SNDT Women's University, Mumbai

Postgraduate Degree / PG Programme (Syllabus as Per NEP) - Faculty of Science & Technology

Master of Computer Applications-Integrated

(MCA-Integrated)

As Per NEP – 2020

Semester – I to IV

Syllabus
(W.E.F. Academic Year 2025-26)

Programme Template

Programme Degree		Master of Computer Applications-Integrated (MCA-Integrated)
Parenthesis if any (Specialization) e.g. History, Human Development, English, etc.		NA
Preamble (Brief Introduction to the programme)		The **MCA-Integrated course** is a comprehensive four-year academic program designed to equip students with in-depth knowledge and practical skills in computer applications, software development, and emerging technologies. Offered under the guidelines of the **National Education Policy (NEP) 2020**, the program integrates undergraduate and postgraduate studies, allowing students to obtain both a Bachelor's and Master's degree in Computer Applications without the need for separate admissions. The curriculum blends core computing concepts such as programming in C and Java, data structures, operating systems, and database management with advanced areas like cloud computing, and cybersecurity. It also includes interdisciplinary open electives, co-curricular activities, and industry-oriented projects that foster holistic and experiential learning. Emphasizing hands-on learning, each semester incorporates practical sessions, continuous assessments, and real-world project development. Students gain proficiency in problem-solving, algorithmic thinking, and software engineering practices. The program nurtures critical thinking, ethical responsibility, and digital sustainability. With exposure to tools like Spring Boot, React, and modern database technologies, graduates are well-prepared for careers in software development, IT consultancy, data analytics, and research. The MCA-Integrated course provides a seamless academic journey that bridges foundational knowledge with advanced specialization, aligning with global technological trends and industry needs.
Programme Specific Outcomes (POs)		**After completing this programme, Learner will**
	1.	Apply foundational and advanced programming skills using C, Java, and web technologies to develop robust software applications.
	2.	Design, implement, and manage software solutions using structured methodologies, including SDLC, OOP, MVC, and full-stack development.
	3.	Develop interactive and secure websites and web applications using HTML, CSS, JavaScript, backend frameworks, and cloud deployment tools.
	4.	Understand and apply concepts of operating systems, computer organization, microprocessors,

		and microcontrollers for system-level programming and embedded solutions.
	5.	Design and manage relational and NoSQL databases, perform transactions, and implement database connectivity in enterprise applications.
	6.	Utilize linear and non-linear data structures and algorithms for problem-solving, performance optimization, and competitive programming.
	7.	Solve computational and real-world problems using discrete mathematics, numerical methods, probability, and linear programming.
	8.	Plan and execute real-time projects using modern tools and frameworks with proper documentation, testing, and deployment strategies aligned to industry standards.
	9.	Demonstrate awareness of ethical, legal, and environmental aspects in computing practices, including digital literacy, e-waste management, and cybersecurity.
	10.	Integrate knowledge from interdisciplinary open electives, co-curricular activities, and ability enhancement courses for personal and professional development.
Eligibility Criteria for Programme		A female student who has passed 10+2 examination with Mathematics/Statistics/ Accountancy as compulsory subject. Obtained at least 45% marks in general category and 40% marks in case of candidates belonging to reserved category, in the above subjects taken together and Valid MAH-MCA Integrated-CET score applicable to that academic year.
Intake (For SNDT WU Departments and Conducted Colleges)		60
Examination passing criteria		Students have to pass both the internal assessment and external assessment separately for each course with minimum 40% passing marks

Structure with Course Titles

SN	Courses	Type of Course	Credits	Marks	Int	Ext
	Semester I					
10135111	Problem solving using C	Major (Core)	4	100	50	50
10135102	Fundamental of Computer science	Major (Core)	2	50	50	0
10435111	1. Fundamentals of Computer Science and Programming	OEC **(Any One)**	4	100	50	50
10435112	2. Foundations of Data Science					
10435113	3. Introduction to Computer Hardware and Networking					
10635101	Web Technologies-I	VSC	2	50	50	0
10735111	How Computer Works-From Basic to Code Execution	SEC	2	50	0	50
10810111	English For Academic Writing- Paper I (For Students of English Medium)	AEC (English) (Any One)	2	50	0	50
10810112	English Language and Literature- I (For Students of NonEnglish medium)					
	AEC Link: https://sndt.ac.in/pdf/academics/syllabus-as-per-nep/aec-syllabus/ug-degree/ability-enhancement-course.pdf (Available on Website)					
11051111	Indian Knowledge System IKS Link: https://sndt.ac.in/pdf/academics/syllabus-as-per-nep/iks-syllabus/ug-degree/inception-of-indian-knowledge-system.pdf (Available on Website)	IKS	2	50	0	50

10952111	Introduction to Indian Constitution Link: https://sndt.ac.in/pdf/academics/syllabus-as-per-nep/vec-syllabus/ug-degree/introduction-to-indian-constitution.pdf (Available on Website)	VEC	2	50	0	50
11450121	Basics of National Service Scheme	CC (Any One)	2	50	50	0
11450221	National Cadets Corps. (NCC) Studies- I					
11450322	Health and Wellness					
11450421	Performing Arts Exploration					
	CC Link: https://sndt.ac.in/pdf/academics/syllabus-as-per-nep/cc-syllabus/ug-degree/co-curricular-course-as-per-nep-2020-semester-i-syllabus.pdf (Available on Website)					
			22	**550**	**250**	**300**
	*** 1.6, 1.7,1.8,1.9 is as per University Basket**					

	Semester II					
20135111	Object Oriented Programming using Java	Major (Core)	4	100	50	50
20135112	Operating system	Major (Core)	2	50	0	50
20335101	Mathematical Foundation of Computer Science – I	Minor Stream	2	50	50	0
20435111	1. Introduction to Computer Organization and Architecture	OEC (Any One)	4	100	50	50
20435112	2. Computer Applications & Office Automation					
20435113	3. Basics of Web Development (HTML, CSS, JavaScript)					
20635101	Computer Organization and Architecture	VSC	2	50	50	0
20735101	Web Technologies-II	SEC	2	50	50	0
20810111	English For Academic WritingPaper II (For Students of English Medium)	AEC (Any One)	2	50	0	50
20810112	English Language and Literature- II (For Students of Non-English medium)					
	AEC Link: https://sndt.ac.in/pdf/academics/syllabus-as-per-nep/aec-syllabus/ug-degree/ability-enhancement-course.pdf (Available on Website)					
20952111	Environment Awareness Link: https://sndt.ac.in/pdf/academics/syllabus-as-per-nep/vec-syllabus/ug-degree/environment-awareness.pdf (Available on Website)	VEC	2	50	0	50
21450121	Volunteerism and National Service Scheme	CC (Any One)	2	50	50	0
21450221	National Cadets Corps. (NCC) Studies- II					

21450323	Yoga Education					
21450421	Fine Art					
	CC Link: https://sndt.ac.in/pdf/academics/syllabus-as-per-nep/cc-syllabus/ug-degree/co-curricular-course-as-per-nep-2020-semester-ii-syllabus.pdf (Available on Website)					
			22	**550**	**300**	**250**
	*** 2.7,2.8,2.9 is as per University Basket**					

SN	Courses	Type of Course	Credits	Marks	Int	Ext
	Semester III					
30135111	Advance Java Programming	Major (Core)	4	100	50	50
30135112	Database Management System	Major (Core)	4	100	50	50
30335111	Mathematical Foundation of Computer Science - II	Minor Stream	4	100	50	50
30435111	1. Introduction to Web and Internet Technologies	OEC (Any One)	2	50	0	50
30435112	2. Fundamentals of Artificial Intelligence and Automation					
30435113	3. Introduction to Cyber security and Digital Ethics					
30635111	Introduction to Microprocessor and Microcontroller	VSC	2	50	0	50
30810301	रचनात्मक लेखन (Hindi)	AEC (Any One)	2	50	50	0
30810401	मराठी भाषेचा परिचय - भाग १ (MARATHI)					
30810501	Contemporary Sanskrit Nyaya (Sanskrit)					
30810201	શીખો ગુજરાતી – પ્રાથમિક ભાગ ૧: લિપિ પરિચય, શ્રવણ અને વાચન કૌશલ્ય (Gujarati)					
	Modern Indian Language Ability Enhancement Course (AEC) Link: https://sndt.ac.in/pdf/academics/syllabus-as-per-nep/aec-syllabus/ug-degree/aec-semester-iii.pdf (Available on Website)					
31335101	Field Project	FP	2	50	50	0
31450121	Social issues Advocacy and Action	CC	2	50	50	0
31450221	National Cadets Corps. (NCC) Studies – III					
31450321	Traditional Sports and Fitness					

31450421	Unfolding The Beauty of Indian Music					
	Co-Curricular Course (CC) Link: https://sndt.ac.in/pdf/academics/syllabus-as-per-nep/cc-syllabus/ug-degree/co-curricular-course-as-per-nep-2020-semester-iii-syllabus.pdf (Available on Website)					
			22	**550**	**300**	**250**
	*** 3.6 is as per University Basket** *** 3.7 Field Project will be internal projects assigned to individual student on major subjects.**					

	Semester IV					
40135111	Data Structure and Algorithm	Major (Core)	4	100	50	50
40135112	Data Communication and Networking	Major (Core)	4	100	50	50
40335111	Probability and Statistics	Minor Stream	4	100	50	50
40435111	1. C Programming for Beginners	OEC (Any One)	2	50	0	50
40435112	2. Digital Marketing					
40435113	3. Internet of Things (IoT)					
	Swayam/Chetana/MOOC	SEC	2	50	0	50
40810411	मराठी भाषेचा परिचय - भाग २ (MARATHI)	AEC (Any One)	2	50	0	50
40810411	सूचना प्रौद्योगगकी और ग ंदी भाषा (Hindi)					
40810511	वाल्मिकीगकरामयणे अयोध्याकाण्ड: (Sanskrit)					
40810211	શીખો ગુજરાતી – ભાધ્ંમિક (Gujarati)					
	Modern Indian Language Ability Enhancement Course (AEC) Link: https://sndt.ac.in/pdf/academics/syllabus-as-per-nep/aec-syllabus/ug-degree/aec-semester-iv.pdf (Available on Website)					
41735101	Digital literacy and E- Waste Management	CEP	2	50	50	0
41450122	Personality and Leadership Development through National Service Scheme	CC (Any One)	2	50	50	0
41450121	NSS Volunteers under National service scheme special camp					
41450221	National Cadets Corps. (NCC) Studies – IV					
41450421	Theatre & Dance					

	Co-Curricular Course (CC) Link: https://sndt.ac.in/pdf/academics/syllabus-as-per-nep/cc-syllabus/ug-degree/co-curricular-course-as-per-nep-2020-semester-iv-syllabus.pdf (Available on Website)					
			22	**550**	**250**	**300**
	*** 4.6 is as per University Basket**					

Course Syllabus

Semester: I

Course Code	**1.1 (Major Core)**
Course Title	**PROBLEM SOLVING USING C**
Course Credits	**4**
Course Outcomes	**After Completion of this Course, students will be able to** • Apply logic to create programs in C. • Analyze and understand computer programming language concepts. • Evaluate and interpret the use of pointers, including their declarations, initialization, and operations. • Design and develop applications using basic programming constructs, facilitating the transition to other languages.
Module 1: Introduction to problem solving and Programming Languages (Credit1)	
Learning Outcomes	**After learning this module, learners will be able to** • Learn and apply steps in problem solving. • Understand the overview of programming languages.
Content Outline	• **Introduction to problem solving**: Concept: Steps in problem solving - (Define Problem, Analyze Problem, Explore Solution), Problem solving techniques - (Trial& Error, Brain Storming, Divide & Conquer), Algorithms and Flowcharts (Definitions, Characteristics, Advantage& Disadvantages, Symbols, Examples), Pseudo-code (Definition, Conditional statements, Loops), etc. • **Overview of programming languages**: Definition of the program, Concept- Source code, Object code, Compilation, Interpretation, Execution, Input and Output, Debugging etc., Expressions, control structures; subroutines, Storage management; scoping rules; bindings for names, Storage types: Automatic, external, register and static variables
Module 2: Introduction to C Programming (Credit1)	
Learning Outcomes	**After learning this module, learners will be able to** • Evaluate the concept of I/O functions, header files, and preprocessor directives. • Understand structure, keywords, operators of C programming • Design and apply concepts of the C language.

