Krantiguru Shyamji Krishna Verma Kachchh University

CURRICULAM AND CREDIT FRAMEWORK FOR

M.Sc (CA&IT) 3 YEARS AND 4 YEARS PROGRAMMES

Semester 5 and 6

(Effective from 2025-26)

AS PER THE NEP 2020

	M. Sc. (CA&IT) Course Outline						
Semester	**Course Type**	**Course Code**	**Name of the Subject**	**Theory/ Practical**	**Marks**		**Credits**
					IA	**UA**	
5	Major	CAIT-501	Advanced Web Development using PHP	Theory	50	50	4
	Major	CAIT -502	Advanced Database Management System	Theory	50	50	4
	Major	CAIT -503P	LAB : Practical based on CAIT501 and CAIT502	Practical	50	50	4
	Minor	CAIT -504	Software Engineering and Project Management	Theory	50	50	4
	Minor	CAIT -505	Data Warehousing	Theory	50	50	4
	Skill Enhancement Courses(SEC)	CAIT -506P	Practical Applications of IoT	Practical	25	25	2
					275	**275**	**22**
6	Major	CAIT-601	Mobile Computing	Theory	50	50	4
	Major	CAIT-602	Data mining and visualization	Theory	50	50	4
	Major	CAIT-603P	LAB : Practical based on CAIT601 and CAIT602	Practical	50	50	4
	Minor	CAIT-604	Digital Forensics and Cyber Law	Theory	50	50	4
	Ability Enhancement Courses	CAIT-605	The Startup Sprint	Theory	25	25	2
	Skill Enhancement Course	CAIT-606P	Project	Project / Viva	50	50	4
					275	***275***	***22***
	External Exam Hours: 2 Hrs – 4 Credit Course – 50 Marks *1 Hr – 2 Credit Course – 25 Marks* *Passing Marks: 35%*						

External evaluation will be based on Semester End Evaluation (SEE) pattern.

M.Sc. (CA&IT) – 3 Years and 4 Years Programme Structure of the University or External Exam for 4 Credit Course		
Q-1 All Units	Objective Questions (It can include: definitions, FIBs, True or false, one line answers, MCQs etc)	10
Q-2 (Unit -1)	Answer two short questions carrying 2 marks respectively (Compulsory) Answer two questions, Short notes carrying 3 marks respectively (2 out of 3)	10
Q-3 (Unit -2)	Answer two short questions carrying 2 marks respectively (Compulsory) Answer two questions, Short notes carrying 3 marks respectively (2 out of 3)	10
Q-4 (Unit -3)	Answer two short questions carrying 5 marks respectively OR Any one question which could be a long question, case study, application of concepts, practical problem etc carrying 10 marks	10
Q-5 (Unit -4)	Answer two short questions carrying 5 marks respectively OR Any one question which could be a long question, case study, application of concepts, practical problem etc carrying 10 marks	10

MSc (CA&IT) – 3 Years and 4 Years Programme
Structure of the University or External Exam for 2 Credit Course**

Q-1 All Units	Objective Questions (It can include: definitions, FIBs, True or false, one line answers, MCQs etc)	05
Q-2 (Unit -1)	Answer two short questions carrying 2 marks respectively (Compulsory) Answer two questions, Short notes carrying 3 marks respectively (2 out of 3)	10
Q-3 (Unit -2)	Answer two short questions carrying 2 marks respectively (Compulsory) Answer two questions, Short notes carrying 3 marks respectively (2 out of 3)	10

Note - University examination will be of 25 Marks and 1Hr

** *For VAC and AEC structure can vary according to subject criteria*

Structure of the University or External Practical Exam for 2 Credit Course		
Sr.No	**Contents**	**Marks**
1.	Practical	15
2.	Viva	10
Total		**25**

Structure of the University or External Practical Exam for 4 Credit Course		
Sr.No	**Contents**	**Marks**
1.	Practical	30
2.	Viva	20
Total		**50**

MSC (CA&IT) - Semester: V
(Effective from year 2025-26)

Course Code:	**CAIT-501**		**Course Title:**	**Advanced Web Development using PHP**
Course Credits:	**04**		**Hour of Teaching/Week:**	**04**
Internal Assessment Marks:	**50**		**External Exam Marks:**	**50**
Exam Duration	**2 Hrs**			

- CO1 : Students will gain a strong understanding of PHP syntax, data types, control structures, and functions to build basic web applications.
- CO2: Students will learn to create dynamic web pages using HTML forms, session management, and cookies for improved user experience.
- CO3: Students will apply OOP principles to design and develop structured and maintainable PHP applications.
- CO4: Students will connect PHP applications to MySQL databases, execute CRUD operations, and implement secure database interactions to prevent vulnerabilities.
- CO5: Students will utilize AJAX to build dynamic user interfaces and complete a practical web application project demonstrating comprehensive PHP development skills.

