization and Power BI***	**No. of hours: 10**
Content: • Importance and principles of data visualization • Overview of business intelligence tools • Introduction to Power BI interface and components • Types of data: structured vs unstructured • Importing data into Power BI from Excel, CSV ***Hands-on Case:*** *Create your first report in Power BI using sales data from Excel.*		
Unit Number: 2	**Title: Data Preparation and**	**No. of hours: 10**

	Transformation in Power BI	
Content: • Understanding Power Query Editor • Data cleaning, filtering, and shaping • Merging and appending queries • Working with date/time and text columns • Creating calculated columns and measures using DAX ***Hands-on Case:*** *Clean and transform HR employee records to prepare for a performance analysis report.*		
Unit Number: 3	**Title:** ***Visualizations and Interactive Dashboards***	**No. of hours: 10**
Content: • Creating different types of charts: bar, pie, line, map • Custom visuals and conditional formatting • Using slicers, filters, and drill-throughs • Design principles for dashboards • Themes, layouts, and branding in reports ***Hands-on Case:*** *Design a financial dashboard to track monthly revenue and expenses for a startup.*		
Unit Number: 4	**Title: Data Publishing and Real-World Applications**	**No. of hours: 10**
Content: • Publishing reports to Power BI Service • Setting up data refresh schedules • Collaborating using Power BI workspace • Row-Level Security and data governance • Use cases: marketing analytics, supply chain, healthcare insights ***Hands-on Case:*** *Publish a marketing campaign analysis report and share with stakeholders via Power BI Service.*		

Classroom Learning Experience

Inside Classroom Learning:

- **Interactive Tool Walkthroughs:** Each session begins with a live demonstration of Power BI features, such as data import, DAX formulas, or dashboard layout using real-world datasets (sales, HR, finance).
- **Concept-to-Click Transitions:** Theoretical principles (e.g., types of visualizations, structured vs. unstructured data) are followed by immediate implementation using Power BI Desktop.
- **Hands-on Mini Projects:** During each class, students work on small exercises aligned with business scenarios like building a pie chart for HR data or setting up slicers on a revenue dashboard.

- **Collaborative Problem Solving:** Students are grouped into teams to clean messy datasets or optimize report layouts, encouraging peer learning and design thinking.
- **Case-Based Dashboard Challenges:** Weekly tasks require students to apply concepts like drill-throughs or calculated columns to industry-specific cases (e.g., performance review or campaign reach).
- **Immediate Feedback & Iteration:** Faculty review reports in real-time, offering suggestions on visualization clarity, layout, and interaction.

Outside Classroom Learning:

- **Self-Paced Video Assignments:** Students are assigned Power BI tutorials from Microsoft Learn or YouTube to review and replicate independently.
- **Data Storytelling Reflections:** Students write short explanations of how their dashboard communicates insights to a non-technical audience.
- **Portfolio Development:** Each student maintains a GitHub/OneDrive folder or Google Site showcasing their Power BI reports for HR, finance, marketing, etc.
- **Peer Review Loop:** Students exchange dashboards and give feedback on usability, accuracy, and insight quality, simulating client review scenarios.
- **Real Dataset Exploration:** Students are encouraged to find datasets (e.g., Kaggle, data.gov.in) to practice importing, cleaning, and reporting beyond class-provided material.
- **Capstone Report Submission:** Final reports are submitted as published links via Power BI Service, including refresh schedules and access rights configured for review.

Text Books:

1. **Kolokolov, A., & Zelensky, M. (2024).** *Data visualization with Microsoft Power BI: How to design savvy dashboards*. O'Reilly Media. ISBN: 9781098152789
2. **Blissy, T. (2024).** *Power BI 2024 user guide: An all-encompassing book with new updates to become a self-sufficient and competent Power BI user*. Independently Published. ISBN: 9798328999298

FOUNDATIONS OF DATA-DRIVEN

DECISION MAKING

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: **Foundations of Data-Driven Decision Making**	**Course Code**	**L-T-P**	**Credits**
	SEC	2-0-0	2
Type of Course:	SEC-1		
Pre-requisite(s), if any: None			

Course Perspective:

This course introduces students to the principles of using data as a core component in decision-making processes across industries. Students will explore how to collect, interpret, and apply data insights to drive strategic decisions. By combining foundational concepts in data analysis with real-world examples, learners will develop data literacy and critical thinking skills needed for modern roles.

Course Outcomes (COs):

- **CO1:** Understand the importance and workflow of data-driven decision making.
- **CO2:** Analyze datasets to identify trends and support informed decisions.
- **CO3:** Apply basic data analysis tools and visualization to real-world problems.
- **CO4:** Evaluate outcomes of decisions using measurable indicators and improve strategies.

Unit Number: 1	Title: **Introduction to Data-Driven Decision Making**	No. of hours: 10
Content: • Definition and importance of data-driven decision making • Decision making models: descriptive, predictive, prescriptive • Role of data in business and everyday decisions • Sources of data: internal vs external • The data-informed mindset ***Case Study:*** *Analyze a university's enrollment data to suggest improvements in outreach.*		

<table>
<tr><td><b>Unit Number: 2</b></td><td><b>Title: Fundamentals of Data Collection and Cleaning</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
• Types of data: qualitative vs quantitative
• Primary vs secondary data
• Methods of data collection: surveys, observations, sensors
• Data cleaning basics: missing values, duplicates, formatting
• Importance of data integrity and ethics
<b><i>Hands-on:</i></b> <i>Clean a messy sales dataset and prepare it for analysis.</i></td></tr>
<tr><td><b>Unit Number: 3</b></td><td><b>Title: Basic Data Analysis and Visualization</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
• Exploratory data analysis (EDA): mean, median, mode, standard deviation
• Understanding distributions and trends
• Visual tools: bar chart, histogram, scatter plot, line chart
• Tools used: Excel, Google Sheets, or Power BI (introductory use)
• Telling a story with data
<b><i>Hands-on:</i></b> <i>Analyze and visualize website traffic data to recommend content strategy.</i></td></tr>
<tr><td><b>Unit Number: 4</b></td><td><b>Title: Decision Strategies and Outcome Evaluation</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
• Setting measurable goals and KPIs
• Making decisions based on patterns and evidence
• Feedback loops and continuous improvement
• Communicating findings to stakeholders
• Pitfalls in data-driven strategies
<b><i>Case Scenario:</i></b> <i>Evaluate the impact of a marketing campaign using predefined KPIs.</i></td></tr>
</table>

Classroom Learning Experience

Inside Classroom Learning:

- **Real-Life Contextual Lectures:** Key concepts like decision-making models (descriptive, predictive, prescriptive) are introduced through relatable real-world and business examples (e.g., admissions, marketing, HR).
- **Case-Based Discussions:** Each unit includes a case study discussion (e.g., university enrollment, marketing KPIs) where students brainstorm data-backed improvement strategies using structured frameworks.

- **Hands-on Exercises:** In-class tasks include cleaning raw datasets, calculating statistical summaries, and designing charts using Excel or Google Sheets.
- **Visualization Walkthroughs:** Faculty demonstrate how to create bar charts, histograms, and scatter plots live using Power BI or Sheets, connecting visuals to insights.
- **Interactive Quizzes and Polls:** Real-time polls and quizzes help assess understanding of concepts like KPIs, feedback loops, and pitfalls in decision-making.
- **Collaborative Learning:** Students work in small teams to explore data scenarios, define metrics, and justify decisions to peers, simulating real-world stakeholder presentations.

Outside Classroom Learning:

- **Mini Data Projects:** Students collect or are assigned datasets (e.g., school cafeteria sales or campus event attendance) to clean, analyze, and interpret independently.
- **Reflection Exercises:** After each unit, students submit a short write-up on how the topic (e.g., setting KPIs or using EDA) applies to a personal or campus-based decision.
- **Practice with Open Datasets:** Students are encouraged to explore open data portals like Kaggle, data.gov.in, or UNData to apply collection, cleaning, and visualization skills.
- **Storytelling with Data Tasks:** Learners build small narrative reports summarizing trends and recommendations from their analysis, reinforcing the communication of insights.
- **Peer Review & Feedback:** Students exchange draft analyses and dashboards, offering feedback on logic, visual clarity, and alignment with goals.
- **Capstone Scenario:** Final assignment requires students to choose a campus problem (e.g., declining event turnout) and propose data-driven solutions using methods learned across all units.

Text Books:

1. **Grand, C., & Bang, H. (2024).** *Data-driven decision-making for business*. Routledge.
2. **Steinhardt, G. (2024).** *Data-driven decision-making for product managers: A primer to harness data for impactful product decisions*. Springer.

Semester 2

Essentials of Data Structures

Course Name: Essentials of Data Structures (with Python)	**Course Code:** ETCCDS201	**L-T-P**	**Credits**
		3-0-2	4
Type of Course:	Major		
Pre-requisite(s), if any: Basics of Computer Programming			

Course Perspective:

This course provides a comprehensive introduction to data structures and algorithms usin Python, equipping students with the skills to develop efficient software solutions. It emphasize practical implementation, problem-solving, and real-world applications using Python’s built-i features and libraries like NumPy, collections, heap, and networks. The course fosters algorithmi thinking, optimization, and hands-on coding for technical interviews and industry applications.

The Course Outcomes (COs):

On completion of the course the participants will be:

COs	Statements
CO 1	Implement and analyze linear and non-linear data structures using Python.
CO 2	Apply sorting, searching, and hashing techniques for efficient data processing.
CO 3	Design and implement algorithms using stacks, queues, linked lists, and trees.
CO 4	Utilize graphs, heaps, and advanced data structures for real-world applications.
CO 5	Develop and optimize solutions for practical problems in AI, databases, and networking using Python.

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

Course Outline:

<table>
<tr><td><b>Unit Number: 1</b></td><td><b>Foundations of Data Structures</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
• Introduction to Data Structures in Python: <b>Lists, Tuples, Sets, Dictionaries</b>
• <b>Abstract Data Types (ADT):</b> Definition, Implementation, and Use Cases
• <b>Algorithm Complexity Analysis:</b> Big-O, Ω, Θ Notations
• <b>Arrays:</b> One-dimensional & Multi-dimensional, <b>NumPy Arrays</b>
• <b>Sparse Matrices:</b> Storage Optimization, Addition & Transposition
• <b>File Handling:</b> Reading/Writing Files, <b>Pickling & Unpickling Data</b>
• <b>Python Libraries for Data Structures:</b> NumPy, pandas, collections
• 🌍 <b>Real-World Use Cases:</b>
✓ <b>Database Indexing</b> – Using dictionaries for fast lookups
✓ <b>Image Processing</b> – Storing image pixels as NumPy arrays</td></tr>
</table>

✓ **Scientific Computing** – Using pandas for data analysis
✓ **Game Development** – Managing game entities using Python objects

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

Content:

- **Linked Lists:** Singly, Doubly, and Circular Linked Lists (Manual & collections.deque)
- **Stacks:** Implementation using Lists & collections.deque
- **Queues:** Standard Queues, Circular Queues, Priority Queues (heapq), Deques
- **Applications of Stacks & Queues:** Expression Evaluation, Job Scheduling
- **Choosing Between Lists, Arrays, and Deques:** Performance Considerations
- **Implementing Stacks, Queues, and Linked Lists** using Python Classes
- 🌍 **Real-World Use Cases:**

✓ **Text Editor Undo-Redo Mechanism** – Using stacks
✓ **CPU Scheduling** – Implementing priority queues using heapq
✓ **Handling Browser History & Backtracking** – Stack-based navigation
✓ **Messaging Systems** – Real-time queues for chat applications

Unit Number: 3	**Searching, Sorting, and Hashing**	**No. of hours: 10**

Content:

- **Searching:** Linear Search, Binary Search (Recursive & Iterative)
- **Sorting Algorithms:** Bubble, Selection, Insertion, Quick, Merge, Heap, Radix
- **Python Built-in Sorting:** sorted(), list.sort(), functools.cmp_to_key
- **Hashing:** Hash Functions, Hash Tables, Collision Resolution (Chaining, Open Addressing)
- **Implementing Custom Hash Maps** using Python Dictionaries
- **Competitive Coding Challenges:** Finding duplicates, Kth smallest element

Real-World Use Cases:

✓ **Spam Detection in Emails** – Using hash sets for quick lookups
✓ **E-Commerce Recommendations** – Sorting-based customer insights
✓ **Real-time Stock Market Analytics** – Sorting and searching in financial data

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

Content:

- **Trees:** Terminology, Binary Trees, Binary Search Trees (BST)
- **Self-Balancing Trees:** AVL Trees, Rotations (LL, RR, LR, RL)
- **B-Trees & B+ Trees:** Multi-way Search Trees, Disk-Based Storage

Optimization

- **Heaps:** Min Heap, Max Heap (heapq in Python)
- **Graph Representations:** Adjacency Matrix, Adjacency List (using networkx)
- **Graph Algorithms:** BFS, DFS, Dijkstra, Floyd-Warshall, Topological Sort
- **Minimum Spanning Tree (MST):** Kruskal’s & Prim’s Algorithm
- **Advanced Graph Algorithms:** Bellman-Ford, Tarjan’s Algorithm

Real-World Use Cases:

✓ **File System Organization** – Using B-Trees in databases

✓ **Google Maps & GPS Navigation** – Implementing Dijkstra’s

Text Books:

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). *Introduction to Algorithms* (4th ed.). MIT Press.
2. Goodrich, M. T., Tamassia, R., & Goldwasser, M. H. (2013). *Data Structures and Algorithms in Python.* Wiley.

Lab Experiments – Data Structures with Python

Lab Task 1: Python Basics and Core Data Structures

- **Real-World Scenario:**

Develop a student record system that uses various data structures (lists, tuples, sets, dictionaries) to store and retrieve student details.

- **Sub-Objectives:**
  - Write Python scripts using lists, tuples, sets, and dictionaries.
  - Perform operations like add, delete, search, and update using these structures.
  - Demonstrate file reading/writing using pickling and unpickling.
  - Use NumPy arrays to represent and perform operations on student marks.

Tools: Python, NumPy, pandas

Lab Task 2: Implementing and Comparing Linear Data Structures

- Real-World Scenario:

Design a task scheduler using Stacks and Queues to prioritize tasks based on type (real-time, batch, delayed).

- **Sub-Objectives:**
  - Manually implement Singly and Doubly Linked Lists.
  - Use collections. deque to simulate Stack and Queue operations.
  - Create a circular queue for real-time message buffering.
  - Simulate undo-redo operations using stack classes in Python.

Tools: Python, collections module

Lab Task 3: Sorting, Searching and Hashing in Action

- **Real-World Scenario:**

Build a library management system that organizes and searches books using various sorting and searching algorithms.

- **Sub-Objectives:**

- Implement Linear and Binary Search (both iterative and recursive).
- Implement Bubble, Insertion, Merge, and Quick Sort.
- Create a hash map to index books by author and title.
- Handle collisions in hash table using chaining and open addressing.

Tools: Python, functools, custom classes

Lab Task 4: Trees and Graph-Based Algorithms

- Real-World Scenario:

Design a college routing map to navigate departments using graph traversal and shortest path algorithms.

- **Sub-Objectives:**
  - Manually implement a Binary Search Tree and AVL Tree.
  - Use heapq for priority queue-based traversal.
  - Represent graphs using adjacency matrix and list (via networkx).
  - Apply BFS, DFS, Dijkstra, and Kruskal's algorithms on a graph.

Tools: Python, networkx, heapq

Capstone Project: End-to-End Student Academic Analytics System

- Real-World Scenario:

Build a full application that accepts student data, organizes it using trees, indexes it with hashing, and analyzes it with searching and sorting algorithms.