Content Outline	• **Introduction to 'C' Language:** • History of C Programming, Structures of C Programming, Simple example, Basic Input / Output, Function as building blocks. Language Fundamentals: Character set, C Tokens, Keywords, Identifiers, Variables, Constant, Data Types, Comments. • **Operators**: Types of operators, Precedence and Associativity, Expression. Statement and types of statements, Built-in Operators and function. Console based I/O and related builtin I/O Function: printf(), scanf(), getch(), getchar(), putchar(),etc; Concep to fheader files, Preprocess or directives: #include, #define, Conditional statements and Loops
Module 3: Control Statement, Functions and Pointers (Credit1)	
Learning Outcomes	**After learning the module, learners will be able to** • Gain proficiency in writing C programs to solve various problems. Learn the syntax and semantics of the C language, including its specific features such as pointers and memory management.
Content Outline	• **Control structures** ➢ Decision making structures: If, If-else, Nested If –else, Switch, Loop Control structures ➢ While, Do-while, For, Nested for, while, do-while loop, jumping statements: break, continue, goto, exit. • **Functions:** Definition, Basic types of function, Declaration and definition, Function call, Types of function, Parameter passing, Call by value, Call by reference, Scope of variables, Recursion, String: Declaration, string Functions, String Manipulations • **Pointers:** Introduction to pointers, Pointer notation, Pointer arithmetic, Null Pointer
Module 4: Array, Structure, Union and File handling (Credit1)	
Learning Outcomes	**After learning the module, learners will be able to**
	• Gain proficiency in writing C programs to solve various problems. • Develop programs using Arrays and Pointers. • Analyze the difference between structure and union. • Design and handle operations of the files.
Content Outline	• **Arrays:** Definition, Declaration, Initialization, Bounds checking, One-Dimensional Array, Two-DimensionalArray, Passing Array to a function, pointer to Array • **Structure and Union:** Introduction to Structure, Definition, Declaration of Structure Variables, Dot Operator, Nested Structure, Array

	Of Structure, pointer to structure, Introduction to Union, Difference between Structureand Union • **File Handling:** Concept of File, Definition, File operations (create, open, read, move, write, close), File opening Mode, Closing a file, Input/output operations, Creating and reading a file, Command Line Argument

Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1:

- Create a flowchart and algorithm for basic problems (e.g., calculating the factorial of a number or finding the sum of first N natural numbers).
- Write pseudo-code for the above problem.
- Convert the pseudo-code into a C program and demonstrate input/output operations.
- Illustrate the steps in problem-solving techniques (Trial & Error, Brainstorming, Divide & Conquer) with real-life examples.
- Explain and compare compilation vs. interpretation using an example program.

Module 2:

- Write a C program to demonstrate basic I/O functions (e.g., printf(), scanf(), getchar(), putchar()).
- Develop a program to demonstrate the use of different operators (arithmetic, relational, logical, bitwise, assignment).
- Implement examples to illustrate operator precedence and associativity in expressions.
- Demonstrate the use of header files and preprocessor directives (#include, #define).
- Write a program using conditional statements (if, if-else, switch-case) to evaluate mathematical expressions.

Module 3:

- Write C programs using different looping structures (for, while, do-while).
- Implement nested loops and jumping statements (break, continue, goto).
- Develop a program using functions with parameter passing (call by value and call by reference).
- Write a program that demonstrates recursion, such as calculating the Fibonacci sequence or factorial.
- Implement a C program that uses string functions (strlen(), strcpy(), strcmp(), strcat()) for string manipulation.
- Demonstrate pointer arithmetic operations in a C program.
- Write a program to illustrate the concept of null pointers and pointer initialization.

Module 4:

- Write a program to declare, initialize, and manipulate one-dimensional and two-dimensional arrays.
- Implement a C program that passes an array to a function and performs sorting or searching operations.
- Develop a program to illustrate structures and unions, highlighting their differences.
- Implement pointers to structures and demonstrate accessing structure members using pointers.
- Write a program to perform basic file operations (create, open, read, write, close) in C.

- Implement a program that reads data from a file, processes it, and writes the output to another file.
- Develop a C program that uses command-line arguments for file handling operations.

Evaluation Scheme:

Internal Assessment:

Depending on the activities mentioned above a project should be developed of 50 marks. Internal Assessment to be done on the basis of Project Presentations and Report Writing with proper Rubrics (Given below).

The Rubric will have the following Evaluation Parameters:	
Evaluation Parameters	**Marks**
Functionality (Core Features)	20
Code Quality (Structure, Readability)	10
GUI Design & User Experience	10
Input Handling & Error Management	5
Report writing or Documentation	5

The project Internal assessment will be done by conducting project presentation at the department level, where External Examiner appointed by the department will be evaluating the project.

Two sample projects are given below and it is expected from students to choose a good and advance project topic that aligns with their interests and abilities to work on.

Project 1: Student Record Management System
To develop a C program that stores, retrieves, modifies, and deletes student records using file handling, structures, arrays, and functions.

Project 2: Mini Banking System
Create a menu-driven C program to simulate a simple banking system using functions, arrays, structures, and file operations.

External Assessment:

End Semester examination of 50 marks for 2 hours duration will be conducted by the
University.

References:-

1. Schildt, H. (2000). C: The Complete Reference (4th ed.). Tata McGraw-Hill Education Pvt. Ltd.
2. Ramkumar, & Agrawal. (1996). Programming in ANSI C. Tata McGraw-Hill.
3. Kanetkar, Y. P. (2008). Let Us C. Infinity Science Press.

Course Code	**1.2 (Major Core)**
Course Title	**FUNDAMENTALS OF COMPUTER SCIENCE**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to** • Apply computer fundamentals comprehensively. • Analyze and solve problems using programming and software development skills. • Evaluate proficiency in programming languages, software development life cycles, and debugging techniques. • Design software with strong internet literacy and effective problem-solving skills.
Module 1: Introduction to Computer and Computer Architecture (Credit1)	
Learning Outcomes	**After learning the module, learners will be able to** • Apply knowledge of essential computer components, including CPU, input/output devices, memory, and storage. • Analyze internal data representation, including number systems, binary arithmetic, Boolean algebra, and logic gates. • Evaluate the functions of various input/output devices and interfaces within computer systems. • Design an understanding of basic computer organization and the roles of CPU, memory, and secondary storage.
Content Outline	• **Knowing computer:** What is Computer, Basic Applications of Computer; Components of Computer System, Central Processing Unit (CPU), Input/output Devices, Computer Memory, Concepts of Hardware and Software; Concept of Computing, Data and Information, classification of computers, various generations of computers; What is an Operating System; Different Popular Operating Systems; The User Interface, System Software: System software Vs. Application Software, Types of System Software, Introduction and Types of Operating Systems • **Internal data Representation in Computers and Digital System Design:** Number Systems Used in Computers, Converting from One Number System to Another, Binary Arithmetic, Boolean Algebra, Boolean Functions, Logic Gates, Logic Circuits • **Computer Architecture:** Basic Functions of a Computer, Basic Computer Organization, CPU Architectures, Memory Architectures, Secondary Storage, Classification of Secondary Storage, Memory Storage Devices, Input- Output Devices ,Input Devices, Output Devices, I/O Interfaces.
Module 2: Introduction of Computer languages, software and Internet (Credit1)	
Learning Outcomes	**After learning the module, learners will be able to** • Apply knowledge of computer language evolution and types. • Analyze programming constructs, algorithms, and flowcharting skills. • Evaluate the roles of language processors, software development, and the SDLC. • Design an understanding of basic computer networks, internet connectivity, and the architecture of the World

	Wide Web.
Content Outline	• **Computer Languages:** Definition, Generations of computer languages, Types of Languages, Language Processors: Assembler, Interpreter, Compiler, Linker and Loader. Programming constructs, Algorithm & flowchart. • **Software:** Basic Concepts and Terminologies, What is Software? Relationship between Hardware and Software, Software Development Life Cycle (SDLC), Advantages of SDLC Model, Limitations of SDLC Model, Software Testing and Debugging • **Introduction to Internet:** WWW and Web Browsers: Basic of Computer networks; LAN, WAN; Concept of Internet; Applications of Internet; connecting to internet; What is ISP; Knowing the Internet; Basics of internet connectivity related troubleshooting, World Wide Web; Web Browsing software, Search Engines; Understanding URL; Domain name; IP Address; Using e-governance website

Evaluation Scheme:

External Assessment:

End Semester examination of 50 marks for 2 hours duration will be conducted by the university.

Referen

ces: -

1. Sinha, P. K., & Sinha, P. (2004). Computer Fundamentals. BPB Publications.
2. Srivastava, C. (2010). Fundamentals of Information Technology. Kalyani Publishers.
3. Rajaraman, V. (2003). Fundamentals of Computers (4th ed.). PHI Publication.
4. Jain, R. K. (2010). Fundamentals of Programming. S. Chand Publication.

1.3Open Elective Course-I (OEC-I) Students will choose course offered by other departments.

Course Code	**1.4 (VSC)**
Course Title	**Web Technologies-I**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to** • Apply fundamental concepts of web technology, including the World Wide Web, HTTP protocol, web browsers, and web servers. • Analyze HTML and CSS proficiency to create visually appealing web pages. • Evaluate the use of JavaScript to enhance interactivity and add dynamic functionality to web pages. • Create Responsive website.
Module 1: Introduction to HTML and CSS (1 Credit)	
Learning Outcomes	**After learning this module, learners will be able to** • Understand the basics of web technologies and how websites work • Develop structured web pages using HTML. • Apply CSS for styling and layout of web pages. • Implement responsive design for different screen sizes.
Content Outline	**Introduction to Web Technologies** • Basics of the Internet and Web Technologies • Web Browsers and Web Servers • Client-Server Model **HTML Fundamentals** • Structure of an HTML Document • Basic Tags: Headings, Paragraphs, Lists, Tables • Forms & Input Fields (text, radio, checkbox, button) • Links, Images, and Multimedia • Semantic Elements in HTML5 **CSS Basics** • Introduction to CSS and its Role in Web Design • Selectors, Properties, and Values • Box Model: Margins, Padding, and Borders • CSS Positioning and Display Properties • Layout Techniques: Flexbox and Grid **Styling and Responsive Design** • Media Queries for Mobile-Friendly Design • CSS Transitions & Animations • Best Practices for Modern UI Design
Module 2: Introduction to JavaScript and JavaScript Programming Fundamentals (Credit 1)	
Learning Outcomes	**After learning this module, learners will be able to** • Write basic JavaScript programs for web interactivity. • Use variables, loops, and functions for logic-based execution. • Manipulate DOM elements dynamically using JavaScript. • Handle user events (click, input, hover) and form validation.
Content Outline	**Introduction to JavaScript** • What is JavaScript? • JavaScript in Web Development (Inline, Internal, External) • Basic Syntax, Variables, and Data Types • Operators and Expressions **JavaScript Programming Fundamentals**

	• Conditional Statements (if-else, switch) • Loops (for, while, do-while) • Functions & Scope (function, return, arrow functions) • Arrays & Objects **DOM Manipulation & Events** • Introduction to **Document Object Model (DOM)** • Selecting and Modifying HTML Elements (e.g. getElementById, querySelector) • Event Handling: onclick, onmouseover, onchange, etc. **Basic JavaScript Interactivity** • Form Validation using JavaScript • Simple Animations (setTimeout, setInterval, CSS animations with JS) Introduction to **ES6 Features** (let, const, Template Literals)

Activities towards Comprehensive Continuous Evaluation (CCE)

Module 1:

- Practical 1: Creating a Basic Web Page
  Create a simple webpage using HTML that includes headings, paragraphs, images, and hyperlinks.
  Use lists (ordered and unordered) to structure content properly.
  Add a table displaying student details.
- Practical 2: Forms and Input Fields
  Develop a registration form with various input fields: text, radio buttons, checkboxes, dropdowns, and a submit button.
  Ensure all form elements are appropriately labeled.
- Practical 3: Styling with CSS
  Apply CSS styles to a webpage, including colors, fonts, margins, and borders.
  Use box model properties (padding, margin, border) to structure elements.
- Practical 4: Page Layout with Flexbox and Grid
  Create a webpage layout using CSS Flexbox to align elements properly.
  Implement CSS Grid for a multi-column structure (e.g., a blog page layout).
- Practical 5: Responsive Web Design
  Use media queries to make a webpage responsive for different screen sizes (mobile, tablet, desktop).
  Test responsiveness by resizing the browser window.
- Practical 6: CSS Transitions and Animations
  Implement a hover effect for buttons and images using CSS.
  Create a simple animation (e.g., a fading effect for text or a moving image).
- Project for Module 1:
  Develop a multi-page static website using HTML and CSS with a navigation menu and responsive layout.