Unit	Contents
1	**Foundations of Web and PHP** - Internet Basics: HTTP and HTTPS Protocol, Web Servers, Web Hosting, Virtual Host, Multi-Homing - Website Types: Static vs. Dynamic - Scripting: Client-side vs. Server-side - PHP Installation and Configuration (XAMPP/WAMP/LAMP) - PHP Basics: Syntax, Tags, Comments, Document Root folder - PHP.INI & .htaccess Overview - Variables, Data Types, Constants - Operators, Conditional Statements - Looping Structures: for, while, do-while, foreach
2	**Functional Programming and Data Handling** - Functions: Built-in & User-defined - Parameters, Default & Variable-length arguments - Scope: Local, Global, Static, Superglobals - Arrays: Indexed, Associative, Multidimensional - Array Functions: sort(), explode(), array_merge(), etc. - Strings: String manipulation and handling functions - File Handling: fopen(), fread(), fwrite(), file_exists(), unlink()

3	**Forms, Sessions, OOP in PHP** - HTML Forms Integration - Form Validation: isset(), empty(), filter_input() - GET vs POST methods - File upload: File uploading functions. - Cookies: setcookie(), $_COOKIE - Sessions: session_start(), session_destroy(), $_SESSION - OOP Concepts: Class, Object, Constructor, Destructor - Inheritance, Overriding, self and this keywords - Exception Handling: try, catch, throw, finally
4	**Database Connectivity, AJAX, and Project** - MySQL Integration - Connecting PHP to MySQL: MySQLi - Performing CRUD Operations - Preventing SQL Injection with Prepared Statements - AJAX Introduction: XMLHttpRequest, jQuery AJAX - Working with JSON - Building Dynamic UI with AJAX and PHP - Mini Project: A functional CRUD web application integrating forms, OOP, MySQL, and AJAX

Text Books & References

Begging PHP 5 by Wrox. 2.
Julie C. Meloni, PHP MySQL and Apache, SAMS Teach Yourself, Pearson Education.
Web Development using PHP –Bharat & Co. [ISBN No. : 978-93-81786-39-0]
PHP Manual – https://www.php.net/manual

MSC (CA&IT) - Semester: V

(Effective from year 2025-26)

Course Code:	**CAIT 502**		**Course Title:**	**Advanced Database Management System**
Course Credits:	**4**		**Hour of Teaching/Week:**	**4**
Internal Assessment Marks:	**50**		**External Exam Marks:**	**50**
Exam Duration	**2 Hours**			

- CO1: Understand and apply fundamental concepts of relational databases, including SQL for data definition, manipulation, control, and transaction management.
- CO2: Develop and implement procedural extensions using PL/SQL, including control structures, cursors, stored procedures, functions, and triggers for complex database operations.
- C03: Analyze the architecture and challenges of distributed database systems, including reliability protocols and data distribution strategies.
- CO4: Comprehend the principles and motivations behind NoSQL databases, understanding the CAP theorem and different data models like key-value, columnar, graph, and document stores.
- CO5: Gain practical experience with MongoDB, including installation, basic shell commands, CRUD operations, and the aggregation framework for managing NoSQL data.

Unit	**Contents**
1	**Relational and Procedural Database Concepts** Introduction to Relational Databases and SQL The relational database model: Entities, attributes, relationships , Structured Query Language (SQL) as a declarative language Data Control Language (DCL): Access control and security , Transaction Control Language (TCL): Managing database transactions. **Procedural Extensions and PL/SQL - I** The need for procedural extensions in SQL , Introduction to PL/SQL: Architecture and execution environment , Function-oriented vs. procedure-oriented programming in the context of databases , Advantages of using PL/SQL for complex database operations , Basic PL/SQL block structure
2	**Procedural Extensions and PL/SQL II** Control Structures , Conditional statements: (IF, ELSIF, ELSE) , Looping constructs(LOOP, WHILE LOOP, FOR LOOP), Nested loops and conditional statements , Cursors , implicit and explicit cursors ,Declaring, opening, fetching, and closing cursors ,Using %ROWTYPE and %FOUND, %NOTFOUND, %ROWCOUNT attributes , Parameterized cursors,For Loops with Cursors , Stored Procedures and Functions , Creating and executing stored procedures, Passing parameters to procedures (IN, OUT, IN OUT) , Creating and using functions , Returning values from functions, Advantages of stored procedures and functions, Understanding database triggers, Types of triggers (BEFORE, AFTER, INSTEAD OF).