- **Sub-Objectives:**
  - Integrate all learned data structures into a working application.
  - Accept, store, and update records with different data structures.
  - Visualize student performance with sorted and filtered data.
  - Route data between modules using file handling and graphs.

Tools: Python, NumPy, pandas, networkx

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

Introduction to Discrete Mathematics

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Introduction to Discrete Mathematics	**Course Code**	**L-T-P**	**Credits**
		3-0-2	4
Type of Course:	Major Course		
Contact Hours	40 hrs		
Version			
Pre-requisite(s), if any: None			

Course Perspective: This course provides a comprehensive introduction to discrete mathematics and its applications in computer science and engineering. It covers logic, set theory, combinatorics, graph theory, and algebraic structures, which are fundamental in areas such as algorithm design, cryptography, machine learning, artificial intelligence, and network security. The course emphasizes real-world problem-solving, ensuring students understand where and how discrete structures are used in computing, automation, and decision-making.

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

COs	Statements
CO 1	Apply **propositional and predicate logic** to design and verify computer-based decision-making systems.
CO 2	Use **set theory, functions, and relations** in designing databases and computational models..
CO 3	Utilize **combinatorics and recurrence relations** to analyze algorithms and solve counting problems.
CO 4	Implement **graph theory concepts** in **computer networks, AI, and data structures**.
CO 5	Apply **algebraic structures, lattices, and Boolean algebra** in **cryptography, coding theory, and AI-based reasoning**.

Course Outline:

Unit Number: 1	**Title: Mathematical Logic & Proof Techniques**	**No. of hours: 10**

Content:

- **Propositional Logic** – Logical connectives, Truth tables, Tautologies, Contradictions.
- **Predicate Logic** – Quantifiers, Logical equivalence, Rules of inference.
- **Mathematical Proofs** – Direct proof, Indirect proof, Proof by contradiction, Mathematical induction.

Real-World Use Cases:

✓ **Error Detection in Software Development** – Formal verification of software using logic proofs.
✓ **Artificial Intelligence Chatbots** – Logical inference in rule-based AI chatbots & recommendation engines.

Unit Number: 2	**Title: Set Theory, Relations & Functions**	**No. of hours: 10**

Content:

- **Set Theory** – Operations on sets, Cartesian products, Power sets.
- **Relations & Their Properties** – Reflexive, Symmetric, Transitive, Equivalence, and Partial Ordering Relations.
- **Functions & Their Types** – Injective, Surjective, Bijective functions, Pigeonhole Principle.

Real-World Use Cases:

✓ **Computer Vision & Object Recognition** – Set operations for feature extraction in images.
✓ **Network Theory** – Graph-based relation models in social networks (e.g., Facebook's friend recommendations).

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

Content:

- **Basic Counting Principles** – Permutations, Combinations, Inclusion-Exclusion Principle.
- **Pigeonhole Principle & Applications**.
- **Recurrence Relations & Generating Functions** – Solving recurrence relations, Fibonacci numbers, Master Theorem.

Real-World Use Cases:

✓ **Data Compression Algorithms** – Used in Huffman Encoding for optimizing storage.
✓ **AI-Based Game Theory** – Predicting possible moves in chess & decision-making systems.

Unit Number: 4	**Title Graph Theory, Trees & Algebraic Structures**	**No. of hours: 10**

Content:

- **Graphs** – Representation (Adjacency Matrix, Adjacency List), Graph Traversal (BFS, DFS).
- **Special Graphs & Applications** – Trees, Planar Graphs, Euler & Hamiltonian Graphs.
- **Algebraic Structures** – Groups, Rings, Fields, Monoids, Lattices.
- **Boolean Algebra & Logic Circuits** – Boolean functions, Minimization, Karnaugh Maps (K-Maps).

Real-World Use Cases:

✓ **Shortest Path Algorithms (Dijkstra's, Floyd-Warshall)** – Used in Google Maps & GPS Navigation.

Text Books:

1. **Rosen, K. H. (2019).** *Discrete Mathematics and Its Applications* (8th ed.). McGraw-Hill.
2. **Liu, C. L. & Mohapatra, D. P. (2017).** *Elements of Discrete Mathematics* (4th ed.). McGraw-Hill.

Lab Experiments

Lab Task 1: Logic and Proof Techniques

Real-World Scenario:

Verify logical statements and develop small modules for rule-based inference in software testing.

Sub-Objectives:

- Create truth tables for compound propositions using Python or logic tools.
- Identify tautologies and contradictions with propositional logic.
- Apply rules of inference and quantifiers to validate logical arguments.
- Use Python to simulate direct, indirect, and contradiction-based proofs.
- Implement basic mathematical induction for formula validation (e.g., sum of series).

Tools: Python, Logic Calculator Tools (e.g., logic.ly, Visual Logic)

Lab Task 2: Set Theory, Relations & Functions

Real-World Scenario:

Design a system to model relationships among entities in a social network platform.

Sub-Objectives:

- Perform basic set operations and generate power sets using Python sets.
- Represent and analyze relations: reflexive, symmetric, transitive, and equivalence.
- Visualize Cartesian products and relation graphs using matplotlib or networkx.
- Implement functions and test for injectivity, surjectivity, and bijectivity.
- Apply Pigeonhole Principle in simple distribution problems using simulations.

Tools: Python, matplotlib, networkx

Lab Task 3: Combinatorics & Recurrence Relations

Real-World Scenario:

Simulate data compression and chess move prediction through combinatorial and recurrence models.

Sub-Objectives:

- Solve permutation and combination problems using Python math module.
- Apply inclusion-exclusion principle to overlapping event problems.
- Simulate pigeonhole principle with real-world data distribution examples.
- Formulate and solve recurrence relations using Python functions.
- Visualize Fibonacci and other recurrence-based sequences.

Tools: Python, SymPy

Lab Task 4: Graph Theory, Trees & Algebraic Structures

Real-World Scenario:

Create a route planning model using graph traversal and apply Boolean logic to simplify logic circuits.

Sub-Objectives:

- Implement and visualize graphs using adjacency matrix/list with networkx.
- Perform BFS and DFS traversals for maze or map exploration.
- Simulate trees, spanning trees, and check for Euler/Hamiltonian paths.

- Design and simplify Boolean expressions using Karnaugh maps or truth tables.
- Model basic algebraic structures like monoids and groups in Python.

Tools: Python, networkx, logic simulator, truth table calculators

Capstone Project: Discrete Mathematics in AI-Driven Decision Systems

Real-World Scenario:

Develop a basic AI decision-making system using propositional logic, set-based inference, graph search, and recurrence predictions.

Sub-Objectives:

- Integrate logical rule processing with propositional and predicate logic.
- Design relation models for AI chatbot or recommender system logic.
- Apply graph traversal to simulate recommendation paths or decision trees.
- Predict next-state behavior using recurrence formulas in sequential decision systems.

Tools: Python, networkx, SymPy, custom logic engine or rule-based system

Classroom Learning Experience

Inside Classroom Learning:

- **Real-Life Contextual Lectures:**
  Key concepts like logic, sets, and graphs are taught using examples from AI chatbots, GPS systems, and coding theory.
- **Case-Based Discussions:**
  Units include cases like social networks, recommendation engines, or data compression, connecting theory to real applications.
- **Hands-on Exercises:**
  In-class coding tasks using Python or tools like logic.ly for truth tables, set operations, graph traversal, and recurrence.
- **Visualization Walkthroughs:**
  Faculty demonstrate graphs, Karnaugh maps, and logic circuits using tools like networkx, SymPy, and Python plots.
- **Interactive Quizzes and Polls:**
  Quick quizzes and polls reinforce understanding of logic rules, function types, and graph properties.
- **Collaborative Learning:**
  Group tasks on modeling chatbot logic, social relations, or route-finding using discrete structures.

Outside Classroom Learning:

- **Mini Logic Projects:**
  Build small decision systems using propositional logic, graph search, and recurrence-based predictions.
- **Reflection Exercises:**
  Short write-ups on how discrete math applies to tech tools used daily (e.g., login logic, game AI).

- **Tool-Based Practice:**
  Use Python, logic simulators, and visual tools to practice outside lab hours.
- **Storytelling with Concepts:**
  Explain a concept like induction or graph theory through a real-world tech example.
- **Peer Review & Feedback:**
  Students exchange and critique logic models, recurrence solutions, or graph-based designs.
- **Capstone Scenario:**
  Final project involves designing a rule-based AI system integrating logic, sets, graphs, and recurrence.

Interactive Front- End Development

Course Code:
Program: BCA
Credits: 4 (L–T–P: 3–0–2)
Semester: III
Prerequisites: Basic knowledge of HTML, CSS, and JavaScript

Course Description

This course takes learners beyond the basics of web development into full-stack application building with modern JavaScript and React. Students will explore ES6+ features, asynchronous code patterns, and frontend tooling. With a focus on React, the course covers hooks, routing, state management, and deployment strategies. Through mini-projects and a final capstone, students will build and optimize professional-level SPAs (Single Page Applications) ready for real-world use.

Course Outcomes

After completing this course, students will be able to:

COs	Statements
CO 1	Use advanced JavaScript features (ES6+), async code, and modules in practical scenarios.
CO 2	Design and build React applications using functional components and hooks.
CO 3	Handle SPA-level state, routing, and data management efficiently.
CO 4	Optimize, test, and deploy applications using modern tools and best practices.
CO5	Deliver a full-fledged React project aligned with accessibility and performance benchmarks.

Course Outline:

Unit Number: 1	Title: Modern JavaScript Essentials	No. of hours: 12

Content Summary:

- ES6+ in practice: `let/const`, template strings, destructuring, spread/rest.
- Advanced functions: closures, currying, arrow functions, HOFs.
- Async code: Promises, async/await, fetch, AbortController.
- Tooling: modules, bundlers (Vite basics), linting with ESLint/Prettier.
- JS testing with Jest and mocking.

Lab Focus: Build a Node-based CLI tool that reads JSON and generates an HTML portfolio. Add command-line themes and unit test

Unit Number: 2	Title: React Core Concepts	No. of hours: 10

Content Summary:

- Project bootstrapping with Vite/CRA, folder structures.
- JSX and component rendering flow.
- State, props, events, and React’s core hooks (`useState`, `useEffect`, etc.).
- Styling with CSS Modules, Tailwind, and styled-components.
- Component interactions & prop drilling vs composition.

Lab Focus: Create a finance tracking UI with a dashboard, modals, and localStorage. Style using Tailwind or modules.

Unit Number: 3	**Title: Routing, State, and API Integration**	**No. of hours: 12**

Content Summary:

- Routing with React Router v6 (nested routes, protected routes).
- Context API & `useReducer` for shared state.
- Custom hooks (e.g., for fetch or forms).
- Working with APIs using React Query or SWR.
- Auth basics (JWT, login forms with `react-hook-form`).

Lab Focus: Real-time chat app simulation using context, routing, login/auth logic, and custom WebSocket-like hook.

Unit Number: 4	**Title: Performance, Testing & Deployment**	**No. of hours: 11**
Unit Number: 4	**Title: Optimization, Testing & Deployment**	**No. of hours: 12**

Content Summary:

- Profiling and debugging: React DevTools, performance tuning.
- Memoization: `React.memo`, `useMemo`, `useCallback`.
- Code splitting (`React.lazy`), lazy image loading.
- Accessibility practices and testing using Lighthouse.
- React Testing Library for UI testing; CI/CD using GitHub Actions.
- Deploying to Netlify or Vercel; adding PWA support.

Lab Focus: Optimize and audit a React e-commerce SPA. Add accessibility checks, test suites, and CI-based deployment.

Capstone Project: Smart-City Events Dashboard

- Feature planning with GitHub Projects.
- Integration with public event APIs, map views, filters, and favorites.
- State via Context, fetch logic with React Query.
- PWA optimization and offline support.
- Final deployment and performance report.

Lab Experiments

Lab 1 – ES6+ Portfolio Generator CLI

- Build a Node.js CLI tool that reads JSON and outputs static HTML.
- Use async/await, modules, and CLI arguments for theme selection.
- Add unit tests using Jest and publish to GitHub.

Lab 2 – Personal Finance Tracker (React Core UI)

- Setup a React project and build a dashboard with functional components.

- Manage UI state with useState and persist data in localStorage.
- Apply Tailwind or CSS Modules for styling.

Lab 3 – Real-Time Chat App with Routing & Context

- Implement routing for chat rooms and user login.
- Use Context and useReducer to manage global state.
- Simulate real-time chat using a custom hook and fake polling API.
- Add protected routes and login form with React-Hook-Form.

Lab 4 – E-Commerce SPA Performance Audit

- Optimize an existing React store with lazy loading and memoization.
- Write accessibility tests and unit tests for key components.
- Set up GitHub Actions to run tests and auto-deploy to Netlify.

Lab 5 – Capstone Project: Smart-City Events Dashboard

- Plan features using GitHub Projects Kanban board.
- Fetch public event data using API and cache with React Query.
- Implement filters, map view, and Context-based favourites.
- Optimize with PWA setup, deploy on Vercel, and meet Lighthouse performance scores ≥ 90.

Practical Textbook

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

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.

Cloud Computing

Program Name	Bachelor in Computer Applications (BCA)		
Course Name: **Cloud Computing**	**Course Code**	**L-T-P**	**Credits**
		3-0-0	3
Type of Course:	Major		
Pre-requisite(s): NA			

Course Perspective: This course introduces the fundamentals of cloud computing, including its architecture, service models (IaaS, PaaS, SaaS), deployment models, and practical usage of cloud platforms like AWS, Azure, and Google Cloud. Students will gain hands-on experience with cloud services, virtualization, storage, networking, security, and cloud-native application deployment.

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

CO 1	**Understand** core concepts of cloud computing and its evolution.
CO 2	**Identify** and compare cloud service models and deployment types.
CO 3	**Configure** and use basic services on public cloud platforms.
CO 4	**Design,** deploy, and manage cloud-native applications.
CO 5	**Implement** basic security and cost-optimization strategies on the cloud.

Course Outline:

Unit Number: 1	**Title: Cloud Computing Fundamentals**	**No. of hours: 10**
Content: • Introduction to Cloud Computing • Evolution of cloud & enabling technologies • **Cloud characteristics:** On-demand self-service, scalability, elasticity • **Cloud service models:** IaaS, PaaS, SaaS • **Deployment models:** Public, Private, Hybrid, and Community Cloud **Practical Applications:** • Case studies on cloud adoption • Identify services from AWS, Azure, and GCP corresponding to IaaS, PaaS,		

<table>
<tr><td colspan="3">SaaS</td></tr>
<tr><td><b>Unit Number: 2</b></td><td><b>Title: Virtualization and Cloud Infrastructure</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
• Introduction to virtualization: Hypervisors, VMs, Containers
• Compute, Storage, and Networking in the cloud
• Elasticity and load balancing
• Autoscaling and high availability concepts
<b>Practical Applications:</b>
• Setting up and launching VMs on AWS EC2 or Azure VMs
• Using S3 (or equivalent) for cloud storage
• Configuring virtual networks and load balancers</td></tr>
<tr><td><b>Unit Number: 3</b></td><td><b>Title: Cloud Services and Management</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
• Identity and Access Management (IAM)
• Monitoring, logging, and resource tagging
• Cloud pricing models and billing
• Cloud migration strategies and tools
<b>Practical Applications:</b>
• Creating IAM roles and policies
• Using monitoring tools like AWS CloudWatch or Azure Monitor
• Cost estimation using cloud calculators</td></tr>
<tr><td><b>Unit Number: 4</b></td><td><b>Title: Cloud-native Application Development & Deployment</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
• Microservices architecture
• Containers and orchestration (Docker & Kubernetes basics)
• Serverless computing (e.g., AWS Lambda)
• CI/CD pipelines for cloud deployment
<b>Practical Applications:</b>
• Deploying a containerized app using Docker
• Configuring a serverless function (AWS Lambda or equivalent)
• Building a CI/CD pipeline using GitHub Actions or AWS CodePipeline</td></tr>
</table>

Text Book

1. *Cloud Computing: Concepts, Technology & Architecture* by Thomas Erl, 2013, Prentice Hall.
2. ***Manvi, S., & Shyam, G. (2021).*** Cloud computing: Concepts and technologies*. Taylor & Francis.*

Classroom Learning Experience
Inside Classroom Learning:

- **Real-Life Contextual Lectures:**
  Concepts like IaaS, PaaS, SaaS, and deployment models are explained using real business use cases.
- **Case-Based Discussions:**
  Students analyze cloud migration, startup hosting, and cost optimization scenarios.
- **Hands-on Demonstrations:**
  Live demos on launching VMs, configuring storage, IAM, and deploying serverless apps.
- **Visualization Walkthroughs:**
  Diagrams and cloud architecture visuals are used to explain scaling, virtualization, and CI/CD.
- **Interactive Quizzes and Polls:**
  Quick assessments to reinforce service models, pricing, and security basics.
- **Collaborative Learning:**
  Group tasks to design cloud setups, choose services, and build simple cloud architectures.