Module 2:

- Practical 7: JavaScript Basics
  Write a simple JavaScript program that displays an alert box with a welcome message.
  Demonstrate the use of variables, operators, and expressions in JavaScript.
- Practical 8: Conditional Statements and Loops
  Write a JavaScript program that checks if a number is even or odd using an if-else statement.
  Create a loop that prints numbers from 1 to 10 using a for loop.
- Practical 9: JavaScript Functions and Scope
  Implement a function that calculates and returns the factorial of a number.
  Demonstrate the use of arrow functions in JavaScript.
- Practical 10: Arrays and Objects
  Create a JavaScript array and perform operations like adding, removing, and updating

elements.
Define an object representing a student with properties like name, age, and grade, and access its values.

- Practical 11: DOM Manipulation
  Write a JavaScript program that dynamically changes the background color of a webpage when a button is clicked.
  Modify text content of an HTML element using getElementById().
- Practical 12: Event Handling
  Implement event handling in JavaScript using onclick and onmouseover events.
  Create an image slideshow using setTimeout() and setInterval().
- Practical 13: Form Validation
  Write a JavaScript program to validate a form (check if fields are empty, email format validation, password strength check).
- Practical 14: JavaScript Animations
  Create a simple animation effect (e.g., a div box moving across the screen).
  Implement fade-in and fade-out effects using JavaScript.
- Practical 15: Introduction to ES6 Features
  Demonstrate the difference between var, let, and const.
  Use template literals to dynamically update webpage content.
- Project for Module 2:
  Develop an interactive webpage that includes form validation, dynamic content generation, and event handling using JavaScript.

Evaluation Scheme:

Internal Assessment:

Depending on the activities mentioned above a project should be developed of 50 marks. Internal Assessment to be done on the basis of Project Presentations and Report Writing with proper Rubrics (Given below).

The Rubric will have the following Evaluation Parameters:	
Evaluation Parameters	**Marks**
Functionality (Core Features)	20
Code Quality (Structure, Readability)	10
GUI Design & User Experience	10
Input Handling & Error Management	5
Report writing or Documentation	5

The project Internal assessment will be done by conducting project presentation at the department level, where External Examiner appointed by the department will be evaluating the project.

Two sample projects are given below and it is expected from students to choose a good and advance project topic that aligns with their interests and abilities to work on.

Project 1: Personal Portfolio Website
Create a multi-page responsive personal portfolio website using HTML, CSS, and JavaScript to showcase skills, projects, contact information, and a feedback form.

Project 2: Online Student Feedback System (Frontend Only)
Develop a responsive, single-page application (SPA-style) for collecting student feedback using HTML, CSS, and JavaScript. This simulates a lightweight web app interface.

References: -

1. Lecky-Thompson, G. W. (2009). Web Programming. Cengage Learning.
2. Powell, T. (2000). Web Design: The Complete Reference. Tata McGraw Hill.
3. Powell, T. (2008). HTML and XHTML: The Complete Reference. Tata McGraw Hill.
4. Powell, T., & Schneider, F. (2004). JavaScript 2.0: The Complete Reference (2nd ed.). Tata McGraw Hill.
5. Holzner, S. (2017). PHP: The Complete Reference. Tata McGraw Hill.Semester: II

Course Code	**2.1 (Major Core)**
Course Title	**Object Oriented Programming using JAVA** **Major (Core) Theory**
Course Credits	**4**
Course Outcomes	**After completion of this Course, the students will be able to** • Understand the fundamentals of Java programming. • Apply object-oriented programming (OOP) principles. • Handle exceptions and file operations in Java. • Work with simple databases using JDBC.
Module 1: Introduction to Java & Programming Basics (1 Credit)	
Learning Outcomes	Learners will be able to: • Set up and run Java programs. • Use variables, data types, and operators. • Implement loops and conditional statements.
Module Contents	• Introduction to Java and JVM • Installing & Setting up Java (JDK, IDEs like Eclipse/IntelliJ) • Writing & Running a Java Program • Variables, Data Types, and Operators • Control Flow Statements (If-else, Loops, Switch) • Methods and Functions in Java • Arrays and Strings
Module 2: Object-Oriented Programming (OOP) Concepts (1 Credit)	
Learning Outcomes	Learners will be able to: • Understand and apply OOP concepts. • Create classes and objects in Java. • Implement inheritance, polymorphism, and encapsulation.
Module Contents	• Introduction to OOP – Why OOP? • Classes and Objects • Constructors and Overloading • Inheritance – Super & Subclasses • Polymorphism – Method Overloading & Overriding • Encapsulation – Getters & Setters • Abstraction – Abstract Classes & Interfaces
Module 3: Exception Handling and File Handling (1 Credit)	
Learning Outcomes	Learners will be able to: • Effectively handle exceptions in Java. • Differentiate between checked and unchecked exceptions. • Explore Read and write files using Java's file handling API

Module Contents	• Exception Handling – Try, Catch, Finally, Throws, Throw
	• Advanced Exception Handling
	  ○ Nested try-catch Blocks
	  ○ throws and throw Keywords
	  ○ User-Defined Exceptions
	• File Handling – Reading & Writing to Files

Module 4: Database Connectivity & Essential Java Collections (1 Credit)	
Learning Outcomes	Learners will be able to: • Establish a database connection using JDBC. • Explore Execution of SQL queries from Java programs. • Understand Handle Result Sets and Prepared Statements. • Explore Manage database transactions using JDBC.

Module Contents	**Introduction to JDBC**
	• What is JDBC?
	• JDBC Architecture & JDBC Drivers
	• Steps for Database Connectivity:
	  ○ Loading a Driver
	  ○ Connecting to a Database
	  ○ Executing SQL Statements
	Using JDBC API
	• Executing SQL Queries in Java
	• Handling SQL Exceptions (try-catch, SQLException)
	• **Accessing Result Sets**: Methods of ResultSet Interface
	• **Prepared Statements** (insert, update, delete)
	• **Managing Database Transactions** (commit, rollback)
	• Stored Procedures in JDBC
	• Using Metadata in JDBC (DatabaseMetaData, ResultSetMetaData)
	Essential Java Collections (Minimal Coverage)
	• **Lists:** ArrayList, LinkedList (Basic Usage)
	• **Maps:** HashMap, TreeMap (Storing Key-Value Pairs)

Activities towards Comprehensive Continuous Evaluation	
Module 1: • Write a simple program to print your name, department, and a welcome message. • Accept two numbers and perform +, −, ×, ÷ based on user input. • Write a program to check if a number is odd/even and print all prime numbers from 1 to 100. • Accept and print elements of an array in reverse. **Module 2:** • Create a class BankAccount with deposit, withdraw, and display methods. • Use constructors and inheritance for classes like Student, GraduateStudent. • Create abstract class Shape with subclasses Circle, Rectangle, etc. using method overriding. **Module 3:** • Write a program to handle divide-by-zero, null pointer, and array index exceptions. • Create a custom exception InvalidAgeException for voting eligibility. • Read and write student records from/to a file using BufferedReader and FileWriter. • Demonstrate exception propagation and handling for multiple error types. **Module 4:** • Connect Java to MySQL/PostgreSQL and print all records from a sample table. • Create a Java program to insert, update, delete, and view employee records using JDBC. • Store employee details in ArrayList and sort by name/salary. • Use HashMap to store and retrieve product info (ID → Name).	

Evaluation Scheme:

Internal Assessment:

Depending on the activities mentioned above a project should be developed of 50 marks. Internal Assessment to be done on the basis of Project Presentations and Report Writing with proper Rubrics (Given below).

The Rubric will have the following Evaluation Parameters:	
Evaluation Parameters	**Marks**
Functionality (Core Features)	20
Code Quality (Structure, Readability)	10
GUI Design & User Experience	10
Input Handling & Error Management	5

Report writing or Documentation	5

The project Internal assessment will be done by conducting project presentation at the department level, where External Examiner appointed by the department will be evaluating the project.

Two sample projects are given below and it is expected from students to choose a good and advance project topic that aligns with their interests and abilities to work on.

Project 1: Student Management System (Console-Based Application)
Create a console-based application that manages student records using object-oriented design, file handling, and exception handling.

Project 2: Library Book Inventory System with JDBC
Build a Java application to manage a simple library inventory using MySQL and JDBC.

External Assessment:

End Semester examination of 50 marks for 2 hours duration will be conducted by the
University.

References:

1. Java: The Complete Reference (12th Edition) – Herbert Schildt
2. Effective Java (3rd Edition) – Joshua Bloch
3. Java How to Program (10th Edition) – Deitel & Deitel
4. Core Java Volume I & II – Cay S. Horstmann

Course Code	**2.2 (Major Core)**
Course Title	**OPERATING SYSTEMS**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to** • Apply knowledge of computer system organization and architecture to solve complex problems. • Analyze the structure, operations, and design principles of operating systems. • Evaluate virtualization techniques, including virtual machines and operating system generation, for efficiency and effectiveness. • Design processor management and scheduling strategies for multi-processor systems to optimize performance.
Module 1: Introduction to Operating Systems, CPU Management and Processing (Credit1)	
Learning Outcomes	Learners will be able to: • Apply knowledge of computer system organization and architecture, with a focus on the role of operating systems. • Analyze the structure and operations of operating systems, including key components like process management, memory management, storage management, protection, and security. • Evaluate the various operating system services available to users and applications for efficiency and effectiveness. • Design understanding of the user-operating system interface, system calls, and types of system programs to enhance system functionality and usability.
Module Contents	• **Introduction to Operating Systems (OS):** Computer- System Organization, Computer-System Architecture, Operating-System Structure, Operating-System Operations, Process Management, Memory Management, Storage Management, Protection and Security, Distributed Systems, Special-Purpose Systems, Computing Environments. • Operating-System Services, User Operating-System Interface, System Calls, Types of System Calls, System Programs, Operating-System Design and Implementation, Operating-System Structure, Virtual Machines, Operating-System Generation. • **CPU Management:** Process concept, Process scheduling, Operations on Processes, Interprocess Communication, Multithreading models, threading issues, Process scheduling algorithms, Thread scheduling, Multiple processor Scheduling. • **CPU Process Coordination:** Synchronization, Semaphores, Monitors, Deadlocks characterization, Methods for handling deadlocks, Deadlock prevention, Deadlock Avoidance, Deadlock detection, recovery from deadlock.
Module 2: Memory Management and File Handling (1 Credit)	
Learning Outcomes	Learners will be able to: • Apply various memory management techniques such as swapping, contiguous memory allocation, paging, and segmentation in practical scenarios. • Analyze the structure and function of the page table to understand its role in memory addressing. • Evaluate fundamental file concepts, including file access methods, directory structures, file sharing, and protection mechanisms, to ensure efficient file management. • Design efficient file systems by understanding their organization, structure, directory implementation, allocation

	methods, and free-space management to optimize storage and retrieval operations.
Module Contents	• **Memory Management:** Swapping, Contiguous Memory Allocation, Paging, Structure of the Page Table, Segmentation • **Virtual memory Management:** Demand Paging, Copy- on-Write, Page replacement, Allocation of Frames, Thrashing. • **File Management:** File Concept, File Access Methods, Directory Structure, File Sharing, File Protection, File-System Structure

Evaluation Scheme:

External Assessment:

End Semester examination of 50 marks for 2 hours duration will be conducted by the university.

References/Text Books:-

1. Silberschatz, A., Galvin, P. B., & Gagne, G. (2005). Operating System Concepts (7th ed.). John Wiley & Sons, Inc.
2. Milenkovic, M. (2001). Operating Systems Concepts And Design (2nd ed.). McGraw-Hill International Editions.
3. Stallings, W. (2005). Operating Systems: Internals and Design Principles (5th ed.). Prentice Hall.
4. Tanenbaum, A. (2007). Modern Operating Systems (3rd ed.). Pearson Education.

Course Code	**2.3 (Minor Stream)**
Course Title	**Mathematical Foundation of Computer Science - I**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to:** • Apply sets, relations, functions, and mathematical reasoning techniques like the Binomial Theorem and Induction to solve discrete mathematical problems. • Analyze and investigate a variety of discrete mathematical problems using acquired knowledge and skills. • Evaluate the effectiveness of sets, relations, functions, and mathematical reasoning techniques in solving discrete mathematical problems. • Design solutions for discrete mathematical problems by applying sets, relations, functions, and mathematical reasoning techniques effectively.
Module 1: Foundations of Discrete Mathematics (Credit1)	
Learning Outcomes	Learners will be able to: • Apply algorithms to solve problems related to GCD, LCM, and various theorems. • Analyze and solve problems involving sets, relations, and functions. • Evaluate solutions to basic problems using the Binomial Theorem and Mathematical Induction. • Design problem-solving strategies for GCD, LCM, theorems, sets, relations, functions, the Binomial Theorem, and Mathematical Induction to address various mathematical challenges effectively.
Content Outline	• **Properties of Integers:** Definition of GCD, LCM, Theorems Euclidean algorithm and problems ➢ **Set Theory**: Definition of Sets, Subsets, Cardinality of Sets, Types of sets: Equal Sets, Universal Sets, Finite and Infinite Sets, proper set, power sets, Operations on Sets: Union, Intersection, Complement of Sets, set difference, Cartesian Product, Venn Diagrams, and Algebra of sets ➢ **Relations:** Definitions of Relation, Reflexive Relation, Symmetric Relation, Transitive relation, Equivalence Relation Recurrence Relation: Definitions and problems ➢ **Functions**: Define Function, Injective Functions, Surjective Functions, Bijective Functions, Composite Function, Inverse of a Function, Domain and Range ➢ **Binomial Theorem and Mathematics Induction:** Binomial Theorem: Statement and basic problems and Mathematical Induction: Principles and basic problems.
• **Module 2: Advanced Discrete Structures (Credit1)**	
Learning Outcomes	Learners will be able to: • Apply permutations and combinations concepts to solve mathematical problems. • Analyze and solve problems involving matrices and determinants. • Evaluate solutions to permutations, combinations, matrices, and

	determinants problems. • Design problem-solving strategies for permutations, combinations, matrices, and determinants to address various mathematical challenges effectively.
Content Outline	• **Permutations and Combinations:** Permutation: Definition, Basic Permutation, Permutation with Repetition and Circular Permutation and basic problems, Combination: Definition and basic problems • **Matrices and Determinants:** Matrices: Definition, Operations on Matrices, Square Matrix and its Inverse, Solution of Equations using Matrices, Determinants: Properties of Determinants, Solution of equations using Determinants

Evaluation Scheme:

External Assessment:

End Semester examination of 50 marks for 2 hours duration will be conducted by the university.

References: -

1. Kolman, B., Busby, R. C., & Ross, S. (Year). Discrete Mathematical Structures and Graph Theory.
2. Doerr, A., & Levasseur, K. (1988). Applied Discrete Structures for Computer Science. Galgotia Publications.
3. Trembley, J. P., & Manohar, R. (1987). Discrete Mathematical Structures with Applications to Computer Science. McGraw-Hill.
4. Chakraborty, S. K., & Sarkar, B. K. (2011). Discrete Mathematics. Oxford Higher Education.
5. Liu, C. L., & Mohapatra, D. P. (2008). Elements of Discrete Mathematics: A Computer Oriented Approach (3rd ed.). Tata McGraw-Hill.

2.4 Open Elective Course-II (OEC-II) Students will choose course from another department

Course Code	**2.5 (VSC)**
Course Title	**Computer Organization & Architecture**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to** • Understand the fundamental concepts of computer organization and architecture. • Analyze the working of CPU, memory hierarchy, and input-output systems. • Explore instruction set architectures, addressing modes, and control unit design. • Learn about pipelining, parallelism, and performance optimization techniques.
Module 1: Fundamentals of Computer Organization (1 Credit)	
Learning Outcomes	Learners will be able to • Understand the basic structure of computer systems. • Understand instruction execution and memory organization. • Describe I/O mechanisms and system buses.
Module Contents	• Introduction to Computer Organization & Architecture – Basic concepts & evolution • Von Neumann vs. Harvard Architecture • CPU Organization – Registers, ALU, Control Unit • Instruction Set Architecture (ISA**)** – Types of instructions & addressing modes • Memory Hierarchy – Cache Memory, Virtual Memory, RAM, ROM • I/O Systems & Buses – Communication between CPU and peripherals
Module 2: Advanced Concepts in Computer Architecture (1 Credit)	
Learning Outcomes	Learners will be able to: • Analyze CPU design and control unit mechanisms. • Evaluate the impact of pipelining and parallelism on performance. • Compare different processor architectures and emerging trends.
Module Contents	• Microprogrammed vs. Hardwired Control Unit • **Pipelining & Parallel Processing** – Basic pipeline structure, hazards, and solutions • RISC vs. CISC Architectures • **Multiprocessor & Multicore Systems** – Interconnection networks & shared memory • Performance Metrics – MIPS, FLOPS, Speedup, Amdahl's Law • Emerging Trends – Introductory concepts of Quantum Computing & Neuromorphic Computing

Evaluation Scheme:

External Assessment:

End Semester examination of 50 marks for 2 hours duration will be conducted by the university.

References:

1. Computer Organization and Design: The Hardware/Software Interface – David A. Patterson, John L. Hennessy
2. Computer System Architecture – M. Morris Mano

3. Computer Organization and Architecture – William Stallings
4. Structured Computer Organization – Andrew S. Tanenbaum
5. Modern Processor Design: Fundamentals of Superscalar Processors – John Paul Shen, Mikko H. Lipasti

Course Code	**2.6 (SEC)**
Course Title	**Web Technologies-II**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to** • Develop full-stack web applications with backend integration. • Implement authentication, security, and performance optimization. • Deploy applications to cloud platforms. • Work with modern frameworks (React, Vue, Angular) for dynamic UI.
Module 1: Backend Development & Databases (1 Credit)	
Learning Outcomes	Learners will be able to: • Understand the client-server architecture and server-side scripting. • Implement backend logic using a server-side language (e.g., Python, PHP, Node.js). • Integrate databases with web applications using SQL & NoSQL databases. • Handle HTTP requests and responses. •
Module Contents	**1. Introduction to Backend Development** • Understanding Client-Server Architecture • Role of Backend in Web Applications • Overview of Server-Side Programming Languages (Python, PHP, Node.js) **2. Server-Side Scripting Basics** • Introduction to Express.js (Node.js) or Flask (Python) • Handling HTTP Requests and Responses (GET, POST, PUT, DELETE) • Working with RESTful APIs **3. Databases for Web Applications** • Relational Databases (SQL) – MySQL, PostgreSQL • NoSQL Databases – MongoDB, Firebase • Connecting Web Apps with Databases **4. Authentication & Session Management** • User Authentication: JWT, OAuth, Session-based Authentication • Managing Sessions and Cookies • Role-Based Access Control (RBAC)
Module 2: Advanced Concepts (1 Credit)	
Learning Outcomes	Learners will be able to: • Implement modern front-end frameworks (React, Angular, or Vue.js). • Secure web applications using best practices. • Optimize web performance and scalability.
Module Contents	**1. Introduction to Frontend Frameworks** • Overview of React, Angular, Vue.js • Component-Based Architecture • State Management Basics **2. Web Security & Best Practices** • Common Security Threats (XSS, CSRF, SQL Injection) • HTTPS and SSL/TLS • Input Validation and Secure Authentication **3. Web Performance Optimization** (4 Hours) • Lazy Loading, Caching, Minification • Content Delivery Networks (CDN)

	• Progressive Web Apps (PWA) **4. Introduction to Cloud & DevOps** (3 Hours) • Cloud Deployment (AWS, Firebase, Heroku) • CI/CD Pipelines for Web Apps • Containerization (Docker Basics)

Activities towards Comprehensive Continuous Evaluation

Module 1:

- Practical 1: Understanding Client-Server Architecture
  Write a basic Node.js Express or Flask server that responds with "Hello, World!" on a GET request.
  Demonstrate how the client sends HTTP requests and how the server handles responses.
- Practical 2: Handling HTTP Requests and RESTful APIs
  Create a REST API in Express.js (Node.js) or Flask (Python) with:
  GET: Fetch all users
  POST: Add a new user
  PUT: Update a user
  DELETE: Delete a user
- Practical 3: Integrating Databases (SQL & NoSQL)
  SQL: Use MySQL/PostgreSQL to store and retrieve user information from a users table.
  NoSQL: Use MongoDB to store user details in a users collection.
- Practical 4: User Authentication & Session Management
  Implement JWT-based authentication in Node.js or Flask.
  Store session data using cookies and session storage.
  Implement Role-Based Access Control (RBAC) for different user roles (admin, user).

Module 2:

- Practical 5: Frontend Frameworks (React/Vue.js)
  Set up a React.js or Vue.js project using create-react-app or Vue CLI.
  Create a component-based UI with a navigation bar and pages.
- Practical 6: State Management & API Integration
  Use React's useState, useEffect (or Vue's ref, computed) to manage frontend state.
  Fetch data from the backend REST API and display it in a table.
- Practical 7: Web Security Best Practices
  Implement input validation to prevent SQL Injection and Cross-Site Scripting (XSS).
  Use HTTPS, CORS, and CSRF protection in a web application.
- Practical 8: Web Performance Optimization
  Enable lazy loading of images and components.
  Implement minification and caching techniques.
  Use a CDN for serving static assets.
- Practical 9: Cloud Deployment & DevOps
  Deploy a Node.js or Flask app to Heroku/Firebase/AWS.
  Implement Continuous Integration & Continuous Deployment (CI/CD) using GitHub Actions.
  Create a Docker container for the web application.

Evaluation Scheme:

Internal Assessment:

Depending on the activities mentioned above a project should be developed of 50 marks. Internal Assessment to be done on the basis of Project Presentations and Report Writing with proper Rubrics (Given below).

The Rubric will have the following Evaluation Parameters:	
Evaluation Parameters	**Marks**
Functionality (Core Features)	20
Code Quality (Structure, Readability)	10

GUI Design & User Experience	10
Input Handling & Error Management	5
Report writing or Documentation	5

The project Internal assessment will be done by conducting project presentation at the department level, where External Examiner appointed by the department will be evaluating the project.

Two sample projects are given below but it is expected from students to choose a good and advance project topic that aligns with their interests and abilities to work on.

Project 1: CampusConnect - College Portal System
Full-Stack Web Application

Project 2: BookNook - Online Bookstore
E-Commerce-like Application

Reference Books:

1. Casciaro, M., & Mammino, L. (2016). *Node.js Design Patterns*. Packt Publishing.
2. Grinberg, M. (2018). *Flask Web Development*. O'Reilly Media.
3. Bradshaw, S., Brazil, E., & Chodorow, K. (2019). *MongoDB: The Definitive Guide*. O'Reilly Media.
4. Banks, A., & Porcello, E. (2020). *Learning React*. O'Reilly Media.
5. McDonald, M. (2020). *Web Security for Developers*. No Starch Press.
6. Grigsby, J. (2018). *Progressive Web Apps*. O'Reilly Media.
7. Buyya, R., Broberg, J., & Goscinski, A. M. (2011). *Cloud Computing: Principles and Paradigms*. Wiley.

Semester- III

Course Code	**3.1 (Major Core)**
Course Title	**Advanced Java Programming**
Course Credits	**4**
Course Outcomes	**After completion of this Course, the students will be able to** • To Develop Graphical User Interfaces (GUI) using Swing & AWT. • Implement event handling mechanisms for interactive applications. • Understand the fundamentals of servlets, JSP, Spring, and Hibernate. • Work with database connectivity using Hibernate ORM.
Module 1: (1 Credit)	
Learning Outcomes	Learners will be able to: • Understand Java AWT & Swing frameworks. • Design interactive GUI applications. • Implement various layout managers
Module Contents	**Introduction to AWT (Abstract Window Toolkit)** • AWT Components (Labels, Buttons, Text Fields, Checkboxes) • AWT Layout Managers (FlowLayout, BorderLayout, GridLayout) **Swing Basics** • JFrame, JPanel, JButton, JLabel, JTextField, JTextArea • JTable, JTree, JList, JTabbedPane, Advanced Swing Features (Dialogs, Menus, Toolbars) **Event Handling in GUI** • Event Listeners (ActionListener, MouseListener, KeyListener) • Adapter Classes Advanced Swing Features
Module 2: Event Handling & MVC Design Pattern (1 Credit)	
Learning Outcomes	Learners will be able to: • Implement event-driven programming in Java. • Understand Java's event delegation model. • Apply the MVC pattern in GUI development.
Module Contents	**Java Event Handling Model** • Event Sources & Listeners • Delegation Event Model **Event Classes & Listener Interfaces** • ActionEvent, MouseEvent, KeyEvent • ActionListener, MouseListener, WindowListener **Model-View-Controller (MVC) Pattern** • Separation of Concerns in GUI • Implementing MVC with Swing
Module 3: Introduction to Servlets & JSP (1 Credit)	
Learning Outcomes	Learners will be able to: • Understand the role of Servlets in Java Web Development. • Implement JSP for dynamic web applications. • Work with session management and cookies.
Module Contents	**Introduction to Web Applications** • Client-Server Architecture • Role of Servlets & JSP in Java Web Development