Outside Classroom Learning:

- **Mini Cloud Labs:**
  Students launch VMs, use S3, and deploy simple apps using AWS/Azure free tiers.
- **Reflection Exercises:**
  Short notes on real-life cloud usage (e.g., Drive, Zoom) and key learnings.
- **Practice on Consoles:**
  Explore cloud dashboards to configure IAM, estimate costs, and monitor resources.
- **Tech Diaries:**
  Weekly logs on cloud tasks and troubleshooting steps.
- **Peer Review & Feedback:**
  Exchange and critique cloud designs or deployment plans.
- **Capstone Scenario:**
  Final project to design and deploy a complete cloud-native app with CI/CD.

Introduction to Design Thinking and Prototyping

Program Name	**Bachelor in Computer Applications (BCA)**			
Course Name: **Course Title: 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	Identify user needs and frame design challenges through empathy-driven research.
CO 2	Generate multiple innovative solutions utilizing creative ideation techniques.
CO 3	Develop functional digital prototypes using Figma.
CO 4	Conduct usability evaluations and effectively analyze feedback.
CO 5	Present and communicate design ideas persuasively, demonstrating iterative improvement.

Course Outline:

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

Topics:

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	For safely storing project materials and	10–12 lockers

/ Cabinets	supplies	
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

Semester 3

ANALYSIS AND DESIGN OFALGORITHMS

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: **Analysis and Design of Algorithms**	**Course Code**	**L-T-P**	**Credits**
	ENCS202	3-0-2	4
Type of Course:	Major Course		
Contact Hours	40 hrs		
Version			
Pre-requisite(s), if any: None			

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.

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

COs	Statements
CO 1	Analyze and compare algorithm efficiency using asymptotic notation and mathematical proofs.
CO 2	Implement divide and conquer, greedy, and dynamic programming techniques for problem-solving.
CO 3	Utilize advanced data structures such as tries, Fibonacci heaps, B+ trees, and binomial heaps in algorithm design.
CO 4	Understand NP-completeness, approximation algorithms, and parallel processing techniques for large-scale computations.
C0 5	Solve real-world computational problems using advanced algorithmic techniques and coding.

Course Outline:

Unit Number: 1	**Unit 1: Complexity Analysis & Fundamental Algorithms**	**No. of hours: 10**

Content:

- **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.**

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

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

Topics Covered:

- **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.
✔ **Cloud Network Routing** – Shortest path optimization for real-time traffic management (Google Maps, Uber, Waze).

Unit Number: 3	**Title: Graph Algorithms & Advanced Data Structures (10 Hrs)**	**No. of hours: 10**

Topics Covered:

- **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:

✔ **Database Indexing** – Trie, B+ Trees used in indexing large-scale datasets.
✔ **Blockchain & Cryptography** – Data structures used in ledger verification & encryption.

Unit Number: 4	**NP-Completeness, Approximation Algorithms, & Parallel Processing (10 Hrs)**	**No. of hours: 10**

Topics Covered:

- **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.

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.

Text Books:

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). *Introduction to Algorithms* (4th ed.). MIT Press / McGraw-Hill.
2. Skiena, S. (2020). *The Algorithm Design Manual* (3rd ed.). Springer.

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.

Lab Experiments

List of Experiments

Ex. No	**Experiment Title**	**Mapped CO/COs**
1	**Algorithm Complexity and Foundational Techniques** • **Real-World Scenario:** Develop a benchmarking tool that evaluates the performance of various sorting and searching algorithms on large datasets. • **Sub-Objectives:** • Implement Merge Sort, Quick Sort, Heap Sort,	CO 1

	Counting Sort, and Radix Sort. • Implement Binary Search, Interpolation Search, and Hashing. • Analyze algorithm complexity using Big-O, Omega, and Theta notations. • Solve recurrence relations using Master Theorem. • Perform amortized analysis on dynamic arrays or stacks. **Tools:** Python, time module, matplotlib for plotting performance graphs	
2	**Divide & Conquer, Greedy, and Dynamic Programming** • **Real-World Scenario:** Build an optimization engine for a delivery service that selects fastest paths and optimal schedules using different algorithmic strategies. • **Sub-Objectives:** • Implement Divide & Conquer algorithms like Convex Hull and Closest Pair of Points. • Apply Greedy Algorithms: Huffman Encoding, Kruskal’s & Prim’s MST. • Implement Dynamic Programming problems: 0/1 Knapsack, LCS, Floyd-Warshall. • Compare recursion vs. tabulation vs. memoization. • Visualize decision paths for activity selection problems. **Tools:** Python, networkx, matplotlib	CO 2
3	**Graph Algorithms and Advanced Data Structures** • **Real-World Scenario:** Simulate a decentralized network that manages routes and resources using graph algorithms and advanced trees. • **Sub-Objectives:** • Perform BFS, DFS, and find Strongly Connected Components. • Construct MSTs using Kruskal’s and Prim’s algorithms. • Implement Dijkstra’s, Bellman-Ford, and Floyd-Warshall algorithms. • Create and visualize Tries, B-Trees, B+ Trees. • Use and compare Binomial and Fibonacci Heaps. **Tools**: Python, networkx, custom data structure libraries	CO 3
4	**NP-Completeness and Parallel Approaches** • **Real-World Scenario:** Design and solve constraint optimization problems using approximation and parallel processing approaches. • **Sub-Objectives:** • Formulate and solve N-Queens, Hamiltonian Cycle	C03

	using backtracking and branch and bound. • Identify and implement NP-Complete problems like TSP, Graph Coloring. • Create approximation solutions for Vertex Cover and Set Cover. • Simulate basic parallel processing using Python multiprocessing or MapReduce model. • Evaluate trade-offs between exact and approximate solutions. **Tools:** Python, multiprocessing, PySpark (optional for advanced students)	
5	**Capstone Project: Intelligent Traffic Routing and Resource Allocation System** • **Real-World Scenario**: Build an end-to-end intelligent system that uses graph algorithms, DP, approximation techniques, and complexity benchmarks to optimize real-time urban mobility and cloud deployment. • **Sub-Objectives:** • Design dynamic traffic paths using shortest-path algorithms and graph traversal. • Use DP for cost-efficient schedule management. • Apply approximation methods for constrained resource allocation. • Simulate real-time analytics with performance profiling. • Visualize outcomes through charts and node-edge graphs. **Tools:** Python, networkx, matplotlib, multiprocessing	CO 4

Back-End Web Development

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Back-End Web Development	**Course Code**	**L-T-P**	**Credits**
	ENCS202	3-0-2	4
Type of Course:	Major Course		
Contact Hours	40 hrs		
Version			
Pre-requisite(s), if any: None			

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	Design scalable backend applications using Node.js and Express.js.
CO 2	Build secure, RESTful APIs with full CRUD functionality.
CO 3	Integrate and model data using databases like MongoDB or PostgreSQL.
CO 4	Implement authentication and authorization using JWTs.
C0 5	Apply engineering principles like LLD and write production-grade code.
C0 6	Integrate AI APIs to add intelligent features.
C0 7	Deploy containerized applications to cloud platforms using Docker.

Course Outline:

Unit Number: 1	**Unit 1: Foundations of Node.js**	**No. of hours: 6**

Content:

- 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: 12**

Topics Covered:

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**

Topics Covered:

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: 06**

Topics Covered:

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

Lab Experiments

List of Experiments

Ex.	Experiment	Mapped
No	Title	CO/COs
1	**Simple Express API (Unit 2)** • 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 1
2	**CRUD API with MongoDB (Units 3–4)** • 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()	CO 2

3	**Auth & JWT (Unit 5)** • 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	**AI-Powered Endpoint (Unit 6)** • 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	C03

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	CO 4

Deliverables:

Live deployed backend, GitHub repo with code and README, demo video (optional)

Cybersecurity Essentials and Practices

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Cybersecurity Essentials and Practices	**Course Code**	**L-T- P**	**Credits**
		3-0-2	4
Type of Course:	DSE		
Contact Hours	45 hrs		
Version			
Pre-requisite(s), if any: None			

Course Perspective: This course introduces the fundamentals of cyber security, including threats, system hardening, encryption, and web security. Students will analyze attacks, secure systems, and respond to incidents through practical assignments and tools used in real-world scenarios. The course prepares students for entry-level roles such as SOC Analyst, Junior Pentester, or IT Security Assistant**The Course Outcomes (COs):** On completion of the course the participants will be:

COs	Statements
CO 1	**Explain** core cybersecurity concepts such as CIA triad and threat models.
CO 2	**Perform** basic operating system hardening and network protection.
CO 3	**Apply** cryptographic methods for securing communication.
CO 4	**Detect** and simulate common cyberattacks like SQLi and phishing.

Course Outline:

Unit Number: 1	Title: **Introduction to Cyber Security & Threat Landscape**	**No. of hours: 10**
Content:		

- CIA Triad: Confidentiality, Integrity, Availability
- Common threats: Malware, Phishing, Ransomware, Insider Attacks
- Attacker types: Hacktivists, Insiders, Script Kiddies, Nation-State
- Security domains: Network, Application, Cloud, Endpoint
- Cyber Kill Chain and attack lifecycle
- Risk, vulnerabilities, exposure definitions
- Governance and security policy introduction

Hands-On / Real Case:

- Analyze the Equifax or SolarWinds breach and map to the Cyber Kill Chain
- Create a personal cyber hygiene checklist
- Compare real-world phishing emails vs. secure email policies

Unit Number: 2	**Title: System Security & Hardening**	**No. of hours: 10**

Content:

- Windows and Linux OS security basics
- File system permissions and access control
- Patch management and system updates
- Host-based firewalls: UFW (Linux), Windows Defender
- Antivirus basics and endpoint protection
- Virtualization: Intro to VirtualBox and secure VM usage

Hands-On / Real Case:

- Harden a Linux system using user roles and UFW
- Identify open ports and vulnerable services using nmap
- Configure automatic updates and patch logging on Ubuntu

Unit Number: 3	**Title: Network Security & Cryptography Basics**	**No. of hours: 10**

Content:

- TCP/IP model and network communication
- Introduction to DNS, HTTP, HTTPS
- Symmetric encryption: AES
- Asymmetric encryption: RSA
- Hash functions: MD5, SHA-256
- Public Key Infrastructure, digital signatures
- TLS/SSL basics and VPN concept
- Packet inspection basics using Wireshark

Hands-On / Real Case:

- Use Wireshark to observe plain HTTP vs encrypted HTTPS traffic
- Encrypt and decrypt files using OpenSSL (AES)

Create and verify a file hash to ensure integrity

Unit Number: 4	**Title: Web Security, Attacks & Incident Handling**	**No. of hours: 10**

Content:

- OWASP Top 5 (XSS, SQLi, CSRF, Broken Auth, Misconfigurations)
- Web Application Firewalls (WAF)
- Session hijacking and weak authentication
- Introduction to logs and open-source SIEM concepts
- Basic incident response process (NIST framework)
- Simple threat hunting and log forensics

Hands-On / Real Case:

- Simulate SQLi and XSS using DVWA
- Analyze logs from a mock breach and generate an incident report
- Perform basic log review in ELK or Splunk Free

LAB Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Threat Mapping and Breach Analysis** **Objectives**: • Identify threats using the CIA triad • Analyze a real-world cybersecurity breach • Map the breach to attack lifecycle stages • Build a risk matrix and propose mitigation • Practice using cyber hygiene best practices **Tools**: Internet research, MITRE ATT&CK Navigator, Word/Excel	CO 1
2	**Hardening a Linux System** **Objectives**: • Install and configure a Linux VM • Set user roles and manage file permissions • Enable firewall rules and verify effectiveness • Detect open ports and shut down unnecessary services • Understand impact of security configurations **Tools**: VirtualBox, Ubuntu, UFW, Nmap	CO 2
3	**Secure Communication and Encryption Basics** **Objectives**: • Encrypt/decrypt files using symmetric and asymmetric keys • Observe secure vs insecure protocols using Wireshark • Validate file integrity using SHA-256 hashes • Explore public key certificates in a browser • Understand PKI and digital signatures **Tools**: OpenSSL, Wireshark, Web browsers	CO 3
4	**Vulnerability Testing and Response** **Objectives**: • Deploy and test DVWA (Damn Vulnerable Web App) • Perform SQLi and XSS attacks in a safe environment • Analyze access logs to trace unauthorized activities • Recommend secure coding practices • Build an incident response report **Tools**: DVWA, XAMPP, Firefox/Chrome DevTools, Log files	CO 4

5	**Designing & Defending a Secure Network** **Objectives**: • Create a secure mini-network layout • Harden endpoints (Linux and Windows) • Monitor with Wireshark and analyze threats • Simulate a brute-force login attack • Prepare a final security audit and incident report **Tools**: Draw.io (network diagram), VirtualBox, Wireshark, logs, Nmap	CO 5

Practical Textbooks

1. **Kim, D., & Solomon, M. G.** (2022). *Fundamentals of Information Systems Security* (4th ed.). Jones & Bartlett.
2. **Mark Ciampa.** (2021). *Security+ Guide to Network Security Fundamentals*. Cengage Learning.
3. **OWASP Foundation.** OWASP Top 10 Documentation and DVWA Lab Exercises.

Verbal Ability

Program Name	**Bachelor in Computer Applications (BCA)**			
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 Building**	**No. of hours: 8**
Content: • Topics Covered: o Roots, Prefixes, and Suffixes o Synonyms and Antonyms o Analogies and Word Associations o Idioms and Phrases o Contextual Vocabulary o Techniques for Vocabulary Acquisition		
Unit Number: 2	**Title: Grammar and Usage**	**No. of hours: 8**
Co**ntent:** o Parts of Speech o Tenses and Verb Forms o Subject-Verb Agreement o Articles, Prepositions, Conjunctions o Sentence Structure and Types o Common Grammatical Errors		
Unit Number: 3	**Title: Reading Comprehension**	**No. of hours: 6**
Content: o Identifying Main Ideas and Supporting Details o Understanding Implicit and Explicit Information o Making Inferences and Drawing Conclusions o Analyzing Text Structure and Organization o Different Reading Strategies (Skimming, Scanning, Intensive Reading) o Comprehension of Various Text Types (Narrative, Expository, Argumentative)		
Unit Number: 4	**Title: Written Expression**	**No. of hours: 4**
Content: o Principles of Effective Writing (Clarity, Conciseness, Coherence) o Paragraph Writing and Development o Essay Writing (Structure, Argumentation, Supporting Evidence) o Formal and Informal Writing Styles o Techniques for Improving Writing Fluency and Accuracy		

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/**

Community Service

Program Name	**Bachelor in Computer Applications (BCA)**			
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	To **engaging** students in meaningful social service activities.
CO 2	To **developing** socially responsible engineers
CO 3	To **applying** technical and non-technical skills for the benefit of society.
CO 4	To **fostering** community engagement and support.