	Servlet Basics • Servlet Lifecycle (init(), service(), destroy()) • Handling GET & POST requests **JSP Basics** • JSP vs Servlets • JSP Scripting Elements (Declarations, Expressions, Scriptlets) **Session Management & Cookies** • HTTP Sessions • URL Rewriting & Hidden Fields
Module 4: Introduction to Spring & Hibernate (1 Credit)	
Learning Outcomes	Learners will be able to: • Understand the basics of Spring Framework. • Implement Hibernate ORM for database interaction. • Build simple Spring Boot applications.
Module Contents	**Introduction to Spring Framework** • Inversion of Control (IoC) & Dependency Injection (DI) • Spring MVC Architecture **Getting Started with Hibernate** • What is ORM? • Hibernate vs JDBC • Hibernate Annotations & Configuration **Spring Boot Basics** • Creating a Simple Spring Boot Application Connecting with Hibernate
Assignments/ Activities towards Comprehensive Continuous Evaluation	
Module 1: • Create a GUI using AWT components: Labels, TextFields, Checkboxes, and Buttons. o Apply various Layout Managers: FlowLayout, BorderLayout, and GridLayout. o Extend the GUI using Swing components like: • JFrame, JPanel, JButton, JLabel • JTextField, JTextArea, JList, JTable o Add Menus, Dialogs, and Toolbars. o Submit screenshots of working application and code snippets. **Module 2:** • Design an interactive calculator or quiz app using Swing. • Implement event handling using: o ActionListener, MouseListener, KeyListener • Use Adapter classes where appropriate. • Refactor the code to follow the MVC design pattern: o Model: business logic (e.g., calculation, score tracking) o View: GUI components o Controller: event handling and user interaction **Module 3:** • Design a login system with: o Login Page (JSP) o LoginServlet to process authentication o Home Page after successful login • Implement: o Servlet lifecycle methods o Handling GET and POST requests	

- Use JSP Scripting Elements to display user info dynamically.
- Add session tracking using:
  - HTTP Sessions
  - Cookies or URL rewriting

Module 4:

- Create a Spring Boot application for managing a product inventory or student records.
- Use:
  - Dependency Injection (DI)
  - Spring MVC Controllers
- Integrate Hibernate ORM for database operations.
  - Use annotations to map entities.
  - Perform basic CRUD: Create, Read, Update, Delete
- Connect to MySQL or H2 database.

Evaluation Scheme:

Internal Assessment:

Depending on the activities mentioned above a project should be developed of 50 marks. Internal Assessment to be done on the basis of Project Presentations and Report Writing with proper Rubrics (Given below).

The Rubric will have the following Evaluation Parameters:	
Evaluation Parameters	**Marks**
Functionality (Core Features)	20
Code Quality (Structure, Readability)	10
GUI Design & User Experience	10
Input Handling & Error Management	5
Report writing or Documentation	5

The project Internal assessment will be done by conducting project presentation at the department level, where External Examiner appointed by the department will be evaluating the project.

Two sample projects are given below and it is expected from students to choose a good and advance project topic that aligns with their interests and abilities to work on.

Project 1: Student Information Management System
Full-Stack Java Web Application

Project 2: Product Inventory Management System
End-to-End Java Application (Desktop + Web)

External Assessment:

End Semester examination of 50 marks for 2 hours duration will be conducted by the University.

References:

1. Core Java Volume I – Fundamentals (11th Edition) – Cay S. Horstmann
2. Java: The Complete Reference (11th Edition) – Herbert Schildt
3. Head First Design Patterns – Eric Freeman, Bert Bates

4. Murach’s Java Servlets and JSP (3rd Edition) – Joel Murach
5. Head First Servlets and JSP (2nd Edition) – Kathy Sierra, Bert Bates
6. Spring in Action (6th Edition) – Craig Walls
7. Java Persistence with Hibernate (2nd Edition) – Christian Bauer, Gavin King

Course Code	**3.2 (Major Core)**
Course Title	**DATABASE MANAGEMENT SYSTEMS**
Course Credits	**4**
Course Outcomes	**After completion of this Course, the students will be able to** • Understand and apply database concepts. • Analyze and Design ER models and apply relational model. • Analyze, normalize relational design. • Evaluate and interpret the use of transactions, schedules.
Module 1: Introduction to Database Model (1 Credit)	
Learning Outcomes	Learners will be able to: • Understand the importance of database systems • Understand structure and architecture of DBMS systems. • Design and apply concepts of the ER Modelling. • Understand various relational models in DBMS
Module Contents	• **Introduction to Database Management Systems**: Database System Applications, Database Systems versus File Systems, View of Data, Data Models, Database Languages, Database Users and Administrators, DBA Roles and activity, Database System Structure • **Entity –Relational Model:** Basic Concepts, Constraints, Keys, Entity-Relationship Diagram, Weak Entity Sets, Extended E-R features, Design of E-R Database Schema, Reduction of an E-R Schema to Tables. • **Relational Model:** Structure of Relational Databases, Relational Algebra, Tuple Relational Calculus, Domain Relational Calculus
Module 2: SQL and Database Design (1 Credit)	
Learning Outcomes	Learners will be able to: • Learn the syntax and semantics of SQL. • Gain Proficiency in writing SQL queries • Analyze schema design and normalize the schema.
Module Contents	• **SQL:** SQL commands, Functions, Data Constraints, Grouping Data, Subqueries, Joins, Performance Tuning, Security Management • **Integrity & Security:** Domain Constraints, Referential Integrity, Assertions, Triggers, Privileges in SQL • **Relational Database Design:** Functional Dependencies, Decomposition, Normalization – 1NF – 5NF, BCNF
Module 3: Storage and Transactions (1 Credit)	
Learning Outcomes	Learners will be able to: • Understand indexing file structure. • Understand and apply concept of transactions. • Apply serializability testing.
Module Contents	• **Storage & File Structure:** RAID, Improvement of Reliability & Performance Indexing & Hashing – Basic Concepts, Ordered Indices, B+ & B Tree Index Files, Static & Dynamic Hashing, Comparison of Ordered Indexing & Hashing. • **Transactions:** Transaction Concept & State, Implementation of

	Atomicity & Durability, Serializability, Recoverability, Testing for Serializability.
Module 4: Concurrency control and Recovery system (1 Credit)	
Learning Outcomes	Learners will be able to: • Learn and apply Concurrency protocol • Learn and apply log based recovery techniques.
Module Contents	• **Concurrency Control:** Protocols- Lock Based, Timestamp-based, Validation Based, Deadlock Handling & Concurrency • **Recovery System:** Failure Classification, Storage Structure, Recovery & Atomicity, Log based Recovery, Shadow Paging, Recovery with Concurrent Transactions, Buffer management, failure with loss of nonvolatile storage, advanced recovery techniques.
Assignments/ Activities towards Comprehensive Continuous Evaluation	
Module 1: • Design a database schema and ER diagram for a real-world application like: o Hospital Management System o E-commerce Website o Online Banking System Module 2: • Implement various SQL queries using DDL and DML. • Normalize a given unoptimized schema to BCNF or 5NF. Module 3: • Implement transactions with rollback, commit, and isolation levels. Module 4: • Simulate multi-user concurrent database access with lock-based mechanisms. • Implement log-based recovery techniques after a simulated database crash.	

Evaluation Scheme:

Internal Assessment:

Depending on the activities mentioned above a project should be developed of 50 marks. Internal Assessment to be done on the basis of Project Presentations and Report Writing with proper Rubrics (Given below).

The Rubric will have the following Evaluation Parameters:	
Evaluation Parameters	**Marks**
Functionality (Core Features)	20
Code Quality (Structure, Readability)	10
GUI Design & User Experience	10
Input Handling & Error Management	5
Report writing or Documentation	5

The project Internal assessment will be done by conducting project presentation at the department level, where External Examiner appointed by the department will be evaluating the project.

Two sample projects are given below and it is expected from students to choose a good and advance project topic that aligns with their interests

and abilities to work on.

Project 1: Hospital Management System (HMS)
Build a relational database system to manage hospital operations including patient records, doctor schedules, treatment logs, billing, and pharmacy inventory.

Project 2: Online Course Registration System
Create a backend database system to manage online course enrollments, student profiles, instructors, course prerequisites, and grading.

External Assessment:

End Semester examination of 50 marks for 2 hours duration will be conducted by the University.

References:

1. Korth, Henry, Abraham Silberschatz, and S. Sudarshan. Database System Concepts. 5th Edition. McGraw-Hill.
2. Elmasri, Ramez, and Shamkant B. Navathe. Fundamentals of Database Systems. 3rd Edition. Pearson Education India, 2008.
3. Ramakrishnan, Raghu, and Johannes Gehrke. Database Management Systems. McGraw-Hill International Edition, 2002.
4. Hoffer, Jeffrey A., Mary Prescott, and Fred McFadden. Modern Database Management. 7th Edition. Prentice Hall, 2004.
5. Rob, Peter, and Carlos Coronel. Database Systems: Design, Implementation and Management. Thomson Publication, 2004.
6. Kroenke, David M. Database Processing: Fundamentals, Design, Implementation. 10th Edition. Prentice-Hall, 2005.
7. O'Neil, Patrick. Database Principles Programming Performance. 2nd Edition. Morgan Kaufmann Publishers, Inc., 2000.

Course Code	**3.3 (Minor Stream)**
Course Title	**Mathematical Foundation of Computer Science – II**
Course Credits	**4**
Course Outcomes	**After completion of this Course, the students will be able to** • Apply logic and proof techniques in mathematical reasoning. • Understand algebraic structures such as groups and semigroups. • Utilize numerical methods for solving equations, interpolation, and integration. • Implement linear programming
Module 1: Logic and Methods of Proofs (1 Credit)	
Learning Outcomes	Learners will be able to: • Understand propositional logic and truth tables. • Apply logical equivalence, CNF, and DNF transformations. • Utilize rules of inference and proof techniques.
Module Contents	• **Propositional Logic**: Logical operations, Basic connectives, Compound statements, Construction of truth tables. • **Logical Reasoning**: Quantifiers, Conditional statements, Tautology, Contradiction, Contingency, Logical equivalence. • **Normal Forms**: Conjunctive Normal Form (CNF), Disjunctive Normal Form (DNF). • **Rules of Inference**: Modus Ponens, Modus Tollens, Syllogism. • **Methods of Proofs**: Proof by contradiction, Mathematical induction.
Module 2: Algebraic Structures (1 Credit)	
Learning Outcomes	Learners will be able to: • Explain fundamental algebraic structures. • Differentiate between semi-groups, monoids, and groups. • Apply group theory in cryptography and coding theory.

Module Contents	• **Fundamental Structures**: Sets, Relations, Binary operations.
	• **Algebraic Systems**:
	  ○ **Semi-group**: Definition, Associativity.
	  ○ **Monoid**: Identity element.
	  ○ **Group**: Definition, Properties, Subgroups, Cyclic groups.
	• **Applications**: Cryptography, Coding theory.

Module 3: Fundamentals of Numerical Computation (1 Credit)	
Learning Outcomes	Learners will be able to: • Understand different types of numerical errors and their impact on computations. • Apply numerical techniques to find roots of equations using Bisection and Newton-Raphson methods. • Analyse the accuracy and efficiency of root-finding techniques. • Implement numerical methods in solving real-world scientific and engineering problems.
Module Contents	• **Errors in Computation:** Concept and significance of numerical errors. • **Root Finding Methods:** • Bisection method • Newton-Raphson method

	• **Applications:** Basic numerical solutions to scientific and engineering problems.
Module 4: Interpolation and Numerical Integration (1 Credit)	
Learning Outcomes	Learners will be able to: • Understand interpolation methods for estimating values between known data points. • Apply Newton's and Lagrange's interpolation formulas to approximate functions. • Use numerical integration techniques (Trapezoidal and Simpson's rules) to compute definite integrals. • Solve real-world scientific and engineering problems using interpolation and numerical integration.
Module Contents	**Interpolation Techniques:** • Newton's Forward Interpolation • Newton's Backward Interpolation • Lagrange's Interpolation Formula **Numerical Integration:** • Trapezoidal rule • Simpson's 1/3 rule **Applications:** Numerical solutions to scientific and engineering problems.
Assignments towards Comprehensive Continuous Evaluation	
Module 1: • Construct truth tables for given logical expressions and determine tautologies, contradictions, and contingencies. • Convert logical statements into CNF and DNF forms and verify equivalence. • Apply rules of inference (Modus Ponens, Modus Tollens, Syllogism) to prove given logical statements. • Solve problems using proof techniques (proof by contradiction, mathematical induction). • Write a short report on the importance of propositional logic in computer science applications (e.g., circuit design, AI). **Module 2:** • Identify and prove whether given algebraic structures (sets with binary operations) are semigroups, monoids, or groups. • Solve problems involving subgroup verification and cyclic groups. • Discuss the role of group theory in cryptography with an example of RSA encryption. • Implement a simple coding theory application using group properties (e.g., error detection with parity checks). • Write a comparison of the properties of semigroups, monoids, and groups with real-world examples. **Module 3:** • Explain and analyze the types of numerical errors with examples. • Solve root-finding problems using the Bisection Method for given equations. • Apply Newton-Raphson method to find roots of nonlinear equations. • Compare the accuracy and convergence of Bisection and Newton-Raphson methods. • Implement a Python program for root-finding using the Bisection and Newton-Raphson methods. **Module 4:**	

- Compute interpolated values using Newton's Forward and Backward Interpolation methods.
- Solve function approximation problems using Lagrange's Interpolation formula.
- Apply the Trapezoidal rule to evaluate definite integrals numerically.
- Use Simpson's 1/3 rule to estimate integrals and compare accuracy with the Trapezoidal rule.
- Develop a Python program to implement interpolation and numerical integration techniques.