Course Outline:

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

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

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.

Lever-Powered Wheelchairs: Develop control applications to enhance mobility for differently-abled individuals.

Assistive Devices: Design simple devices using basic sensors to improve daily living for people with disabilities.

Environmental Monitoring: Build introductory systems using Arduino and web dashboards to raise community awareness about air and water quality.

Eco-Friendly Practices: Create web applications that promote sustainable living and track user participation.

Waste Management: Implement basic data management systems for efficient waste management in local communities.

Energy Optimization: Develop algorithms to optimize energy consumption in households and public buildings.

Water Quality Monitoring: Design systems with sensors and mobile apps to ensure safe drinking water in rural areas.

Smart Agriculture: Create tools using microcontrollers to support farmers with automated irrigation and soil condition monitoring.

Cybersecurity: Implement basic practices to protect sensitive data in sustainable technology applications.

Health Tracking: Develop simple mobile applications to monitor fitness and wellness metrics, benefiting public health initiatives.

Recycling Sorters: Create introductory computer vision projects for sorting recyclables to aid municipal recycling programs.

Environmental Data Analysis: Conduct basic projects on environmental data sets to identify trends and propose solutions for urban planning and conservation efforts.

Chemical Analysis Programs: Create Python programs to support educational institutions.

Electronic Circuits for Physics: Develop circuits to aid students in experiments.

Engineering Mathematics Tools: Design simulation tools to assist in academic research.

Education and Mentorship

Tutoring and Mentorship: Provide tutoring and mentorship to underprivileged children.

Day Camps: Organize and run day camps for low-income children during weekends.

Educational Opportunities for Incarcerated Individuals: Volunteer to provide educational programs and mentorship to incarcerated individuals.

Skill Development Workshops: Conduct workshops to teach various skills to children based on students' expertise.

Community Service and Development

Local Charities and Community Projects: Volunteer with local charities to support community development projects.

Entrepreneurship Initiatives: Help villagers improve their livelihood through entrepreneurship initiatives.

Women Empowerment Programs: Empower women through skill enhancement, awareness programs, and entrepreneurship training.

Digital Awareness Programs: Conduct programs on cybersecurity and social media safety to protect against digital frauds.

Cultural and Traditional Skills

Traditional Skills Learning: Spend time with villagers to learn traditional skills such as pottery, carpentry, weaving, etc.

Artisan Marketing Assistance: Help artisans market their crafts through digital platforms and e-commerce.

Technology for Social Good

Problem-Solving with Technology: Use technology to solve specific problems faced by certain sections of society, such as developing apps for community support.

Community Development Tools: Create tools and resources to assist in community development and problem-solving.

Healthcare Domain

Health Awareness Campaigns: Organize campaigns to raise awareness about hygiene, nutrition, and preventive healthcare.

Medical Camp Assistance: Volunteer at medical camps to support healthcare delivery in underserved areas.

Mental Health Support: Conduct workshops and support groups focusing on mental health awareness and assistance.

Telemedicine Services: Assist in setting up and running telemedicine services for remote communities.

Print Media and Social Platforms

Community Newsletters: Create and distribute newsletters to share important community news and stories.

Social Media Campaigns: Run social media campaigns to raise awareness on various social issues and promote community initiatives.

Other Possible Domains

Environmental Conservation: Participate in tree planting drives, clean-up campaigns, and conservation projects.

Disaster Relief Support: Assist in disaster relief efforts, providing aid and support to affected communities.

Animal Welfare: Volunteer at animal shelters, support animal rescue operations, and promote animal welfare initiatives.

Cultural Preservation: Work on projects to preserve and promote local cultural heritage and traditions.

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.

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.

Student Report Template

Title Page:
Course Title: Community Engagement Service (VAC-II)
Student Name:
Enrollment Number:
Semester: II
Program: BCA
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.

COMPETITIVE CODING-I

Program Name:	**Bachelor in Computer Applications (BCA)**			
Course Name: COMPETITIVE CODING-I	**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.

Summer Internship-I

Program Name:	**Bachelor in Computer Applications (BCA)**		
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:

Industry Internship (Online/Offline):

Students must produce a joining letter at the start and a relieving letter upon completion.

Global Certifications:

Students can opt for globally recognized certification programs relevant to their field of study.

Government/Research Institution Internship:

Students can engage in a research internship with premier government or research organizations such as IITs, IISc, ISRO, DRDO, CSIR, NPL, etc.

On-Campus Industry Internship Programs:

The university will offer on-campus internships in collaboration with industry partners.

Deliverables and Documentation:

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

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

Evaluation Metrics

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

Evaluation Component	Weightage	Description
Internship Report/File	10%	Completeness, professional formatting, relevance to internship tasks.
Video Presentation	20%	Content quality, clarity, communication skills, professional presentation.
Certificate of Completion	70%	Authenticity, completion of internship/certification within stipulated time, relevance to program objectives.

Internship Evaluation Rubric:

S. 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	25–30

		and key findings	
		- 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.

VAC-II

Startup Ideation & Validation

Program Name	**Bachelor in Computer Applications (BCA)**			
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**	**No. of hours: 8**

	Research	
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**

<table>
<tr><td></td><td></td><td></td></tr>
<tr><td colspan="3"><b>Content:</b>
• 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</td></tr>
</table>

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	**Bachelor in Computer Applications (BCA)**			
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.

Digital Communication, Personal Branding & Influence

Program Name	**Bachelor in Computer Applications (BCA)**			
Course Name 7: **Digital Communication, Personal Branding & Influence**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- II			
Pre-requisite(s): 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	**Bachelor in Computer Applications (BCA)**			
Course Name 7: Purposeful Living and Ikigai: Designing a Meaningful Life	**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 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:	**Life Design and Prototyping Your**	**No. of hours: 8**

3	**Future**	
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	**Bachelor in Computer Applications (BCA)**			
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	**Bachelor in Computer Applications (BCA)**			
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**

<table>
<tr><td></td><td></td><td></td></tr>
<tr><td colspan="3">• 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</td></tr>
</table>

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	**Bachelor in Computer Applications (BCA)**			
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 4

Essentials of Object-Oriented Programming with Java

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: **Essentials of Object-Oriented Programming with Java**	**Course Code**	**L-T-P**	**Credits**
		3-0-2	4
Type of Course:	Major Course		
Contact Hours	40 hrs		
Version			
Pre-requisite(s), if any: None			

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	**Develop** Java programs using fundamental programming constructs and object-oriented principles.
CO 2	Handle **exceptions, multithreading, and concurrency** to build efficient applications.
CO 3	Work with **file handling and the Java Collections Framework** to manage and manipulate data.
CO 4	Implement **database connectivity using JDBC** and build simple Java-based data-driven applications.
CO 5	Develop **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:		

Topics Covered: 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: 10**
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: 10**
Content: **Topics Covered:** Java Database Connectivity (JDBC) JDBC API Overview, JDBC Drivers Connecting Java with MySQL/PostgreSQL Executing Queries (Statement, Prepared Statement, 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 Books:

Schildt, H. (2018). *Java: The Complete Reference* (11th ed.). McGraw Hill.

Deitel, P. J., & Deitel, H. M. (2017). *Java: How to Program* (10th ed.). Pearson.

Lab Experiments

	Experiment Title	**Mapped CO/CO**

		s
	Lab Task 1: Java Basics and OOP Concepts Create a class-based **Bank Account System** that supports deposits, withdrawals, and balance checks. **Sub-Objectives:** Setup Java environment (JDK + IDE like IntelliJ or Eclipse). Write basic Java programs using variables, data types, operators, and control flow. Implement user-defined functions (pass by value, recursion). Define classes and objects to simulate bank account functionality. Apply OOP principles: encapsulation, inheritance, polymorphism, abstraction. Demonstrate method overloading and overriding using different account types.	CO1
	Lab Task 2: Exception Handling & Multithreading Build a **Multithreaded Stock Price Fetcher** that retrieves stock prices concurrently and handles exceptions. **Sub-Objectives:** Implement exception handling with try-catch-finally and custom exceptions. Create multiple threads using Thread and Runnable for data fetch simulation. Synchronize shared resources to simulate stock price updates. Use wait(), notify(), and handle deadlock scenarios. Utilize Executors framework for thread pool management. Simulate ATM with concurrent withdrawal and deposit threads with locking.	CO2
	Lab Task 3: File Handling & Java Collections Build a **Contact Management System** that stores and manages contacts using file I/O and collections. **Sub-Objectives:** Read/write text data using FileReader, BufferedReader, and FileWriter. Perform serialization/deserialization of Java objects to save contacts.	CO3

	Use ArrayList, HashMap, and TreeSet to store and organize contacts. Implement search and sort using Collections.sort() and Streams API. Use Iterator and Stream for filtering contact entries. Build a log analyzer to extract error lines from a server log file.	
	Lab Task 4: JDBC & Real-World Applications Create a **Payroll Management System** using JDBC to store, retrieve, and update employee salary data. **Sub-Objectives:** Connect Java with MySQL/PostgreSQL using JDBC. Create and manage tables using SQL from Java. Perform CRUD operations using Statement, PreparedStatement. Implement batch processing and transactions (commit, rollback). Apply coding best practices and optimize queries. Debug and measure application performance using Java profilers.	CO3
	Capstone Project: Online Library Management System **Scenario:** Develop a full-fledged Java application that allows users to register, borrow, and return books using a database backend and file logs. **Integrated Objectives:** Use OOP to model users, books, and transactions. Handle concurrent borrowing using threads and synchronization. Persist data using file serialization and JDBC. Provide user-friendly console-based UI with error handling. Implement search, sorting, and category-wise listing using Java Collections	CO4

Database Management System

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: **Database Management System**	**Course Code**	**L-T-P**	**Credits**
		3-0-2	4
Type of Course:	Major Course		
Contact Hours	40 hrs		
Version			
Pre-requisite(s), if any: Familiarity with computer fundamentals and basic programming.			

Course Perspective:
This course is designed to give students a clear understanding of how to store, manage, and retrieve data using databases. It introduces database concepts, SQL programming, relational design, transaction control, and NoSQL models. The course emphasizes practical applications with real-world datasets and simplified use cases relevant to business, education, and social media.

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

COs	Statements
CO 1	Understand core concepts of database systems and identify DBMS advantages over traditional file systems.
CO 2	Design relational schemas using ER modeling techniques with real-world applicability.
CO 3	Create and manage databases using SQL and PL/SQL commands.
CO 4	Analyze and apply normalization techniques to improve data consistency.
CO 5	Use MongoDB and explore document-based NoSQL solutions for dynamic data use cases.

Course Outline:

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

Content:

Introduction to Databases: File Systems vs. DBMS, key characteristics and use cases
DBMS vs. RDBMS: Codd’s 12 Rules and relational compliance

Entity-Relationship Model: Entities, attributes, relationships, generalization, aggregation
Keys and Constraints: Primary key, foreign key, candidate key, integrity rules
Normalization: Functional dependencies, 1NF, 2NF, 3NF, BCNF
Data Modeling for Analytics: Star and Snowflake schemas
Tools Used: Lucidchart or dbdiagram.io for ER diagrams; MySQL for schema design

Practical Component:
Design a Library Management System from ER model to relational schema using MySQL
Understand schema design in banking and hospital database systems

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

Content:
SQL Basics: DDL (CREATE, ALTER), DML (INSERT, UPDATE), DCL, TCL
Working with Queries: Nested queries, joins (inner, outer, self), set operations
Views and Indexes: Purpose, creation, query optimization using indexes
Advanced SQL Concepts: Window functions, CTEs (WITH clause)
Tools Used: MySQL Workbench, pgAdmin for execution and performance tuning

Practical Component:
Write and execute queries on a sample e-commerce dataset
Perform analytics on customer behavior using window functions and subqueries

Unit Number: 3	Title: File Handling & Java Collections Framework	No. of hours: 10

Content:
Database Transactions: ACID properties, Commit, Rollback, Savepoints
Concurrency & Locking: Isolation levels, deadlocks, locking strategies
PL/SQL Programming: Triggers, stored procedures, cursors, functions
Security in Databases: Roles and privileges, RBAC model, SQL injection prevention
Tools Used: MySQL/PostgreSQL for procedural programming

Practical Component:
Create a secure banking application with transactions and stored procedures
Implement role-based access for different user types (admin, user, guest)

Unit Number: 4	Title: JDBC & Real-World Java Applications	No. of hours: 10

Content:
Performance Optimization: Query plans, indexing (B-Trees, Hash indexes)
NoSQL Basics: Key-value databases, document-oriented databases (MongoDB, Firebase)
Cloud Databases: Deployment and usage of AWS RDS, Google BigQuery, Azure SQL
Big Data Overview: ETL pipelines, data lakes vs. data warehouses
Tools Used: MongoDB Atlas, Firebase Console for NoSQL exploration

Practical Component:
Design a NoSQL database for a social media platform using MongoDB
Understand how cloud databases manage scalability in real-time apps like YouTube or Amazon

Practical Component:

Design a NoSQL database for a social media platform using MongoDB

Understand how cloud databases manage scalability in real-time apps like YouTube or Amazon

Textbooks

Silberschatz, A., Korth, H. F., & Sudarshan, S. (2020). *Database System Concepts* (7th ed.). McGraw-Hill Education.

Pratt, P. J., & Last, M. Z. (2020). *Concepts of database management* (10th ed.). Cengage Learning.

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.

Lab Experiments

Ex. No	**Experiment Title**	
1	**Lab Task 1: College Management System using SQL** Sub-Objectives: Design ER diagram and relational schema for Student, Faculty, Subjects. Create and normalize tables using MySQL/PostgreSQL. Apply DDL and DML operations for CRUD. Use joins and constraints like Primary Key, Foreign Key.	CO 1
2	**Lab Task 2: Clinic Record System using PL/SQL** **Sub-Objectives:** Create tables for Patients, Doctors, Appointments. Write procedures for new appointments and cancellations. Implement triggers to avoid overlapping appointments. Use transactions to ensure data consistency.	CO 2
3	**Lab Task 3: Online Retail Platform with Web Integration** **Sub-Objectives:** Design product and order schema. Integrate DB with web interface using Python Flask/PHP. Perform queries and updates from a web form. Use triggers for stock updates post-purchase.	CO 3
4	**Lab Task 4: NoSQL Data Model with MongoDB** **Sub-Objectives:** Setup MongoDB and create collections for blogs or products. Perform CRUD operations and aggregations. Compare document model with relational structure. Use queries to filter data using operators and conditions.	CO 3

5	**Capstone Project: University ERP Mini System** **Sub-Objectives:** Build schema for academic records, fees, and attendance. Use PL/SQL for functions and triggers. Connect DB with a simple UI to manage student info. Generate reports using SQL and exportable formats.	CO 4

Network Defense and Security Protocols

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Network Defense and Security Protocols	**Course Code**	**L-T- P**	**Credits**
		3-0-2	4
Type of Course:	DSE		
Contact Hours	45 hrs		
Version			
Pre-requisite(s), if any: None			

Course Perspective: This course provides students with foundational knowledge and practical skills in network defense and security management. It covers network security fundamentals, threats and vulnerabilities, firewall management, intrusion detection systems, and incident response procedures to effectively protect and manage network infrastructures.