Evaluation Scheme:

Internal Assessment:

Depending on the activities mentioned above a project should be developed of 50 marks. Internal Assessment to be done on the basis of Project Presentations and Report Writing with proper Rubrics (Given below).

The Rubric will have the following Evaluation Parameters:	
Evaluation Parameters	**Marks**
Functionality (Core Features)	20
Code Quality (Structure, Readability)	10
GUI Design & User Experience	10
Input Handling & Error Management	5
Report writing or Documentation	5

The project Internal assessment will be done by conducting project presentation at the department level, where External Examiner appointed by the department will be evaluating the project.

Two sample projects are given below and it is expected from students to choose a good and advance project topic that aligns with their interests and abilities to work on.

Project 1: Cryptography and Coding Theory with Group Theory
This project focuses on applying group theory concepts to cryptography and error detection. Students will explore how algebraic structures such as groups, semigroups, and monoids are used in the field of cryptography, with a specific focus on RSA encryption. Additionally, students will implement error detection methods in coding theory using group properties.

Project 2: Numerical Methods for Solving Engineering Problems
This project focuses on the application of numerical methods to solve real-world engineering problems, such as finding roots of equations, interpolating functions, and evaluating definite integrals. Students will apply techniques like Bisection Method, Newton-Raphson Method, and Numerical Integration (Trapezoidal and Simpson's rules) to solve problems related to engineering computations.

External Assessment:

End Semester examination of 50 marks for 2 hours duration will be conducted by the University.

Reference Books:

1. Kenneth H. Rosen, *Discrete Mathematics and Its Applications*, 7th Edition, McGraw-Hill, 2012.
2. Epp Susanna S., *Discrete Mathematics with Applications*, 4th Edition, Cengage

Learning, 2011.
3. David Gries, *The Science of Programming*, Springer, 1981.
4. John B. Fraleigh, *A First Course in Abstract Algebra*, 7th Edition, Pearson, 2002.
5. Joseph A. Gallian, *Contemporary Abstract Algebra*, 9th Edition, Cengage Learning, 2017.
6. Richard L. Burden & J. Douglas Faires, *Numerical Analysis*, 10th Edition, Cengage Learning, 2015.
7. M.K. Jain, S.R.K. Iyengar, & R.K. Jain, *Numerical Methods for Scientific and Engineering Computation*, 6th Edition, New Age International, 2012.

3.4OEC Courses to be taken from university basket (2 Credit)

Course Code	**3.5 (VSC)**
Course Title	**Introduction to Microprocessor and Microcontroller**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to** • Understand the fundamental concepts of microprocessors and microcontrollers. • Explain the architecture, instruction set, and interfacing techniques of microprocessors and microcontrollers. • Develop assembly language programs for microprocessors and microcontrollers. • Implement basic interfacing circuits and applications using microcontrollers.
Module 1: Introduction to Microprocessors (1 Credit)	
Learning Outcomes	Learners will be able to: • Describe the evolution and basic architecture of microprocessors. • Explain the working of the 8085 microprocessor and its instruction set. • Develop simple assembly language programs for 8085. • Demonstrate memory and I/O interfacing techniques.
Module Contents	• Introduction to Microprocessors: Evolution, basic architecture, and comparison with microcontrollers. • 8085 Microprocessor Architecture: Functional blocks, registers, ALU, control unit, and memory organization. • Addressing Modes and Instruction Set: Classification of instructions, timing diagrams. • 8085 Programming: Assembly language programming with examples. • Memory and I/O Interfacing: Interfacing memory, input/output devices, and interrupt handling. • Introduction to Advanced Microprocessors: Overview of 8086, comparison with modern processors.
Module 2: Introduction to Microcontrollers (1 Credit)	
Learning Outcomes	Learners will be able to: • Differentiate between microprocessors and microcontrollers. • Explain the internal architecture and features of the 8051 microcontrollers. • Develop assembly language programs for 8051. • Implement basic interfacing techniques and applications using microcontrollers.
Module Contents	• Introduction to Microcontrollers: Differences from microprocessors, applications. • 8051 Microcontroller Architecture: Internal structure, CPU, memory, I/O ports, and timers. • Instruction Set and Assembly Programming: Data transfer, arithmetic, logical, and branching instructions. • Interfacing and Applications: GPIO, timers, serial communication, ADC/DAC, sensor interfacing. • Embedded System Basics: Basics of C programming for microcontrollers, simple projects like LED blinking, LCD display,

	and motor control.

Assignments/ Activities towards Comprehensive Continuous Evaluation

Module 1:

- Practical 1: Understanding Microprocessor Basics
  What are the key differences between a microprocessor and a microcontroller?
  Explain the architecture of the 8085 microprocessor with a neat block diagram.
  Write a simple 8085 assembly language program (ALP) to add two 8-bit numbers stored in registers.

- Practical 2: 8085 Assembly Language Programming
  Write an 8085 ALP to perform the following:
  Addition of two 8-bit numbers.
  Subtraction of two 8-bit numbers.
  Multiplication using repeated addition.
  Division using repeated subtraction.
  Develop an 8085 ALP to check if a given number is even or odd.
  Write an 8085 ALP to find the largest and smallest number in an array.
  Write an 8085 ALP to count the number of 1's in an 8-bit binary number stored in a register.

- Practical 3: Memory & I/O Interfacing
  Demonstrate memory read and write operations in the 8085 microprocessor.
  Design an interfacing circuit for connecting an LED to the 8085 microprocessor and write an assembly program to turn it ON and OFF.

- Practical 4: Interrupt Handling & Advanced Microprocessors
  Explain interrupts in the 8085 microprocessor and demonstrate a simple interrupt handling program.
  Compare 8086 with modern microprocessors (Pentium, Core i-Series).

Module 2:

- Practical 5: Understanding 8051 Microcontroller
  Explain the internal architecture of the 8051 microcontroller with a block diagram.
  Compare 8051 microcontrollers with 8085 microprocessors in terms of architecture and applications.

- Practical 6: 8051 Assembly Language Programming
  Write an 8051 ALP to perform addition, subtraction, multiplication, and division of two 8-bit numbers.
  Write an 8051 ALP to move a block of data from one memory location to another.
  Implement a looping and conditional statement in 8051 assembly language (e.g., to generate a sequence of numbers).

- Practical 7: GPIO & Timer Interfacing
  Write an 8051 ALP to toggle an LED connected to a port using a delay loop.
  Demonstrate the use of timers in the 8051 microcontroller with an assembly language program.

- Practical 8: Serial Communication & ADC/DAC Interfacing
  Write an 8051 program for serial communication to send a string to a computer using UART.
  Interface an ADC with 8051 and write a program to read an analog voltage.
  Interface a DAC with 8051 and write a program to generate a sine wave.

- Practical 9: Sensor Interfacing & Embedded System Basics

Interface a temperature sensor with the 8051 microcontroller and display the temperature on an LCD screen.
Write a C program for the 8051 microcontroller to blink an LED at regular intervals.

- Develop a mini-project using a microcontroller, such as:
  Automated temperature monitoring system.
  Simple home automation using sensors and relays.
  Basic motor speed control using PWM.

Evaluation Scheme:

Internal Assessment:

Depending on the activities mentioned above a project should be developed of 50 marks. Internal Assessment to be done on the basis of Project Presentations and Report Writing with proper Rubrics (Given below).

The Rubric will have the following Evaluation Parameters:	
Evaluation Parameters	**Marks**
Functionality (Core Features)	20
Code Quality (Structure, Readability)	10
GUI Design & User Experience	10
Input Handling & Error Management	5
Report writing or Documentation	5

The project Internal assessment will be done by conducting project presentation at the department level, where External Examiner appointed by the department will be evaluating the project.

Two sample projects are given below and it is expected from students to choose a good and advance project topic that aligns with their interests and abilities to work on.

Project 1: Smart Temperature-Controlled Fan System
Design a system where a fan turns ON/OFF automatically based on room temperature. This system uses the 8051 microcontroller to read data from a temperature sensor (e.g., LM35), processes it, and drives a DC fan using a relay.

Project 2: Home Security System with Keypad Access
Build a home security system that activates an alarm when motion is detected and can only be disarmed using a password via a 4x4 keypad. The system uses 8051 for main logic and simulates password validation in 8085.

References:

6. Ramesh Gaonkar, Microprocessor Architecture, Programming, and Applications with 8085
7. Kenneth J. Ayala, The 8051 Microcontroller
8. Douglas V. Hall, Microprocessors and Interfacing
9. Muhammad Ali Mazidi, The 8051 Microcontroller and Embedded Systems
10. Raj Kamal, Embedded Systems: Architecture, Programming and Design

3.6 Ability Enhancement Courses (AEC) to be chosen from university basket (2credit)

3.7 Field project

- MCA Integrated students undertaking field projects in Java Programming and Databases should begin by selecting a real-world problem relevant to industries such as healthcare, finance, or e-commerce, with clear objectives and expected outcomes.
- They should set up a development environment using Java (JDK 17+), an IDE like IntelliJ or Eclipse, and a version control system such as Git. Database selection depends on project requirements, with options including relational databases like MySQL and PostgreSQL or NoSQL databases like MongoDB.
- Proper database design, including ER diagrams and normalization, ensures efficiency, while SQL queries and stored procedures enhance functionality. Java programming should follow OOP principles, with best practices such as modularity, exception handling, and design patterns like MVC.
- Backend development should involve JDBC or ORM frameworks like Hibernate, with RESTful APIs built using Spring Boot and secured through JWT or OAuth authentication. If a frontend is required, JavaFX, JSP, or modern frameworks like React can be used.
- Testing should be conducted using JUnit and Mockito, with API validation in Postman. Deployment can be done on cloud platforms such as AWS, Google Cloud, or Heroku, ensuring security and scalability.
- Comprehensive documentation, including an SRS report, ER diagrams, and UML models, is essential.
- Finally, students should prepare a structured project report and presentation, demonstrating their work with a live execution while being ready for evaluation and questions from faculty or industry experts.

3.8 Co-curriculum course-III (CC) to be taken from the University Basket (2 Credits)

Semester -IV

Course Code	**4.1 (Major Core)**
Course Title	**Data Structures and Algorithms**
Course Credits	**4**
Course Outcomes	**After completion of this Course, the students will be able to** • Analyze the time and space complexities of basic data structure operations. • Analyze the time and space complexities of algorithms. • Apply data structures to solve specific computational problems • Evaluate the efficiency and suitability of data structures in different scenarios.
Module 1: Linear Data Structures (1 Credit)	
Learning Outcomes	Learners will be able to: • Define the concept of data structures and their role in organizing and storing data. • Identify and classify linear data structures such as arrays, linked lists, stacks, and queues. • Analyze the advantages and limitations of each linear data structure
Module Contents	• **Introduction:** Data types, ADT, data structure: Definition & classification Analysis of algorithms (recursive and non-recursive) with emphasis on best case, average case and worst case • **Linear Data structures with applications:** ○ **List:** Introduction, implementation using array & linked list (singly, doubly, circular, multi-list), Applications: Polynomial representation, Sparse matrix ○ **Stack:** Introduction, implementation using array & linked list, Applications: Function call, Recursion, balancing of parenthesis, Polish Notation: infix to post fix conversion and evaluation of post fix expression ○ **Queue:** Introduction (queue, circular queue, deque, priority queue), implementation using array &linked list, Applications: Job Scheduling
Module 2: Non-Linear Data Structures (1 Credit)	
Learning Outcomes	Learners will be able to: • Differentiate between linear and non-linear data structures. • Understand tree structures, including binary trees and n-ary trees. • Implement tree traversal algorithms (e.g., in-order, preorder, post-order). • Define graphs and their components (vertices and edges). • Implement basic graph traversal algorithms and graph related operations. • Analyze the advantages and limitations of each nonlinear data structure
Module Contents	**Non-Linear data structures:** • **Tree:** Introduction and representation, Forest, Tree traversal, Binary Tree (representation using array and links): Binary tree traversal (recursive & non-recursive implementation), Expression tree • **Graph:** Introduction, representations, Traversal (BFS, DFS), Applications: Shortest path (Single source-all destinations), Minimal spanning tree (Prim's algorithm, Kruskal's algorithm)
Module 3: Searching and Sorting (1 Credit)	