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

COs	Statements
CO 1	**Understand** network security principles and fundamental networking concepts.
CO 2	**Identify and mitigate** common network threats and vulnerabilities.
CO 3	**Configure and manage** firewalls and secure network devices.
CO 4	**Implement** intrusion detection and prevention mechanisms, and execute effective incident response procedures.

Course Outline:

Unit Number: 1	Title: **Introduction and Fundamentals**	No. of hours: 10
Content: Introduction to Network Security Concepts • Importance and basics of cybersecurity • Information security triad (CIA – Confidentiality, Integrity, Availability) • Security threats: internal vs. external threats Fundamental Networking Principles • TCP/IP protocols and their vulnerabilities • OSI Model layers and security considerations • IP addressing and subnetting basics Overview of Network Infrastructure • Network devices: Routers, Switches, Modems • Wireless network fundamentals and associated risks • Network topologies and security implications		
Unit Number: 2	**Title: Threats, Vulnerabilities, and Firewall Management**	**No. of hours: 10**
Content: Common Network Threats and Vulnerabilities • Types of attacks: Malware, Phishing, Man-in-the-middle, DDoS, SQL Injection • Basics of vulnerability scanning and assessment using Nessus Firewall Implementation and Management • Introduction to firewall types: Packet Filtering, Stateful Inspection, Proxy firewalls • Firewall configuration with open-source tools: pfSense, UFW • Firewall rules creation and management practices Secure Network Configuration and Management • Secure router and switch configuration best practices • VLAN setup and network segmentation • Access Control Lists (ACLs)		
Unit Number: 3	**Title: Intrusion Detection & Prevention**	**No. of hours: 10**
Content: Intrusion Detection and Prevention Systems (IDS/IPS) • IDS vs IPS: Types, functionality, and deployment strategies • Practical implementation using Snort and Suricata • Real-time monitoring and analysis of alerts Basic Security Measures for Routers and Switches • Hardening practices for network devices		

- Password security management
- Firmware updates and patch management

Network Access Control and Authentication Methods

- Authentication methods (Passwords, MFA, Certificates)
- Implementing RADIUS and TACACS+
- Network Access Control (NAC) principles

Unit Number: 4	Title: **Incident Handling and Policy Implementation**	No. of hours: 10

Content:

Incident Response and Basic Network Forensics

- Incident response lifecycle: Preparation, Identification, Containment, Eradication, Recovery, Lessons Learned
- Network forensic analysis with Wireshark and TCPdump
- Log analysis and basic forensic investigation techniques

Security Policies and Network Defence Best Practices

- Creation and implementation of security policies
- Overview of compliance and regulatory frameworks (ISO 27001, GDPR, HIPAA)
- Employee cybersecurity training and awareness strategies

Textbooks:

- Stallings, William. "Network Security Essentials: Applications and Standards," 6th Edition, Pearson Education.
- Bejtlich, Richard. "The Practice of Network Security Monitoring: Understanding Incident Detection and Response," No Starch Press**.**

LAB Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Understanding Networking and Security Fundamentals** **Title**: Exploring Network Topologies, OSI Layers, and TCP/IP Vulnerabilities **Objectives**: 1. Analyze different network topologies using Cisco Packet Tracer. 2. Simulate OSI layer data transmission and identify security risks at each layer. 3. Perform basic IP addressing and subnetting exercises using tools like IP Subnet Calculator. 4. Demonstrate vulnerabilities in TCP/IP (e.g., SYN Flood simulation using Hping3 or equivalent in a lab-safe environment). **Tools Used**: Cisco Packet Tracer / GNS3, IP Subnet Calculator, VirtualBox/Kali Linux, Hping3	CO 1

2	**Threat Assessment and Firewall Configuration** **Title**: Vulnerability Scanning and Firewall Rule Implementation **Objectives**: 1. Scan a test network using **Nessus** or **OpenVAS** to identify vulnerabilities. 2. Install and configure **pfSense** firewall and apply basic rules. 3. Configure **UFW (Uncomplicated Firewall)** on a Linux system and test port blocking/allowing. 4. Set up VLANs and ACLs on simulated switches (Packet Tracer or physical lab) and verify segment isolation. **Tools Used**: Nessus/OpenVAS, pfSense, UFW, Cisco Packet Tracer, Wireshark	CO 2
3	**Intrusion Detection and Network Hardening** **Title**: IDS/IPS Configuration and Secure Device Management **Objectives**: 1. Install and configure **Snort** or **Suricata** for basic intrusion detection. 2. Simulate and analyze suspicious traffic and detect it using IDS. 3. Perform hardening steps on routers/switches (password policies, SSH access, disabling unused ports). 4. Implement **RADIUS** or **TACACS+** on simulated network devices and demonstrate user authentication. **Tools Used**: Snort, Suricata, Wireshark, Cisco Packet Tracer, FreeRADIUS, Ubuntu Server	CO 3
4	**Incident Response and Forensics** **Title**: Incident Response Simulation and Log Forensics **Objectives**: 1. Simulate a network breach and perform a full incident response lifecycle activity. 2. Capture network traffic using **Wireshark** and analyze packets for malicious activity. 3. Use **TCPdump** to monitor traffic in real-time and extract critical evidence. 4. Analyze system logs and generate a forensic investigation report. **Tools Used**: Wireshark, TCPdump, Syslog viewer, Kali Linux, LogParser, Security Onion (optional)	CO 4
5	**Capstone Lab Project – Building a Secure Network Infrastructure** **Title**: Design and Implement a Complete Secure Network Framework **Objectives**: 1. Design a small enterprise network with secure segmentation (VLANs, ACLs). 2. Implement firewall rules using pfSense/UFW. 3. Deploy IDS/IPS and monitor traffic for suspicious	CO 5

	patterns. 4. Document and respond to a simulated incident with a full forensic and policy report. **Tools Used**: Combination of all above: Cisco Packet Tracer/GNS3, pfSense, Snort, Nessus, Wireshark, Ubuntu/Kali Linux	

Communication & Personality Development

Program Name	**Bachelor in Computer Applications (BCA)**			
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.

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/

Minor Project-II

Program Name	**Bachelor in Computer Applications (BCA)**		
COURSE NAME: **Minor Project-I**	**Course Code**	**L-T-P**	**Credits**
	ETCCPR405	0-0-4	2
TYPE OF COURSE:	Project		
PRE-REQUISITE(S), IF ANY: NA			

Course Perspective:

The objective of Minor Project-III for the Bachelor in Computer Applications (BCA) 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-II immerses fourth-semester students in solution design, development, testing, and deployment of different domain 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 Bachelor in Computer Applications (BCA) (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	Progress visualization, CO/PO attainment

Dashboard	
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.

COMPETITIVE CODING -II

Programme Name:	**Bachelor in Computer Applications (BCA)**			
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		

Value Added Course (VAC)-III

Product Deployment & Growth Hacking

Program Name	**Bachelor in Computer Applications (BCA)**			
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	Bachelor in Computer Applications (BCA)			
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

<table>
<tr><td colspan="3">• 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)</td></tr>
<tr><td><b>Unit Number: 2</b></td><td><b>Fundraising Fundamentals</b></td><td><b>No. of hours: 8</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
• 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</td></tr>
<tr><td><b>Unit Number: 3</b></td><td><b>Pitch Deck Structure & Financial Storytelling</b></td><td><b>No. of hours: 8</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
• 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</td></tr>
<tr><td><b>Unit Number: 4</b></td><td><b>Live Pitching & Investor Simulation</b></td><td><b>No. of hours: 6</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
• 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</td></tr>
</table>

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	**Bachelor in Computer Applications (BCA)**			
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

Digital Entrepreneurship: Monetizing Skills in the Creator Economy

Program Name	**Bachelor in Computer Applications (BCA)**			
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

Design Thinking for Social Innovation

Program Name	**Bachelor in Computer Applications (BCA)**			
Course Name 5: Design Thinking for Social Innovation	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	VAC	2-0-0	2	30
Type of Course:	VAC- III			
Pre-requisite(s):				

Course Perspective. This course introduces students to design thinking as a human-centered framework to tackle complex social challenges. Learners will explore empathy-driven problem-solving, ideation, prototyping, and iterative development, applying these principles to real-world issues such as sustainability, public health, education, and inclusion.

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

COs	**Statements**
CO1	**Understanding** the design thinking process and its relevance to social impact
CO2	**Conducting** empathy research and identify unmet needs in marginalized communities
CO3	**Generating** creative, user-centric solutions to complex social problems
CO4	**Prototyping** and test social innovations using low-cost methods
CO5	**Developing** and present an implementable impact solution

Course Outline:

Unit Number: 1	**Introduction to Design Thinking & Social Impact**	**No.of hours: 8**
Content: • Principles and stages of design thinking • Systems thinking vs linear problem solving • Role of empathy, ethics, and equity in social innovation		

• Challenges in the social sector: complexity, context, and constraints • Examples from IDEO.org, Stanford d.school, and Ashoka Fellows		
Unit Number: 2	**Empathy & Need-Finding**	**No. of hours: 8**
Content: • Principles and stages of design thinking • Systems thinking vs linear problem solving • Role of empathy, ethics, and equity in social innovation • Challenges in the social sector: complexity, context, and constraints • Examples from IDEO.org, Stanford d.school, and Ashoka Fellows		
Unit Number: 3	**Ideation & Prototyping**	**No. of hours: 8**
Content: • Divergent and convergent thinking • Ideation tools: brainwriting, worst possible idea, SCAMPER • Low-fidelity prototyping: paper, cardboard, digital mockups • Feedback loops and rapid iteration • Testing prototypes with real users		
Unit Number: 4	**Social Innovation & Impact Deployment**	**No. of hours: 6**
Content: • Impact canvas and theory of change • Pilot testing and measuring social impact • Design for scalability and sustainability • Communicating solutions: storytelling and pitch deck • Building partnerships with NGOs, government, or communities		

Learning Experiences

Inside Classroom Learning*:*

1. Empathy-based workshops on journey mapping, stakeholder analysis, and POV.
2. Brainstorming techniques like SCAMPER, worst idea, and visual ideation.
3. Case studies of Embrace Warmer, Aravind Eye Care, and Recyclebank.
4. In-class prototyping labs with low-cost materials and group critiques.

Outside Classroom Learning

1. Field empathy interviews and ethnographic observations.
2. —How Might We‖ challenge framing and ideation outside the classroom.
3. Prototype testing and refinement with real users or peer groups.
4. Final capstone presentations showcasing innovation solution with impact plan.

Textbooks and Reference Books:

1. Brown, T. (2009). *Change by design: How design thinking creates new alternatives for business and society*. Harvard Business Press.
2. Liedtka, J., & Ogilvie, T. (2011). *Designing for growth: A design thinking toolkit for managers*. Columbia University Press.

Case Studies to Demonstrate

1. **Embrace Infant Warmer –** Low-cost innovation for premature babies in rural India
2. **Aravind Eye Care** – High-quality, low-cost healthcare delivery
3. **Digital Green –** Using tech for agricultural knowledge sharing
4. **Recyclebank –** Gamified recycling awareness
5. **Selco Solar –** Sustainable solar solutions for underserved areas

Lab/Hands-On Tasks

- Conduct 2 empathy interviews on a chosen social theme
- Create a journey map for a target stakeholder
- Frame 3 HMW problem statements based on field research
- Ideate 15+ ideas using brainwriting in groups
- Build and test a low-fidelity prototype for a social innovation
- Final capstone: pitch your impact idea with a prototype and implementation plan

No-Code & Low-Code Product Development

Program Name	**Bachelor in Computer Applications (BCA)**			
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	**Bachelor in Computer Applications (BCA)**			
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

Semester 5

Operating Systems

Program Name	**Bachelor in Computer Applications (BCA)**			
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.

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:

<table>
<tr><td><b>Unit Number: 1</b></td><td><b>Foundations, Process Model, & Command-Line Essentials</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3">• 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.</td></tr>
<tr><td><b>Unit Number: 2</b></td><td><b>Title: Concurrency, Scheduling & Synchronization</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3">• 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</td></tr>
</table>

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	CO 2

	1. Write a command-line simulator that reads a file of ⟨arrival, burst⟩ pairs and computes average turnaround and waiting time for FCFS, SJF (non-pre-emptive), Round-Robin (2 ms slice) and static-priority. 2. Generate a simple ASCII Gantt chart for each policy. 3. Compile two CPU-bound loops (spinA, spinB). Launch them at default priority, measure per-process CPU time with pidstat or top. 4. Re-run with nice -n 10 spinB; note the change. 5. Set spinB to real-time SCHED_RR with chrt -r -p 80 <pid> and record context-switch counts using perf stat -e context-switches. Explain each difference in a short note.	
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	CO 3

	again batching fsync() every 100 writes; record the speed-up and discuss durability trade-off.	
5	**Experiment 5 — —Capstone: Tiny Guestbook Service‖** ***(Units 1-4 combined)*** Main problem Combine your earlier work into a small but complete network service and tune it end-to-end. Sub-problems 1. Reuse the thread-pool server pattern; each POST (name,msg) stores an entry via the log-safe file store, then returns the last 10 messages. 2. Replace default malloc with a small fixed-size pool (adapt ideas from Experiment 3). 3. Create a systemd service file, add a dedicated user, and set restrictive file permissions. 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.	CO 4

System Design Essentials

Programme Name:	**Bachelor in Computer Applications (BCA)**			
Course Name: System Design Essentials	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		3-0-2	4	45
Type of Course:	Major			
Pre-requisite(s), if any: OOP, Computer Networks, Web Development				

Course Perspective: This course helps BCA students bridge theory with industry-ready practices in software architecture. It introduces both high-level and low-level system design skills essential for software engineering interviews and scalable app development. With hands-on labs and a capstone project, learners gain experience building systems that can handle real-world loads and failures.

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

COs	Statements
CO 1	**Understand** how to break down product requirements into scalable systems
CO 2	**Build** design diagrams and plan architecture for real-time applications
CO 3	**Use** design patterns, testable code principles, and sequence diagrams
CO 4	Handle distributed computing challenges such as consistency, availability, and monitoring

Course Outline:

Unit Number: 1	**Title: System Basic & Interview Interview Readines**	**No. of hours: 15**
Content: • HLD vs LLD roles in interviews • Functional and non-functional requirements • Understanding request flow (DNS → Load Balancer → Web → DB → CDN) • Key components: Stateless web apps, DB replicas, load balancers • Metrics: Requests per second, latency targets, storage estimates • **Lab Exercises:** • Use DevTools and traceroute to trace real HTTP requests • Python script to convert QPS + payload size into egress bandwidth and DB IOPS • Build a Mini URL Shortener with Flask + SQLite • Postman task to compare REST vs GraphQL responses • Whiteboard mock session: —What happens when you press Enter?‖		
Unit Number: 2	**Title: D Level D**	**No. of hours: 12**
Content: Designing for Scale (High-Level Design)		

- OLTP vs NoSQL design trade-offs
- Sharding, replication, consistency models
- Redis cache patterns: cache-aside, TTL, eviction
- Message queues (Kafka, RabbitMQ), async workers
- Token-based access control and rate-limiting gateways
- **Lab Exercises:**
- Simulate user sharding using Python scripts
- Redis caching for improved throughput
- Kafka-based email sender with latency tracking
- Lua-based NGINX rate limiter PoC
- Add artificial DB delays and log effects

Unit Number: 3	**Title:Component Design & Software Patterns**	**No. of hours: 10**

Content

- SOLID principles in practice
- UML diagrams (class, sequence)
- Interview-level design patterns: Singleton, Factory, Observer, Builder, Strategy
- Creating unit-testable architecture and using dependency injection
- **Lab Exercises:**
- UML diagrams for Parking Lot system
- Elevator logic with unit-tested state transitions
- Observer-based multi-channel notification push
- 30-min live code challenge with design patterns
- Refactor Shortener to follow Interface Segregation and 80% test coverage

Unit Number: 4	**Title: Distributed Systems and Mock Interviews**	**No. of hours: 8**

Content:

- CAP theorem, BASE, circuit breakers, retries
- Consistent hashing, quorum reads/writes, leader election
- Monitoring system health: RED metrics, Grafana, Prometheus
- Debugging slow responses, hot shards, GC pauses
- Conduct mock design interviews and receive peer feedback
- **Lab Exercises:**
- Visualize consistent hashing key migrations
- Build monitoring dashboards with alerts
- Observe leader failover using raft-kv demo
- Analyze uneven shard loads using SQL queries
- Peer mock panel for —Real-Time Chat App‖ design

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*

Applied Cryptography for Cybersecurity

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Applied Cryptography for Cybersecurity	**Course Code**	**L-T- P**	**Credits**
		3-0-2	4
Type of Course:	DSE		
Contact Hours	45 hrs		
Version			
Pre-requisite(s), if any: None			

Course Perspective: This course offers a practical and foundational understanding of classical and modern cryptography. It covers core cryptographic algorithms, encryption methods, hashing, digital signatures, and secure communication protocols. Students will implement and analyze cryptographic systems using tools like Python libraries, OpenSSL, CrypTool, and GPG, preparing them to evaluate and apply security techniques in real-world applications without redundancy from prior cybersecurity courses.