Learning Outcomes	Learners will be able to: • Explain the concept of hashing and hash functions. • Implement and analyse hash tables for efficient data retrieval • Apply data structures to solve real-world problems and scenarios. • Understand and Implement search's and Sort's in various scenarios. • Analyze the time and space complexity of search and sort algorithms. • Compare and contrast the efficiency of various sorting algorithms. • Choose the most appropriate sorting algorithm for specific scenarios
Module Contents	• **Searching and Sorting:** Linear Search, Binary Search, Transpose sequential search, Binary search tree, Heap tree (application in priority queue and sorting), AVL tree, Splay tree, M-way search tree, B tree (insertion), B+ tree (Definition and introduction), Application of B tree and B+ tree in File Structures • **Hash Tables:** Introduction, hash functions and hash keys, Collisions, Resolving collisions, Rehashing Sorting with algorithm analysis (best case, worst case, average): Bubble, Selection, Insertion, Shell, Merge, Quick, Heap, Radix
Module 4: NP-Completeness and the P and NP Classes (1 Credit)	
Learning Outcomes	Learners will be able to: • Understand the concept of P and NP Classes, NP-hard, NP - Completeness • Analyze various problems and scenarios
Module Contents	• NP-Completeness and the P & NP Classes • Introduction, Polynomial Time & Verification, NP-Completeness and Reducibility, The Vertex Cover Problem, The Traveling Salesman Problem, The Set Covering Problem
Assignments/ Activities towards Comprehensive Continuous Evaluation	
Module 1: • Concepts & Classification: Define data structures and explain their role in organizing and storing data. Differentiate between primitive data types, ADT (Abstract Data Type), and data structures. Compare array-based and linked list-based implementations of lists in terms of memory usage and performance. • Array & Linked List Operations: Implement basic array operations: Insertion, Deletion, Searching, and Traversal. Implement a singly linked list, doubly linked list, and circular linked list with operations (Insert, Delete, Search, Display). Represent a sparse matrix using a linked list and perform addition of two sparse matrices. • Stack Applications: Explain stack operations using an array and linked list. Convert an infix expression to postfix notation using a stack. Implement an algorithm to evaluate a postfix expression. • Queue Applications:	

Differentiate between Queue, Circular Queue, and Deque.
Implement a priority queue using an array and linked list.
Write an algorithm for job scheduling using a queue.

Module 2:

- Trees and Traversal:
  Differentiate between binary trees and n-ary trees.
  Implement preorder, inorder, and postorder traversal of a binary tree (both recursive and non-recursive).
  Construct and evaluate an expression tree for a given mathematical expression.

- Graph Representation & Traversal:
  Explain different graph representations (Adjacency Matrix vs. Adjacency List).
  Implement BFS (Breadth-First Search) and DFS (Depth-First Search) for a given graph.
  Solve the single-source shortest path problem using Dijkstra's Algorithm.

- Graph Algorithms:
  Implement Kruskal's Algorithm to find the Minimum Spanning Tree.
  Implement Prim's Algorithm to find the Minimum Spanning Tree.
  Given a graph, find its connected components.

Module 3:

- Searching Algorithms:
  Compare Linear Search vs. Binary Search with time complexity analysis.
  Implement Transpose Sequential Search and analyze its efficiency.
  Implement a Binary Search Tree (BST) and perform insert, delete, and search operations.

- Sorting Algorithms:
  Implement and analyze the time complexity of:
  Bubble Sort
  Selection Sort
  Insertion Sort
  Shell Sort
  Merge Sort
  Quick Sort
  Heap Sort
  Radix Sort
  Compare the best-case, average-case, and worst-case complexities of sorting algorithms.

- Hashing & Hash Tables:
  Explain the concept of hashing, hash functions, and collision resolution techniques.
  Implement a hash table using open addressing.
  Implement a hash table with chaining (linked list-based collision handling).

- Tree-Based Searching & Sorting:
  Implement an AVL Tree and perform insertion and deletion operations.
  Implement a B-Tree and perform insertion operations.
  Explain the application of B+ Tree in file structures.

Module 4:

- Complexity Classes & Reductions:
  Explain the difference between P, NP, NP-Hard, and NP-Complete problems.
  What is Polynomial Time Reduction, and how does it help classify problems as NP-Complete?
  Explain the concept of verification in NP problems.

- NP-Complete Problems:

Discuss the Vertex Cover Problem and prove it is NP-Complete.
Discuss the Traveling Salesman Problem (TSP) and why it is NP-Hard.
Explain the Set Covering Problem and its computational complexity.

- Applications & Challenges:
  Discuss real-world applications of NP-Complete problems.
  Why can't we solve NP-Complete problems efficiently?
  Discuss any approximation algorithm for an NP-Hard problem.

Evaluation Scheme:

Internal Assessment:

Depending on the activities mentioned above a project should be developed of 50 marks. Internal Assessment to be done on the basis of Project Presentations and Report Writing with proper Rubrics (Given below).

The Rubric will have the following Evaluation Parameters:	
Evaluation Parameters	**Marks**
Functionality (Core Features)	20
Code Quality (Structure, Readability)	10
GUI Design & User Experience	10
Input Handling & Error Management	5
Report writing or Documentation	5

The project Internal assessment will be done by conducting project presentation at the department level, where External Examiner appointed by the department will be evaluating the project.

Two sample projects are given below and it is expected from students to choose a good and advance project topic that aligns with their interests and abilities to work on.

Project 1: Campus Navigation System Using Graphs and Dijkstra's Algorithm
Build a campus navigation tool that helps students find the shortest path between two buildings on campus, similar to Google Maps but limited to a known, static graph (nodes = buildings, edges = walkways/roads).

Project 2: Student Management System with Dynamic Data Structures
Create a Student Information Management System that handles operations like add, search, delete, and update student records, using different dynamic data structures depending on the task.

External Assessment:

End Semester examination of 50 marks for 2 hours duration will be conducted by the University.

References:

1. Lipschutz, S. (2017).Data Structures with C. McGraw-Hill.
2. Weiss, M. A. (2003). Data Structures and Algorithm Analysis in C (2nd ed.). Pearson Education.
3. Pai, G. A. V. (2008). Data Structures and Algorithms: Concepts, Techniques, and Applications (1st ed.).
4. Horowitz, E., Sahni, S., & Anderson-Freed, S. (2007). Fundamentals of Data Structures in C (2nd ed.). University Press.

5. Tremblay, J.-P., & Sorenson, P. G. (2007). An Introduction to Data Structures with Applications (2nd ed.). Tata McGraw-Hill.
6. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2003). Introduction to Algorithms (2nd ed.). PHI.
7. Kanetkar, Y. (BPB publications). Data Structures Through C

Course Code	**4.2 (Major Core)**
Course Title	**Data Communications and Networking**
Course Credits	**4**
Course Outcomes	**After completion of this Course, the students will be able to** • Apply understanding of key concepts in data communications, including digital and analog signals, bandwidth, data transmission modes, and network types. • Analyze the OSI (Open Systems Interconnection) model and TCP/IP protocol suite, understanding their structure, functions, and importance in data communication and networking. • Evaluate different networking protocols such as TCP, UDP, IP, and HTTP, understanding their functions, characteristics, and application scenarios. • Design and implement basic computer networks based on specific requirements, including choosing the appropriate networking equipment and setting up connections.
Module 1: Fundamentals of Computer Networks and Data Communication (1 Credit)	
Learning Outcomes	Learners will be able to: • Apply understanding of the concept of computer networks and their significance in modern computing, including data sharing, resource sharing, and communication efficiency. • Analyze the purposes and advantages of connecting computers in a networked environment such as improved collaboration, resource sharing, and cost-effectiveness. • Differentiate between common network topologies like bus, star, ring, mesh, and tree, understanding their structure and working principles. • Evaluate the strengths and weaknesses of various topologies in different scenarios, considering factors like scalability, fault tolerance, installation and maintenance costs.
Module Contents	• **Introduction:** Computer Networks and its uses, Network categorization and Hardware: Broadcast and point-to-point networks, Local Area Network (LAN), Metropolitan Area Network (MAN), Wide Area Networks (WAN), Inter networks, Topologies, Wireless Networks, Network Software: Protocols, Services, network architecture, design issues, OSI Reference model, TCP/IP Reference model, Comparison of OSI and TCP/IP Models. Introduction to Example Networks: Internet, Connection Oriented Networks–X.25, Frame Relay, ATM • **Data Communication Model**: Digital and Analog data and signals, bit rate, baud, bandwidth, Nyquist bit rate, Guided Transmission Media – Twisted Pair, Coaxial cable, Optical fiber; wireless transmission–Radio waves, microwaves, infrared waves; Satellite Communication.
Module 2: Switching Techniques, Multiplexing, and Error Control (1 Credit)	
Learning Outcomes	Learners will be able to: • Differentiate circuit switching from other switching techniques like packet switching and message switching. Circuit switching involves establishing a dedicated communication path between devices for the duration of their conversation. • Describe the process of establishing a circuit in a circuit-switched network, which includes call setup, data transfer, and call teardown phases. • Analyze the advantages and disadvantages of circuit switching. Advantages include predictable performance and minimal delay,

	while disadvantages include inefficiency in resource utilization and potential for wasted capacity. • Understand common circuit switching protocols such as Integrated Services Digital Network (ISDN), which is a set of communication standards for simultaneous digital transmission of voice, video, data, and other network services.
Module Contents	• **Switching:** Circuit Switching, Packet switching; Multiplexing: Frequency Division Multiplexing, Time Division Multiplexing, Synchronous and Asynchronous TDM, Modems, Transmission impairments, Manchester and differential Manchester encoding • **Error Detection and Correction:** Types of errors Redundancy, Detection Versus Correction, Error Detection, Error Correction, Hamming Code, Cyclic Redundancy Check, Check sum and Its idea.
Module 3: Data Link Layer and Medium Access Control (MAC) (1 Credit)	
Learning Outcomes	Learners will be able to: • Apply understanding of the role and functions of the Data Link Layer in the OSI model, as well as its relationship with the Physical Layer. • Analyze various framing techniques used in the Data Link Layer, including character count, flag bytes with byte stuffing, and bit stuffing. • Evaluate the efficiency and effectiveness of these framing techniques in ensuring accurate and reliable data transmission. • Design and implement framing algorithms for efficient data encapsulation and transmission across networks.
Module Contents	• **Data Link Layer Design issues:** Framing, error control, Flow Control, Error Detection and correction; Elementary Data Link Protocols, Sliding Windows Protocols; Medium Access Control: Aloha, CSMA protocols, Collision free protocols, Limited Contention Protocols; Wave length division Multiple access protocol, Wireless • **LAN Protocol:** MACA; IEEE 802.3Ethernet, IEEE 802.4 Token Bus; IEEE 802.5 Token ring, Binary Exponential Back off algorithm • **Digital Cellular, Radio :** Global System for Mobile, Communication (GSM), Code Division Multiple Access (CDMA)
Module 4: Network Layer and Routing Algorithms (1 Credit)	
Learning Outcomes	Learners will be able to: • Apply understanding of the responsibilities of the Network Layer in the OSI model and differentiate it from other layers. • Analyze and design efficient addressing schemes for packet routing, implementing them as per network requirements. • Evaluate the performance, advantages, and disadvantages of virtual circuit switching compared to other switching techniques. • Design network configurations using appropriate switching techniques to meet specific networking needs.
Module Contents	• Network Layer, Design issues Virtual circuit and Datagram Subnet, Routing Algorithms, Optimality principle, shortest path routing, Flooding, Distance Vector Routing, Link State Routing, Hierarchical Routing, Broadcast and Multi Cast Routing, Routing for Mobile hosts, Routing in Ad hoc Networks • Congestion Control Algorithm, General Principals Traffic Shaping, Leaky Bucket, Token Bucket, choke packets, Load Shedding

Assignments/ Activities towards Comprehensive Continuous Evaluation
Test students' understanding of fundamental concepts in data communications and networking. **Module 1:** • Activity 1: Write a report discussing the concept of computer networks and their significance in modern computing. Discuss the advantages of connecting computers in a networked environment and differentiate between common network topologies like bus, star, ring, mesh, and tree. • Activity 2: Choose a specific network topology and evaluate its strengths and weaknesses in a given scenario. Consider factors like scalability, fault tolerance, installation, and maintenance costs. **Module 2:** • Activity 1: Write a report differentiating between circuit switching and other switching techniques like packet switching and message switching. Discuss the process of establishing a circuit in a circuit-switched network and analyse the advantages and disadvantages of circuit switching. • Activity 2: Design a simple circuit-switched network using a common circuit switching protocol. Document and reflect on your process. **Module 3:** • Activity 1: Write a report on the role and functions of the Data Link Layer in the OSI model. Analyse various framing techniques used in the Data Link Layer and evaluate their efficiency and effectiveness. • Activity 2: Implement a simple framing algorithm for efficient data encapsulation and transmission across networks. Document and reflect on your process and any challenges you encountered. **Module 4:** • Activity 1: Write a report discussing the responsibilities of the Network Layer in the OSI model and differentiating it from other layers. Discuss efficient addressing schemes for packet routing. • Activity 2: Design a network configuration using an appropriate switching technique to meet a specific networking need. Reflect on the performance, advantages, and disadvantages of your chosen technique

Evaluation Scheme:

Internal Assessment:

Depending on the activities mentioned above a project should be developed of 50 marks. Internal Assessment to be done on the basis of Project Presentations and Report Writing with proper Rubrics (Given below).