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

COs	Statements
CO 1	**Understanding** mathematical concepts and classical cipher systems.
CO 2	**Applying** modern symmetric and asymmetric encryption techniques.
CO 3	**Implementing** hash functions, HMAC, and digital signature algorithms.
CO 4	**Using** cryptographic tools for encryption, key generation, and secure messaging.

Course Outline:

Unit Number: 1	**Title: Classical Cryptography & Mathematical Foundations**	**No. of hours: 10**
Content:		

Topics:

- Cryptographic Terminology: plaintext, ciphertext, keys
- Classical Ciphers: Caesar, Monoalphabetic, Vigenère
- Attacks: Frequency analysis, brute-force
- Mathematical Foundations:
  - Modular arithmetic, GCD, multiplicative inverse
  - Euler’s and Fermat’s theorems
  - Introduction to number theory

Hands-On / Real Case:

- Break a Vigenère cipher using frequency analysis on English text
- Use Python to encrypt/decrypt messages using Caesar and Vigenère
- Solve modular problems simulating cryptographic conditions

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

Content:

- Block vs. Stream Ciphers
- DES and AES: Structure, round functions
- AES Modes of Operation: ECB, CBC, CFB, OFB, CTR
- Padding oracle and replay attacks
- Key reuse issues and IV-based weaknesses
- Key distribution and management

Hands-On / Real Case:

- Encrypt sample text and images using AES ECB and CBC in Python
- Demonstrate the impact of IV reuse and ECB pattern leak
- Capture ciphertext to analyze structure leakage

Unit Number: 3	**Title:Public Key Cryptography & Key Exchange**	**No. of hours: 10**

Content:

- Symmetric vs Asymmetric Encryption
- RSA: Math, key generation, operations
- Diffie-Hellman Key Exchange
- ElGamal Encryption
- ECC (Elliptic Curve Cryptography) basics
- Public Key Infrastructure (PKI) and Certificate Authorities

Hands-On:

- Generate RSA keys using OpenSSL/GPG
- Simulate RSA and DH in Python for secure chat
- Encrypt and sign files to validate real-world secure mail scenarios

Unit Number: 4	**Title: Hash Functions, MACs, Digital Signatures & Protocols**	**No. of hours: 10**

Content:

- Cryptographic Hashes: MD5, SHA-1, SHA-2, SHA-3
- Message Authentication Codes (HMAC)
- Attacks: Birthday attack, preimage, collisions
- Digital Signatures: RSA, DSA, ECDSA
- TLS/SSL, PGP, and S/MIME
- Introduction to Zero-Knowledge Proofs and Blockchain relevance

Hands-On:

- Hash large files using SHA-256 and detect integrity change
- Create and verify digital signatures using GPG
- Observe TLS handshake in Wireshark and extract certificate details

LAB Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Breaking Classical Ciphers** **Objectives**: • Encrypt messages with Caesar and Vigenère manually and via Python • Perform frequency analysis to decrypt a cipher • Explore weaknesses of substitution ciphers **Tools**: Python, CrypTool, Pen & Paper (manual practice)	CO 1
2	**AES Encryption Modes in Practice** **Objectives**: • Encrypt files/images with AES in ECB vs CBC • Detect encryption patterns visually (especially with ECB) • Demonstrate IV reuse and simulate replay attack logic **Tools**: Python Cryptography / PyCryptoDome, OpenSSL	CO 2
3	**Public-Key Cryptography and Key Exchange** **Objectives**: • Generate and manage RSA key pairs • Encrypt/decrypt messages using GPG/OpenSSL • Simulate Diffie-Hellman key exchange via Python • Understand key trust and certificate validation **Tools**: GPG, OpenSSL, Python, Linux CLI	CO 3
4	**Digital Signatures and Message Integrity** **Objectives**: • Generate hashes and verify document integrity • Sign and verify digital files using GPG • Implement HMAC logic and verify MAC on tampered data **Tools**: GPG, Python, Wireshark	CO 4
5	**Capstone Assignment: Design and Implement a Secure Communication Protocol** **Objectives**: • Design a hybrid cryptographic protocol combining AES + RSA + HMAC • Implement complete workflow: encryption, key exchange, digital signature • Simulate a secure file transfer or messaging app prototype • Identify vulnerabilities and provide mitigation recommendations **Tools**: Python, GPG, OpenSSL, Draw.io (for protocol diagramming)	CO 5

Recommended Textbooks

1. **William Stallings** (2023). *Cryptography and Network Security: Principles and Practice* (8th Edition). Pearson.
2. **Christof Paar & Jan Pelzl** (2010). *Understanding Cryptography: A Textbook for Students and Practitioners*. Springer.
3. **Bruce Schneier** (2015). *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. Wiley.

Building Secure Applications

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Building Secure Applications	**Course Code**	**L-T- P**	**Credits**
		3-0-2	4
Type of Course:	DSE		
Contact Hours	45 hrs		
Version			
Pre-requisite(s), if any: None			

Course Perspective: This course enables students to develop secure software and web applications by integrating cybersecurity practices into the software development lifecycle. The course emphasizes secure coding, threat modeling, authentication mechanisms, secure APIs, and vulnerability mitigation following OWASP guidelines. Students will build, break, fix, and secure real-world application components using modern tools and secure frameworks.

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

COs	Statements
CO 1	**Explain** secure software development principles and the SSDLC lifecycle.
CO 2	**Identify**, exploit, and mitigate common web application vulnerabilities.
CO 3	**Apply** secure coding techniques for input validation, session handling, and storage.
CO 4	**Implement** secure authentication and authorization mechanisms in applications.
CO 5	**Use** security testing tools to assess and secure application code and APIs.

Course Outline:

Unit Number: 1	**Title: Introduction to Application Security & SSDLC**	**No. of hours: 8**
Content:		

- Secure Software Development Lifecycle (SSDLC)
- Software vulnerabilities vs. bugs
- Threat modeling basics (STRIDE, DREAD)
- OWASP Top 10 overview
- Security by design and defense in depth

Hands-On / Real Case:

- Identify threats in an e-commerce app using STRIDE
- Review insecure login code and suggest fixes
- Map a software feature to the SSDLC phases

Unit Number: 2	**Title: Input Validation and Secure Coding Practices**	**No. of hours: 8**

Content:

- Input validation and output encoding
- SQL Injection and XSS vulnerabilities
- Secure coding patterns in Python/PHP/JavaScript
- Secure error handling and logging
- Code smells and anti-patterns

Hands-On / Real Case:

- Inject and patch SQLi in a login form
- Simulate XSS and sanitize using encoding
- Use ESLint or Bandit to scan for coding issues

Unit Number: 3	**Title: Authentication, Authorization, and Session Management**	**No. of hours: 8**

Content:

- Authentication types: Basic, Form-based, Token-based
- Secure password storage (bcrypt, Argon2)
- Session management and cookie security
- Access control: RBAC, MAC, DAC
- Preventing broken authentication attacks

Hands-On / Real Case:

- Create a login system with salted password hashes
- Simulate session hijacking and apply mitigations
- Implement user roles using JWT or session tokens

Unit Number: 4	**Title: API and Web Application Security**	**No. of hours: 8**

Content:

- RESTful API security best practices
- Insecure Direct Object Reference (IDOR)
- CORS, CSRF, and secure headers
- API rate limiting and input validation
- Secure file uploads and data handling

Hands-On / Real Case:

- Harden a public REST API with JWT and input checks
- Identify and prevent CSRF attacks
- Analyze headers using Burp Suite or browser tools

Unit Number: 5	**Title: Application Security Testing and**	**No. of hours: 8**

<table>
<tr><td></td><td><b>Deployment</b></td><td></td></tr>
<tr><td colspan="3"><b>Content:</b>
• Static and dynamic analysis of applications
• SAST and DAST tools overview (SonarQube, ZAP, Burp Suite)
• Secure DevOps principles
• Secure deployment practices (HTTPS, patching, container security)
• Open-source libraries and dependency scanning
<b>Hands-On / Real Case</b>:
• Use OWASP ZAP to test a web app
• Interpret scan results and fix vulnerabilities
• Deploy a secure web app with HTTPS on localhost</td></tr>
</table>

LAB Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Threat Modeling and SSDLC Mapping** **Objectives**: • Create a basic threat model using STRIDE • Map SSDLC to a given feature (e.g., login page) • Identify potential security checkpoints **Tools**: OWASP Threat Dragon, Draw.io	CO 1
2	**Identifying and Fixing Input Vulnerabilities** **Objectives**: • Inject SQLi and XSS in a test environment • Implement proper sanitization and escaping • Validate secure coding using Bandit/ESLint **Tools**: OWASP Juice Shop, Flask/Django or PHP	CO 2
3	**Authentication and Access Control Implementation** **Objectives**: • Implement a login system with bcrypt-hashed passwords • Create secure user sessions with expiry • Implement role-based access control **Tools**: Flask/Django, Postman, JWT	CO 3
4	**Securing REST APIs** **Objectives**: • Design a basic API and protect endpoints • Prevent IDOR and add secure headers • Simulate CSRF and apply tokens **Tools**: Python REST API (Flask), Postman, Browser DevTools	CO 4

5	**Capstone Lab: Secure Application Mini Project** **Title**: “Secure Complaint Submission Portal” **Objectives**: • Develop a full web app with login and complaint submission • Secure input, authentication, and file handling • Test using OWASP ZAP and generate a vulnerability report • Document all security measures used	CO 5

Recommended Textbooks

1. **OWASP Foundation**. OWASP Top 10 and Application Security Verification Standard (ASVS).
2. **Neil Bergman** (2023). *Web Application Security: Exploitation and Countermeasures*. No Starch Press.
3. **Mark Merkow** (2021). *Secure Coding in C and C++*. Pearson.

Arithmetic and Reasoning Skills

Program Name	**Bachelor in Computer Applications (BCA)**			
Course Name: **Arithmetic and Reasoning Skills**	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		2-0-0	2	40
Type of Course:	AEC			
Pre-requisite(s): None				

Course Perspective: The course aims to improve basic arithmetic skills, speed, and accuracy in mental calculations, and logical reasoning. These abilities are essential for a strong math foundation, helping students succeed in academics and various practical fields.

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

COs	Statements
CO 1	Understanding arithmetic algorithms required for solving mathematical problems.
CO 2	Applying arithmetic algorithms to improve proficiency in calculations.
CO 3	Analyzing cases, scenarios, contexts and variables, and understanding their inter-connections in a given problem.
CO4	Evaluating & deciding approaches and algorithms to solve mathematical & reasoning problems.

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

Course Outline:

Unit Number: 1	**Title: Mathematical Essentials**	**No. of hours: 15**
Content Summary: Vedic Maths, Classification of Numbers and Divisibility Rule, Percentage, Ratio and Proportion		
Unit Number: 2	**Title: Fundamentals of Logical Reasoning**	**No. of hours: 6**

Content Summary: Blood Relations, Direction Sense, Coding Decoding		
Unit Number: 3	**Title:** Elementary Quantitative Skills	**No. of hours: 18**
Content Summary: Simple and Compound Interest, Average, Partnership, Time and Work, Time Speed & Distance		
Unit Number: 4	**Title:** Advanced Quantitative Skills	**No. of hours: 6**
Content Summary: Permutation & Combination, Probability		

Learning 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/ Videos /Podcast etc.

- 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:	**Bachelor in Computer Applications (BCA)**			
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:	**Bachelor in Computer Applications (BCA)**			
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 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.

Semester 6

Introduction to Computer Networks

Programme Name:	**Bachelor in Computer Applications (BCA)**			
Course Name: Introduction to 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	CO1

	• 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)	
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 • 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	CO3, CO4
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)	CO4

	• 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)	
5	**Capstone Project: Design, Simulate, and Troubleshoot a Departmental Network** **Real-World Scenario:** You are hired by a university IT team to build a network for departments like Admin, Library, and Labs. The network must support dynamic IP, secure HTTP access, proper routing, and subnetting across departments. **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	CO4

Agile Software Engineering Essentials

Programme Name:	**Bachelor in Computer Applications (BCA)**			
Course Name: Agile Software Engineering Essentials	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
		3-0-2	4	
Type of Course:	Audit/credit Course			
Pre-requisite(s), if any: Basic understanding of software engineering principles and software development life cycles.				

Course Perspective: This course introduces the fundamental concepts, principles, and practices of Agile software development. It emphasizes iterative and incremental development, collaboration, and responding to change. Students will learn about popular Agile methodologies and gain practical insights into applying Agile techniques for efficient and flexible software projects.

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

COs	**Statements**
CO 1	**Understand** the core values, principles, and mindset of Agile software development and contrast it with traditional software development life cycles.
CO 2	**Explain** and apply key Agile methodologies such as Scrum and Extreme Programming, including their roles, events, and artifacts.
CO 3	**Apply** Agile planning and estimation techniques, including writing user stories, managing backlogs, and participating in sprint planning.
CO 4	**Understand** and evaluate essential Agile practices like Test-Driven Development (TDD), Continuous Integration (CI), and their role in delivering high-quality software.

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

Course Outline:

<table>
<tr><td>Unit Number:
1</td><td><b>Title: Introduction to Agile Software Engineering</b></td><td>No. of hours:
10</td></tr>
<tr><td colspan="3"><b>Topics Covered:</b>
• The Agile Manifesto and its principles
• History and evolution of Agile
• Comparing Agile with Waterfall and other traditional models, Lean principles
• Introduction to prominent Agile methodologies (Scrum, Extreme Programming, Kanban).
<b>Real-World Use Case:</b>
Analyzing a case study of a failed software project using a traditional model and discussing how an Agile approach could have yielded better results.</td></tr>
<tr><td>Unit Number:
2</td><td><b>Title: Agile Methodologies and Practices</b></td><td>No. of hours:
12</td></tr>
<tr><td colspan="3"><b>Topics Covered:</b>
• Deep dive into Scrum: Roles (Product Owner, Scrum Master, Development Team)
• Events (Sprint Planning, Daily Scrum, Sprint Review, Sprint Retrospective) Artifacts (Product Backlog, Sprint Backlog, Increment)
• Introduction to Extreme Programming (XP): Core practices (Pair Programming, TDD, Continuous Integration).
<b>Real-World Use Case:</b>
Simulating a Scrum sprint including daily stand-ups, sprint planning, and review meetings for a small project.</td></tr>
<tr><td>Unit Number:
3</td><td><b>Title: Agile Planning, Estimation, and Requirements</b></td><td>No. of hours:
10</td></tr>
<tr><td colspan="3"><b>Topics Covered:</b>
• Agile Requirements: User Stories - structure, characteristics, and writing effective user stories.
• Backlog Management: Product Backlog creation and refinement, Sprint Backlog.
• Agile Estimation Techniques: Planning Poker, Relative Sizing.
• Release Planning and Sprint Planning.
<b>Real-World Use Case:</b>
Creating a Product Backlog and writing user stories for a simple application, followed by a planning poker session to estimate effort.</td></tr>
<tr><td>Unit Number: 4</td><td><b>Title: Agile Quality and Beyond</b></td><td>No. of hours: 10</td></tr>
<tr><td colspan="3"><b>Topics Covered:</b>
• Agile Testing: Principles of agile testing, Test-Driven Development (TDD), Acceptance Test-Driven Development (ATDD).
• Continuous Integration (CI) and Continuous Delivery (CD) concepts.
• Introduction to Agile Design and Architecture.
• Scaling Agile (Introduction to frameworks like SAFe).
<b>Real-World Use Case:</b>
Implementing a small feature using TDD and setting up a basic Continuous Integration pipeline for a project.</td></tr>
</table>

Text and Reference Books:

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

Lab Experiments

Lab Task 1: Understanding Agile through Comparison & Case Study Sub-

Objectives:

1. Analyze the Agile Manifesto and map its principles to real-world practices.

2. Compare Agile and Waterfall models through a visual flowchart or table.

3. Review a failed project case (e.g., FBI Virtual Case File) and identify missteps using traditional models.

4. Conduct a group discussion: "What would Agile have done better?"

Real-World Scenario:

Teams will dissect a real or fictional failed software project (e.g., a delayed university management portal) and propose how Agile could have improved delivery using Scrum or Kanban.

Lab Task 2: Simulating Scrum and XP Practices Sub-Objectives:

1. Assign Scrum roles within student groups and simulate a sprint for a to-do list app.

2. Conduct Sprint Planning, Daily Stand-ups (mock), Sprint Review, and Retrospective.

3. Implement pair programming for one module using XP principles.

4. Demonstrate Continuous Integration using GitHub Actions or a basic Jenkins setup.

Real-World Scenario:

Teams simulate an Agile sprint to develop a simple blog website, practicing daily Scrum, XP pair coding, and using GitHub for CI.

Lab Task 3: User Story Writing & Agile Estimation Sub-Objectives:

1. Write user stories for a mini project (e.g., College Event Management App).
2. Organize Product and Sprint Backlogs using Trello or Jira.
3. Conduct Planning Poker using cards or online tools to estimate story points.
4. Prioritize user stories using MoSCoW or Value vs. Complexity matrix.

Real-World Scenario:

Students plan a College Festival App: writing user stories (e.g., ―As a student, I want to register for events…‖), estimating tasks in a sprint, and prioritizing features based on event timelines.

Lab Task 4: Agile Testing, CI/CD, and Quality Practices Sub-Objectives:

1. Implement a feature using Test-Driven Development (TDD) in Python/JavaScript.
2. Create basic test cases (unit and acceptance) using a testing framework (e.g., JUnit, pytest).
3. Set up a basic CI pipeline using GitHub Actions, Travis CI, or Jenkins.
4. Discuss scalability: Introduction to SAFe using diagrams and practical reflections.

Real-World Scenario:

Students build and test a login form using TDD, push it to GitHub, and automate testing with CI workflows, simulating a production-quality Agile pipeline.

Capstone Project: End-to-End Agile Simulation

Title: Develop & Deliver a Mini Project Using Agile Lifecycle

Sub-Objectives:

1. Select a real-world app idea (e.g., Online Complaint Portal).
2. Write EPICs and User Stories, estimate and prioritize them.
3. Conduct multiple simulated sprints, each with a working increment.
4. Apply XP coding, CI/CD pipeline, and perform retrospectives after each sprint.
5. Deliver a functional demo with a team presentation.

Real-World Scenario:

A local NGO wants an online portal for collecting public grievances. Your team acts as the Agile team responsible for designing, building, testing, and demonstrating the MVP using Agile practices end-to- end.

Ethical Hacking

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Ethical Hacking	**Course Code**	**L-T- P**	**Credits**
		3-0-2	4
Type of Course:	DSE		
Contact Hours	45 hrs		
Version			
Pre-requisite(s), if any: None			

Course Perspective: This course provides comprehensive knowledge and hands-on practical skills required for ethical hacking and penetration testing. Students will learn to identify vulnerabilities, exploit systems ethically, and utilize advanced cybersecurity tools to strengthen security posture effectively.

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

COs	Statements
CO 1	Demonstrate comprehensive knowledge of ethical hacking principles and methodologies.
CO 2	Perform footprinting, scanning, and enumeration techniques effectively.
CO 3	Identify and exploit various vulnerabilities in web applications, databases, and networks.
CO 4	Apply advanced cybersecurity tools and techniques, including cloud and IoT security.
CO 5	Demonstrate comprehensive knowledge of ethical hacking principles and methodologies.

Course Outline:

Unit Number: 1	**Title: Ethical Hacking Fundamentals**	**No. of hours: 8**
Content:		

- Introduction to Ethical Hacking and Penetration Testing
  - Principles and scope of ethical hacking
  - Types of penetration tests
  - Ethical and legal considerations
- Footprinting and Reconnaissance Techniques
  - Passive and active information gathering
  - OSINT (Open Source Intelligence)
  - Tools: Google Dorks, Whois, Shodan, Maltego
- Network Scanning and Enumeration
  - Network scanning methodologies
  - Port and service enumeration
  - Vulnerability assessment tools: Nmap, Nessus, Wireshark

Lab Activities:

1. OSINT gathering and footprinting using Maltego and Shodan.
2. Network scanning, enumeration, and vulnerability assessment using Nmap and Nessus.

Unit Number: 2	**Title: Vulnerability Exploitation Techniques**	**No. of hours: 8**

Content:

- Vulnerability Identification and Exploitation
  - Types of vulnerabilities
  - Exploitation fundamentals
  - CVE identification and exploitation
- Web Application and Database Hacking
  - OWASP Top 10 vulnerabilities
  - SQL injection, Cross-site scripting (XSS)
  - Tools: Burp Suite, SQLmap
- Wireless Network Security and Hacking Techniques
  - Wireless network vulnerabilities
  - Wireless penetration testing
  - Tools: Aircrack-ng, Wi-Fi Pineapple

Unit Number: 3	**Title: Advanced Attack Methods**	**No. of hours: 8**

Content:

- Malware Threats and Evasion Techniques
  - Types of malware and their behaviors
  - Malware evasion and obfuscation techniques
  - Tools: VirusTotal, Metasploit evasion modules
- Social Engineering Tactics
  - Principles of social engineering
  - Phishing attacks and countermeasures
  - Tools: SET (Social Engineering Toolkit)
- Session Hijacking and Denial of Service (DoS/DDoS) Attacks
  - Understanding session management vulnerabilities
  - Session hijacking tools: Ettercap
  - DoS/DDoS attacks and mitigation strategies

Unit Number: 4	**Title: Emerging Technologies & Penetration Testing Tools**	**No. of hours: 8**

Content:

- Cloud Computing Security and IoT Vulnerabilities
  - Cloud security risks and mitigation
  - IoT device vulnerabilities and attacks
- Cryptography and Encryption Cracking Techniques
  - Basics of cryptographic protocols and vulnerabilities
  - Encryption cracking methods and tools
- Penetration Testing Frameworks and Tools
  - Kali Linux and its essential penetration testing tools
  - Metasploit framework for advanced exploitation
  - Practical penetration testing exercises

Textbooks:

- Walker, Matt. "CEH Certified Ethical Hacker All-in-One Exam Guide," 5th Edition, McGraw-Hill Education.
- Kim, Peter. "The Hacker Playbook 3: Practical Guide To Penetration Testing," CreateSpace Independent Publishing Platform.

LAB Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Reconnaissance, Footprinting & Scanning** **Objectives**: • Perform active and passive information gathering • Use OSINT tools for domain and network analysis • Conduct network scanning and service enumeration **Activities**: 1. Perform footprinting using **Whois**, **Shodan**, and **Maltego** 2. Conduct OS scanning and port scanning using **Nmap** 3. Run a vulnerability assessment scan using **Nessus** 4. Analyze traffic patterns using **Wireshark** **Tools**: Maltego, Shodan, Nmap, Nessus, Wireshark	CO 1
2	**Web & Wireless Exploitation** **Objectives**: • Identify and exploit common web and database vulnerabilities • Simulate wireless hacking in a controlled environment **Activities**: 1. Exploit **SQLi** and **XSS** vulnerabilities on **DVWA** 2. Use **Burp Suite** and **SQLmap** for exploitation 3. Capture and crack WPA2 handshake using **Aircrack-ng** 4. Analyze insecure wireless configurations **Tools**: DVWA, Burp Suite, SQLmap, Aircrack-ng	CO 2
3	**Malware, Social Engineering, and DoS Attacks** **Objectives**:	CO 3

	• Understand payload creation and evasion • Perform a simulated phishing attack • Conduct a DoS simulation and analyze its impact **Activities**: 1. Generate a basic payload using **msfvenom** and test with **VirusTotal** 2. Simulate phishing using **SET Toolkit** (Social Engineering Toolkit) 3. Launch a DoS attack using **Hping3** or **Slowloris** in test setup 4. Monitor and log attack behavior with **Wireshark** **Tools**: Metasploit, VirusTotal, SET Toolkit, Hping3, Wireshark	
4	**Cloud, IoT Security & Penetration Testing Frameworks** **Objectives**: • Analyze cloud and IoT vulnerabilities • Use a penetration testing framework to simulate exploitation • Evaluate a system's security using a standard methodology **Activities**: 1. Scan and analyze open S3 buckets or Azure blobs (safe targets only) 2. Capture IoT device traffic using **Wireshark** 3. Launch an exploit using **Metasploit** against a known vulnerable VM 4. Document attack steps and post-exploitation results **Tools**: Metasploit, Wireshark, AWS CLI, Kali Linux	CO 4
5	**Capstone Project – End-to-End Penetration Test Simulation** **Title**: **“Simulated Attack on an Enterprise Test Network”** **Objectives**: • Execute a full 5-phase ethical hacking workflow: Reconnaissance → Scanning → Exploitation → Privilege Escalation → Reporting • Combine all skills from previous labs • Develop and present a professional pentest report **Deliverables**: • Exploitation walkthrough and tool usage screenshots • Summary of vulnerabilities with OWASP classification • Risk-based remediation recommendations • Structured final report (executive + technical sections) **Tools**: All previously used tools (Kali Linux, Nmap, Burp, Metasploit, etc.)	CO 5

Recommended Textbooks

1. **OWASP Foundation**. OWASP Top 10 and Application Security Verification Standard (ASVS).
2. **Neil Bergman** (2023). *Web Application Security: Exploitation and Countermeasures*. No Starch Press.
3. **Mark Merkow** (2021). *Secure Coding in C and C++*. Pearson.

Securing Cloud Infrastructures and Services

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Securing Cloud Infrastructures and Services	**Course Code**	**L-T- P**	**Credits**
		3-0-2	4
Type of Course:	DSE		
Contact Hours	45 hrs		
Version			
Pre-requisite(s), if any: None			

Course Perspective: To provide students with comprehensive knowledge and practical skills necessary to understand cloud computing fundamentals, secure cloud environments, manage risks, and ensure compliance using modern industry-standard open-source tools and practical techniques.

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

COs	Statements
CO 1	**Understand** fundamental concepts of cloud computing, including architecture, service models, and deployment models.
CO 2	**Implement** and manage effective identity and access management (IAM) using practical industry-standard tools.
CO 3	**Apply** data protection and encryption strategies within cloud environments using open-source tools.
CO 4	**Ensure** compliance and perform security audits using practical cloud-security posture management tools.
CO 5	**Understand** fundamental concepts of cloud computing, including architecture, service models, and deployment models.

Course Outline:

Unit Number: 1	**Title: Fundamentals of Cloud Computing and Practical Deployments**	**No. of hours: 8**

<table>
<tr><td colspan="3"><b>Content:</b>
• Introduction to Cloud Computing
• Cloud Service Models (IaaS, PaaS, SaaS)
• Deployment Models (Public, Private, Hybrid)
• Hands-on with AWS Free Tier, Azure, Google Cloud Platform
• Introduction to Terraform for Infrastructure as Code (IaC)
• Introduction to Kubernetes and Docker for containerization
<b>Industry Applications:</b>
• Automated cloud deployment pipelines in startups
• Containerized application deployments in microservices architecture</td></tr>
<tr><td>Unit Number: 2</td><td>Title: <b>Cloud Architecture, Security Models, and Tools</b></td><td>No. of hours: 8</td></tr>
<tr><td colspan="3"><b>Content:</b>
• Cloud Security Shared Responsibility Model
• Threat Modeling and Risk Assessment
• Open-source Security Tools: Cloud Custodian, Scout Suite, OpenSCAP
• Securing Cloud Infrastructure with practical demonstrations
<b>Industry Applications:</b>
• Securing cloud infrastructure using automated tools in DevOps pipelines
• Risk assessment and mitigation in hybrid cloud deployments
○</td></tr>
<tr><td>Unit Number: 3</td><td>Title: <b>Identity, Access Management, and Authentication Tools</b></td><td>No. of hours: 8</td></tr>
<tr><td colspan="3"><b>Content:</b>
• Practical IAM implementation (AWS IAM, Azure AD)
• Multi-factor Authentication (MFA) setup and Single Sign-On (SSO)
• Open-source IAM tools: Keycloak, OAuth2 Proxy
• Managing privileged access with HashiCorp Vault
<b>Industry Applications:</b>
• Centralized identity management in multi-cloud environments
• Secure remote workforce management using IAM tools</td></tr>
<tr><td>Unit Number: 4</td><td>Title: <b>Data Protection, Compliance, and Security Audits</b></td><td>No. of hours: 8</td></tr>
<tr><td colspan="3"><b>Content:</b>
• Data Encryption using open-source tools: Cryptomator, VeraCrypt
• Secure cloud storage implementations
• Automated compliance management with Cloud Security Posture Management (CSPM) tools: Prowler, Forseti Security
• Conducting cloud security audits with ScoutSuite, CloudSploit
<b>Industry Applications:</b>
• Securely managing healthcare data using open-source encryption tools
• Automated compliance and continuous security audits in FinTech</td></tr>
</table>

Textbooks:

- Walker, Matt. "CEH Certified Ethical Hacker All-in-One Exam Guide," 5th Edition, McGraw-Hill Education.
- Kim, Peter. "The Hacker Playbook 3: Practical Guide To Penetration Testing," CreateSpace Independent Publishing Platform.