The Rubric will have the following Evaluation Parameters:	
Evaluation Parameters	**Marks**
Functionality (Core Features)	20
Code Quality (Structure, Readability)	10
GUI Design & User Experience	10
Input Handling & Error Management	5
Report writing or Documentation	5

The project Internal assessment will be done by conducting project presentation at the department level, where External Examiner appointed by the department will be evaluating the project.

Two sample projects are given below and it is expected from students to choose a good and advance project topic that aligns with their interests and abilities to work on.

Project 1: Simulated LAN Design and Implementation Using Packet Switching and Routing Algorithms
Design and simulate a Local Area Network (LAN) using a packet-switched model. Implement routing algorithms like Dijkstra's shortest path or Distance Vector Routing to simulate how data packets move across the network efficiently.

Project 2: Wireless Network Simulation with CSMA/CA and GSM/CDMA Models
Simulate a wireless communication system for a campus network or smart city using CSMA/CA protocol and GSM/CDMA principles for mobile communication. Focus on medium access control and network setup in wireless environments.

External Assessment:

End Semester examination of 50 marks for 2 hours duration will be conducted by the University.

References: -

1. Forouzan, B. A. (2007). Data Communications and Networking (4th ed.). McGraw Hill.
2. Stallings, W. (2013). Data and Computer Communications (10th ed.). Pearson.
3. Tanenbaum, A. S. (2010). Computer Networks (5th ed.). Pearson.

Course Code	**4.3 (Minor Stream)**
Course Title	**Probability and Statistics**
Course Credits	**4**
Course Outcomes	**After completion of this Course, the students will be able to** • Understand fundamental concepts of probability theory and statistical inference. • Learn various probability distributions and their applications. • Develop skills in statistical estimation and hypothesis testing. • Apply probability and statistics in real-world problem-solving.
Module 1: Descriptive Statistics and Probability Fundamentals (1 Credit)	
Learning Outcomes	Learners will be able to: • Calculate and interpret measures of central tendency and dispersion. • Differentiate between different types of statistical data. • Apply probability rules and Bayes' theorem in problem-solving.
Module Contents	**Introduction to Statistics** – Population vs. Sample, Types of Data **Statistical Measures** • Central Tendency: Mean, Median, Mode • Dispersion: Variance, Standard Deviation, Range, IQR • Skewness and Kurtosis • Difference between Parametric and Nonparametric Tests **Basic Probability Theory** • Set Theory and Counting Techniques • Axioms of Probability • Conditional Probability and Bayes' Theorem
Module 2: Probability Distributions and Theoretical Foundations (1 Credit)	
Learning Outcomes	Learners will be able to: • Identify and analyze key probability distributions. • Compute probabilities, expectation, and variance of distributions. • Explain and apply the Central Limit Theorem.
Module Contents	• **Random Variables** – Discrete and Continuous • **Discrete Distributions** – Bernoulli, Binomial, Poisson • **Continuous Distributions** – Uniform, Normal, Exponential • **Properties of Distributions** – Expectation, Variance, Moment-Generating Functions • **Theoretical Concepts** – Law of Large Numbers, Central Limit Theorem
Module 3: Statistical Inference and Hypothesis Testing (1 Credit)	
Learning Outcomes	Learners will be able to: • Construct confidence intervals and estimate parameters. • Perform hypothesis tests and interpret results. • Apply statistical inference in real-world scenarios.
Module Contents	**Sampling Distributions and Estimation** • Point and Interval Estimation • Maximum Likelihood Estimation (MLE) • Confidence Intervals **Hypothesis Testing** • Steps in Hypothesis Testing • Z-test, T-test, Chi-square test, F-test

	• Type I and Type II Errors • P-values and Significance Levels
Module 4: Regression, Correlation (1 Credit)	
Learning Outcomes	Learners will be able to: • Perform correlation and regression analysis. • Apply nonparametric statistical techniques. • Understand Bayesian inference and real-world applications.
Module Contents	**Correlation and Regression** • Pearson and Spearman Correlation • Simple and Multiple Linear Regression • Coefficient of Determination (R^2)
Assignments/ Activities towards Comprehensive Continuous Evaluation	
Module 1: • Calculate mean, median, mode, variance, and standard deviation for a given dataset. • Compare the spread of two datasets using IQR and standard deviation. • Explain the differences between parametric and nonparametric tests with examples. • Solve problems on probability using set theory and counting principles. • Apply Bayes' Theorem to solve real-world conditional probability problems. **Module 2:** • For a given scenario, identify whether it follows a discrete or continuous distribution. • Compute probability mass function (PMF) and cumulative distribution function (CDF) for Bernoulli, Binomial, and Poisson distributions. • Calculate mean and variance for Uniform, Normal, and Exponential distributions. • Apply the Central Limit Theorem to approximate probabilities using the normal distribution. • Derive moment-generating functions (MGFs) for given probability distributions. **Module 3:** • Construct confidence intervals for population mean and proportion using real-world data. • Perform point and interval estimation using Maximum Likelihood Estimation (MLE). • Conduct hypothesis tests (Z-test, T-test, Chi-square test, F-test) and interpret results. • Explain Type I and Type II errors with examples. • Compute p-values and discuss their significance in hypothesis testing. **Module 4:** • Calculate Pearson and Spearman correlation coefficients for a given dataset. • Perform simple and multiple linear regression analysis and interpret regression coefficients. • Compute the coefficient of determination (R^2) and explain its significance. • Apply nonparametric statistical techniques to analyze real-world data. • Explain Bayesian inference with an example application in data science.	

Evaluation Scheme:

Internal Assessment:

Depending on the activities mentioned above a project should be developed of 50 marks. Internal Assessment to be done on the basis of Project Presentations and Report Writing with proper Rubrics (Given below).

The Rubric will have the following Evaluation Parameters:	
Evaluation Parameters	**Marks**
Functionality (Core Features)	20
Code Quality (Structure, Readability)	10

GUI Design & User Experience	10
Input Handling & Error Management	5
Report writing or Documentation	5

The project Internal assessment will be done by conducting project presentation at the department level, where External Examiner appointed by the department will be evaluating the project.

Two sample projects are given below and it is expected from students to choose a good and advance project topic that aligns with their interests and abilities to work on.

Project 1: Analysis of Student Performance Based on Study Hours
This project focuses on understanding how study hours impact the academic performance of students in a university setting. Students will analyze a dataset of student study hours and their final exam scores. The goal is to use probability and statistical techniques to identify patterns, estimate relationships, and test hypotheses regarding the effects of study time on performance.

Project 2: Analyzing the Effect of Advertising on Sales for a Retail Store
In this project, students will analyze the effect of advertising expenditure on sales performance for a retail store. They will use statistical methods to explore the relationship between the amount spent on advertising (in different media) and the sales revenue of the store.

External Assessment:

End Semester examination of 50 marks for 2 hours duration will be conducted by the University.

Reference Books:

1. **"Probability and Statistics for Engineers and Scientists"** – Ronald E. Walpole, Raymond H. Myers, Sharon L. Myers, Keying Ye
2. **"Introduction to Probability and Statistics"** – William Mendenhall, Robert J. Beaver, Barbara M. Beaver
3. **"Mathematical Statistics with Applications"** – Dennis Wackerly, William Mendenhall, Richard L. Scheaffer
4. **"A First Course in Probability"** – Sheldon Ross
5. **"Introduction to the Practice of Statistics"** – David S. Moore, George P. McCabe, Bruce Craig
6. **"The Elements of Statistical Learning"** – Trevor Hastie, Robert Tibshirani, Jerome Friedman
7. **"Bayesian Data Analysis"** – Andrew Gelman, John B. Carlin, Hal S. Stern, et al.
    a. Covers Bayesian statistics, useful for Module 4.
8. **"Statistical Inference"** – George Casella, Roger L. Berger
    a. A rigorous treatment of estimation, hypothesis testing, and statistical decision theory.

4.3OEC Courses to be taken from university basket (2 Credit)

4.4Swayam/ Chetna / MOOCs

4.5Ability Enhancement Courses (AEC) to be chosen from university basket (2credit)

Course Code	**4.7 (CE)**
Course Title	**Digital literacy and E-Waste Management**
Course Credits	**2**
Course Outcomes	**After completion of this Course, the students will be able to** • Create awareness to educate community members on digital literacy and online safety • Understand the concept, sources, and impact of electronic waste. • Learn about e-waste regulations, recycling techniques, and disposal methods. • Study real-world case studies on e-waste recycling and management.
Module 1: (1 Credit)	**Digital Literacy**
Learning Outcomes	**Learners will be able to:** • Understand the importance of digital literacy for community empowerment • Identify and describe e-governance services and their benefits.
Module Contents	•Introduction to digital literacy •Basic computer skills •Online safety and etiquette •Accessing government services online(e-governance)
Module 2: (1 Credit)	**Introduction E-Waste Management**
Learning Outcomes	**Learners will be able to:** • Define e-waste and identify its sources. • Analyze the environmental and health risks of improper e-waste disposal. • Explain e-waste generation trends and regulatory frameworks.
Module Contents	• Definition and Types of E-Waste • Sources and Categories of E-Waste • Environmental and Health Impacts of E-Waste • Global and Indian E-Waste Generation Trends • Overview of E-Waste Management Rules (India & Global) • Case Studies on Successful E-Waste Recycling Models
Assignments/ Activities towards Comprehensive Continuous Evaluation	
Module 1: • Identify common e-waste items and their impact. • Design posters, flyers, or a website to educate people. • Collect e-waste and ensure proper disposal through authorized recyclers. **Module 2:** • Collect old electronics (mobile parts, circuit boards, batteries). • Identify components that can be reused or refurbished. • Build a working model (e.g., turning old mobile batteries into power banks, LED lights from old circuits). • Document the process and create a sustainability report.	

Evaluation Scheme:

Internal Assessment:

Depending on the activities mentioned above a project should be developed of 50 marks. Internal Assessment to be done on the basis of Project Presentations and Report Writing with proper Rubrics (Given below).

The Rubric will have the following Evaluation Parameters:	
Evaluation Parameters	**Marks**
Functionality (Core Features)	20
Code Quality (Structure, Readability)	10
GUI Design & User Experience	10
Input Handling & Error Management	5
Report writing or Documentation	5

The project Internal assessment will be done by conducting project presentation at the department level, where External Examiner appointed by the department will be evaluating the project.

Two sample projects are given below and it is expected from students to choose a good and advance project topic that aligns with their interests and abilities to work on.

Project 1: Creating a Digital Literacy Awareness Campaign with a Focus on Online Safety and E-Governance

Design and implement an awareness campaign on digital literacy, emphasizing online safety, etiquette, and how to access e-governance services. The campaign will aim to educate the local community about the importance of digital literacy and the advantages of online government services.

Project 2: E-Waste Recycling and Repurposing Workshop

Organize a workshop on e-waste recycling, where students collect old electronics (like mobile phone parts, batteries, and circuit boards) and repurpose them into useful items (such as power banks, LED lights, or simple electronics projects). The project will also educate participants about e-waste management principles and sustainability.

References:

1. Digital literacy and cyber security resources (e.g., videos, infographics, etc.)
2. E-governance service guides and tutorials
3. E-Waste Management: From Waste to Resource – Klaus Hieronymi, Ramzy Kahhat, Eric Williams
4. Electronic Waste Management – Rakesh Johri
5. Waste Electrical and Electronic Equipment (WEEE) Handbook – Vannessa Goodship, Ab Stevels

4.8 Co-curriculum course-IV (CC) to be taken from the University Basket (2 Credits)