LAB Experiments

Ex. No	Experiment Title	Mapped CO/COs
1	**Cloud Infrastructure Deployment & Containerization** **Objective**: To deploy and manage basic cloud infrastructure and containers using cloud consoles and automation tools. **Tools Required**: AWS Free Tier / GCP / Azure, Terraform, Docker, Kubernetes (Minikube or Play with Kubernetes) **Procedure**: 1. Launch a Linux virtual machine on **AWS Free Tier** or **GCP**. 2. Install and deploy a sample application using **Docker**. 3. Write a basic **Terraform script** to provision a VM and storage. 4. Deploy a containerized app using **Kubernetes** (Minikube or online simulators).	CO 1
2	**Securing Cloud Infrastructure Using Open-Source Tools** **Objective**: To assess and secure cloud infrastructure using community-based security tools. **Tools Required**: Cloud Custodian, Scout Suite, OpenSCAP, AWS or GCP account **Procedure**: 1. Scan your AWS/GCP cloud account using **Scout Suite** and analyze vulnerabilities. 2. Create a simple **Cloud Custodian policy** to enforce security (e.g., tag compliance or public S3 blocking). 3. Perform system configuration scanning with **OpenSCAP** on a cloud-hosted Linux VM. 4. Document threats and propose mitigations based on risk score.	CO 2
3	**Identity and Access Management (IAM) Configuration** **Objective**: To implement secure user access and IAM policies using both cloud-native and open-source tools. **Tools Required**: AWS IAM, Azure AD (free tier), Keycloak, OAuth2 Proxy, HashiCorp Vault **Procedure**: 1. Create IAM roles and user policies using **AWS IAM Console** or **Azure AD**.	CO 3

	2. Set up **Multi-Factor Authentication (MFA)** for a user account. 3. Install and configure **Keycloak** to enable Single Sign-On (SSO) for a sample app. 4. Use **Vault** to securely store and access application secrets (tokens, passwords).	
4	**Data Encryption, Storage Protection & Compliance Auditing** **Objective**: To protect cloud-stored data and assess compliance with security benchmarks. **Tools Required**: Cryptomator, VeraCrypt, Prowler, Forseti Security, CloudSploit, ScoutSuite **Procedure**: 1. Encrypt local data using **Cryptomator** or **VeraCrypt** and sync with cloud storage (Google Drive / Dropbox). 2. Run **Prowler** to perform AWS CIS benchmark compliance checks. 3. Install and run **Forseti Security** on GCP to analyze IAM and firewall rules. 4. Execute **CloudSploit** or **ScoutSuite** to conduct a security audit of your cloud environment.	CO 4
5	**Capstone Project – Designing and Securing a Multi-Cloud Environment** **Title**: “Build and Secure a Multi-Service Cloud Infrastructure with Compliance Integration” **Objectives**: • Design a small infrastructure using multiple cloud services (VM + Storage + IAM). • Apply encryption to protect stored data. • Implement IAM roles with MFA and centralized SSO. • Scan for misconfigurations and compliance issues using automated tools. • Submit a structured **Cloud Security Assessment Report**. **Deliverables**: • Terraform scripts / IAM configuration files • Screenshots of tool usage (ScoutSuite, Prowler, Keycloak, etc.) • Final security audit report (risks, tools used, mitigation plan)	CO 5

Comprehensive Placement Preparation

Programme Name:	**Bachelor in Computer Applications (BCA)**			
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		

<table>
<tr><td colspan="3">• <b>Session 10</b>: Professional Etiquette + Body Language Workshop.</td></tr>
<tr><td><b>Unit Number: 4</b></td><td><b>Placement Simulation, Engagement & Evaluation</b></td><td><b>No. of hours: 8</b></td></tr>
<tr><td colspan="3"><b>Content:</b>
• <b>Session 7:</b> Company-Specific Test-1 + Discussion
• <b>Session 13:</b> Group Discussion Workshop + Mock GD Rounds
• <b>Session 14:</b> Mock Interview + Video Resume Workshop
• <b>Session 15:</b> Mock PI Round + Student Video Resume Showcase</td></tr>
</table>

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:	**Bachelor in Computer Applications (BCA)**			
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		

Semester-7

Applied Generative AI

Program Name	**Bachelor in Computer Applications (BCA)**			
Course Name: Applied Generative AI	**Course Code**	**L-T-P**	**Credits**	**Contact Hours**
	ETCCGA703	3-0-2	4	45
Type of Course:	Major			
Pre-requisite(s): A foundational understanding of AI Tools and Techniques				

Course Perspective: This course introduces BCA students to the world of Generative AI and Large Language Models (LLMs) through practical activities and projects. Students will learn how AI tools like ChatGPT work, how to write prompts to control these models, and how to build simple AI apps using free tools like Hugging Face, Gradio, and Google Colab.

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

COs	Statements
CO 1	Understand how Generative AI models work.
CO 2	Write good prompts to generate text, code, and summaries.
CO 3	Use free AI tools and models to solve simple problems.
CO 4	Create basic AI applications like chatbots and text generators.

Course Outline:

Unit Number: 1	**Title: Basics of Generative AI & LLMs**	**No. of hours: 10**
• What is Generative AI? How does it work? • Examples of popular models: ChatGPT, Google Gemma, Meta Llama • Introduction to Hugging Face and Google Colab • Hands-on: Run your first text generator using Hugging Face • Tools Used: Hugging Face, Google Colab Use Case: How ChatGPT answers user questions in customer service.		
Unit Number:2	**Title: Prompting and Getting Better Outputs**	**No. of hours: 10**

<table>
<tr><td colspan="3">What is a "prompt"? How to write a good one?
• Types: Simple, Few Examples, Step-by-Step Instructions
• Prompt tuning vs. model fine-tuning
• Hands-on: Write prompts for creative writing and coding
<b>Tools Used:</b> OpenAI Playground, LangChain
<b>Use Case:</b> How content tools like Copy.ai create social media captions.</td></tr>
<tr><td><b>Unit Number: 3</b></td><td><b>Title: Building Mini Apps with Open AI Models</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3">• Create chatbots using local LLMs
• Use LLMs to generate Python or SQL code
• Automate summaries and text writing
• Hands-on: Use LLaMA 3 or Mistral for small apps
<b>Tools Used:</b> Hugging Face, LangChain, Google Colab
<b>Use Case:</b> How GitHub Copilot helps programmers write code faster.</td></tr>
<tr><td><b>Unit Number: 4</b></td><td><b>Title: Deploying AI Tools Online</b></td><td><b>No. of hours: 10</b></td></tr>
<tr><td colspan="3">• Use LangChain to connect prompts and responses
• Make web apps using Gradio and Streamlit
• Publish your AI tool on Hugging Face Spaces
• Hands-on: Build and launch your own chatbot or summarizer
<b>Tools Used:</b> Streamlit, Gradio, Hugging Face Spaces
<b>Use Case:</b> How AI tools are launched and used by people online.</td></tr>
</table>

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: Run a Text Generator

Real-World Scenario: You've joined a company's customer support automation team. Your first assignment is to test different open-source AI models to decide which gives the best responses to basic user queries like greetings, FAQs, and feedback responses.

Sub-Objectives:

- Set up Hugging Face Transformers in Google Colab.
- Load a model like Gemma or Falcon for basic text generation.
- Process and tokenize input/output strings.
- Compare responses to identical prompts across models.

Tools: Google Colab, Hugging Face

Lab Task 2: Write Prompts for Different Tasks

Real-World Scenario: You work at a startup that automates blog writing and email generation. Your job is to write different types of prompts that get high-quality, relevant outputs from models like ChatGPT or Llama.

Sub-Objectives:

- Write prompts for creative writing, summaries, and coding.
- Practice zero-shot, few-shot, and step-by-step (chain-of-thought) prompts.
- Generate Python and SQL code based on simple instructions.
- Evaluate prompt output quality in OpenAI Playground and Hugging Face Inference.

Tools: OpenAI Playground, LangChain, Hugging Face

Lab Task 3: Build a Chatbot or Code Helper

Real-World Scenario: You've been assigned to develop a virtual assistant that answers college FAQs and can also generate code for basic programming problems to help students.

Sub-Objectives:

- Run open LLMs like LLaMA 3 or Mistral using Hugging Face.
- Build a Q&A chatbot using LangChain or Hugging Face Pipeline.
- Generate code snippets (e.g., sorting, database queries) from user input.
- Add input validation or error messaging for unclear prompts.

Tools: Hugging Face, LangChain, Google Colab

Lab Task 4: Launch Your AI Tool

Real-World Scenario: You're part of a student hackathon team. Your task is to develop and publish an AI-based summarizer or writing assistant that can be accessed online by anyone with a browser.

Sub-Objectives:

- Chain prompts using LangChain for workflow automation.
- Create a user interface using Gradio or Streamlit.
- Deploy the application using Hugging Face Spaces.
- Test the app and document prompt logic and deployment steps.

Tools: Gradio, Streamlit, Hugging Face Spaces

Capstone Project: Create Your Own AI Assistant

Real-World Scenario: You're helping a startup build an AI-based productivity tool that can summarize documents, answer job interview questions, or assist in startup pitch writing.

Sub-Objectives:

- Propose and design an idea for a usable AI tool.
- Choose an LLM (open-source or API-based) and write prompts.
- Build the interface with Gradio or Streamlit.
- Demonstrate the tool, explain use cases, and show its limitations.

Tools: Hugging Face, Google Colab, LangChain, Gradio, Streamlit

Incident Response and Cyber Investigation

Program Name	**Bachelor in Computer Applications (BCA)**		
Course Name: Incident Response and Cyber Investigation	**Course Code**	**L-T- P**	**Credits**
		3-0-2	4
Type of Course:	DSE		
Contact Hours	45 hrs		
Version			
Pre-requisite(s), if any: None			

Course Perspective: This course provides students with the theoretical knowledge and hands-on skills required to handle cybersecurity incidents and conduct digital investigations. Students will learn how to respond to cyberattacks, preserve evidence, analyze logs, and apply forensic techniques to detect intrusions and reconstruct attacks. Emphasis is placed on real-world cyber incident response frameworks, forensic imaging, log correlation, and legal compliance.

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

COs	Statements
CO 1	Understand the phases of incident response and associated frameworks.
CO 2	Apply log analysis and system monitoring techniques for threat detection.
CO 3	Perform digital forensic acquisition and analysis using open-source tools.
CO 4	Document and report security incidents following legal and procedural standards.
CO 5	Conduct end-to-end incident investigation using structured methodologies.

Course Outline:

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

Content:

Topics:

- What is an incident? Definitions and classification
- Common types of incidents (Malware, Phishing, Insider Threat, DoS, Ransomware)
- Incident Response Lifecycle (Preparation, Identification, Containment, Eradication, Recovery, Lessons Learned) – based on **NIST SP 800-61**
- Roles and responsibilities in an Incident Response Team (IRT)
- Real-world breach analysis: Equifax, SolarWinds

Hands-On / Real Case:

- Simulate a phishing attack and outline each IR phase
- Analyze a timeline of a major real-world cyber breach

Unit Number: 2	**Title: Evidence Collection, Chain of Custody & Forensics**	**No. of hours: 8**

Content:

- Forensic fundamentals: Volatile and non-volatile data
- Chain of custody and evidence integrity
- Disk and memory imaging (live vs dead acquisition)
- File systems and metadata analysis (FAT, NTFS, ext4)
- Write blockers and hashing (MD5, SHA-256)

Hands-On / Real Case:

- Use **FTK Imager** or **Autopsy** to acquire a disk image
- Demonstrate hash comparison to validate evidence
  - ○

Unit Number: 3	**Title: Log Analysis, Malware Behavior & Threat Detection**	**No. of hours: 8**

Content:

- Log sources: Windows Event Logs, Syslogs, Web logs, Firewall logs
- Log correlation and time-based event tracking
- Malware behavior indicators (registry, startup entries, connections)
- Introduction to SIEM (Security Information & Event Management)
- Threat intelligence feeds and IOC (Indicators of Compromise)

Hands-On / Real Case:

- Analyze system logs from a simulated malware infection
- Use **ELK Stack** or **Splunk Free** for basic log correlation

Unit Number: 4	**Title: Digital Forensics Techniques & Reporting**	**No. of hours: 8**

Content:

- File carving and data recovery
- Timeline analysis and event reconstruction
- Email and browser forensics
- Mobile forensics basics (Android tools overview)
- Report writing: Technical and Legal perspectives
- Legal considerations: IT Act 2000, GDPR, Chain of Custody

Hands-On / Real Case:

- Recover deleted files and analyze user activity timeline
- Draft a formal incident response and forensics report

Lab Experiments

Lab Task 1 – Simulating and Documenting a Cyber Incident

Objectives:

- Understand the phases of incident response (NIST-based).
- Identify and categorize types of cyber incidents.
- Simulate and document response steps for a phishing incident.

Activities:

1. Simulate a phishing email scenario using a fake form or link (offline setup).
2. Identify indicators of compromise (IoCs) such as suspicious email headers.
3. Document incident handling through the 6-phase IR lifecycle.
4. Create an incident response checklist and escalation flow.

Tools: Web browser, Notepad, Email client, Flowchart maker (e.g., Draw.io)

Lab Task 2 – Evidence Acquisition and Chain of Custody

Objectives:

- Perform disk image acquisition for forensic analysis.
- Ensure integrity using hashing techniques.
- Understand and document the chain of custody for admissible evidence.

Activities:

1. Acquire a forensic image of a USB drive or virtual disk using **FTK Imager**.
2. Calculate **MD5 and SHA-256 hashes** before and after imaging.
3. Fill out a sample chain of custody form with timestamps and handlers.
4. Explore file system metadata and recover file timestamps.

Tools: FTK Imager, HashCalc, USB image/sample VM, Chain of Custody template

Lab Task 3 – Log Analysis and Threat Detection

Objectives:

- Correlate logs from various sources to identify suspicious activity.
- Detect malware behavior using event logs and threat indicators.
- Use SIEM tools to visualize and alert on suspicious patterns.

Activities:

1. Collect and review logs from a simulated system attack (Windows/Linux).
2. Use **Splunk Free** or **ELK Stack** to ingest logs and query anomalies.
3. Identify unauthorized access, malware signatures, or data exfiltration trails.
4. Correlate timestamped events into an incident timeline.

Tools: ELK Stack / Splunk Free, Windows Event Viewer / syslog, Sample log files

Lab Task 4 – Digital Forensics & Incident Reporting

Objectives:

- Analyze disk images for deleted or suspicious data.
- Reconstruct user activity and generate forensic artifacts.
- Create a formal incident response report suitable for legal or managerial review.

Activities:

1. Load a disk image into **Autopsy** and analyze file activity and browser history.
2. Recover deleted files and extract metadata using **Bulk Extractor**.
3. Create a timeline of attacker behavior and affected resources.
4. Draft a professional forensic report including executive summary and evidence list.

Tools: Autopsy, Bulk Extractor, Timeline Generator, Word Processor

Capstone Project – Full Cybersecurity Investigation Simulation

Title: "Investigating a Simulated Enterprise Breach"

Objectives:

- Apply all stages of incident response and forensic investigation.
- Use multiple tools to collect, correlate, and analyze data.
- Compile a comprehensive report including findings, evidence, and mitigation steps.

Activities:

1. Investigate a compromised VM or prepared lab environment (phishing, malware, or unauthorized login).
2. Capture volatile and non-volatile data (memory, disk, logs).
3. Perform forensics analysis and identify root cause.
4. Present findings in a structured report with screenshots, timeline, and recommendations.

Deliverables:

- Forensic images or log exports
- Analysis report (technical + managerial sections)
- Summary presentation (optional for viva)

Tools: All previously used tools (FTK Imager, Autopsy, Splunk,

Major Project- I

Program Name:	**Bachelor in Computer Applications (BCA)**		
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	Activity/Stage
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:	**Bachelor in Computer Applications (BCA)**		
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 8

Summer Internship-IV(Assessments)

Program Name:	**Bachelor in Computer Applications (BCA)**		
Course Name:	**Course Code**	**L-T-P**	**Credits**
Summer Internship- IV (Assesment)	**ETCCIN702**	0-0-16	8
Type of Course:	INT		

Preface:

The **BCA 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 BCA Final Semester Full-Time Project Work

1. Introduction

The **BCA 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.

Major Project- II

Program Name:	Bachelor in Computer Applications (BCA)		
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	Activity/Stage
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.