3	**Distributed Database Systems Theory I** Distributed database architectures , Reliability and commit protocols (2PC, Paxos, Raft) , Data fragmentation and distribution strategies , Distributed database design principles , Distributed algorithms for data management , Heterogeneous and federated database systems. **Non-Relational Databases and Data Modeling** **Introduction to NoSQL Databases** The rise of NoSQL databases and their motivation, Advantages and disadvantages of NoSQL compared to relational databases, The CAP theorem and its implications for distributed data management, Understanding ACID properties
4	**Types of NoSQL Databases and Data Models** Key-value databases: Concepts and use cases , Columnar databases: Concepts and use cases , Graph databases: Concepts and use cases , Document databases: Concepts and use cases , Comparative analysis of RDBMS and NoSQL, with use cases. **Introduction to NoSQL MongoDB, its Basics and CRUD Operations** Installation and Setup, Basic shell commands (show dbs, use <database>, show collections) ,Inserting Documents ,Practice inserting various data types (strings, numbers, dates, arrays, embedded documents),Querying Documents, Use query operators ($eq, $ne, $gt, $lt, $in, $nin),Use logical operators ($and, $or, $not),Implement sorting (sort()), limiting (limit()), and skipping (skip()).,Updating Documents, Deleting Documents ,Aggregation Framework

Text Books and References:

1. Fundamentals of Database Systems (3 edition), Elmasri R. and Navathe S.B., 2000,Addison Wesley, Low Priced Edition
2. An Introduction to Database System by Bipin Desai

MSC (CA&IT) - Semester: V
(Effective from year 2025-26)

Course Code:	**CAIT 503 P**		**Course Title:**	**LAB : Practical based on CAIT501 and CAIT502**
Course Credits:	**4**		**Hour of Teaching/Week:**	**4**
Internal Assessment Marks:	**50**		**External Exam Marks:**	**50**
Exam Duration				

Unit	**Contents**
	Practical based on CAIT501 a. Install and configure PHP , webserver , MYSQL b. Write a program to print "Welcome toPHP". c. Write a simple PHP program using expressions and operators. d. Write a PHP program to demonstrate the use of Decision making control structures using- i. If statement ii. If-else statement iii. Switch statement e. Write a PHP program to demonstrate the use of Looping structures using- a. While statement, b. Do-while statement c. For statement d. For each statement f. Write a PHP program for creating and manipulating- i. Indexed array ii. Associative array iii. Multi dimensional array g. Write a simple PHP program to demonstrate use of Simple function and Parameterized function. h. Design a web page using following form controls: Text box, Radio button, Check box, Buttons, List box, Combo box, Hidden field box i. Write simple PHP program to- i. Set cookies and read it. ii. Demonstrate session Management. iii. Develop a simple application to- iv. Enter data into database v. Retrieve and present data from database. **vi.** Develop a simple application to Update , Delete table data from database.

1	j. Develop an application that will dealing with the file handling function. k. AJAX with Database, File and JSON. l. An Introduction to LARAVEL Framework I. Introduction II. Laravel – Features.. III. LARAVEL – INSTALLATION IV. LARAVEL – CONFIGURATION V. LARAVEL – ROUTING VI. LARAVEL – CONTROLLERS . VII. LARAVEL — REQUEST VIII. LARAVEL – COOKIE IX. LARAVEL — RESPONSE. X. LARAVEL — VIEWS . XI. LARAVEL — REDIRECTIONS XII. LARAVEL — WORKING WITH DATABASE
	DCL & TCL ● User and Role Management: ○ Create new database users with specific usernames and passwords. ○ Grant and revoke system privileges (e.g., CREATE TABLE, CREATE VIEW) to users. ○ Create database roles and assign privileges to roles. ○ Grant and revoke object privileges (e.g., SELECT, INSERT, UPDATE, DELETE) on tables and views to users and roles. ○ Demonstrate how to view granted privileges for users and roles. ● Basic Transaction Management: ○ Write PL/SQL blocks or SQL scripts that include INSERT, UPDATE, and DELETE statements. ○ Use COMMIT and ROLLBACK statements to control transactions. ○ Demonstrate the effects of COMMIT and ROLLBACK on data changes. Practice using SAVEPOINTs to mark specific points within a transaction.

PL/SQL Fundamentals

- Basic PL/SQL Blocks:
- Write PL/SQL blocks to declare variables, assign values, and display output using DBMS_OUTPUT.PUT_LINE.
    - Create nested PL/SQL blocks to demonstrate variable scope.
    - Practice using different data types (NUMBER, VARCHAR2, DATE, BOOLEAN).
- Control Structures:
    - Implement IF-THEN-ELSIF-ELSE statements to perform conditional logic.
    - Create CASE statements for multiple condition handling.
    - Write programs using basic LOOP, WHILE LOOP, and FOR LOOP.
    - Demonstrate the use of EXIT and CONTINUE statements.
- Cursors
    - Write PL/SQL blocks using explicit cursors to fetch data from tables.
    - Use cursor attributes (%FOUND, %NOTFOUND, %ROWCOUNT, %ISOPEN) to control cursor behavior.
    - Create parameterized cursors to fetch data based on input parameters.
- Exception Handling
    - Write PL/SQL blocks to handle predefined exceptions (e.g., NO_DATA_FOUND, TOO_MANY_ROWS).
    - Create user-defined exceptions and raise them using the RAISE statement.
    - Use the EXCEPTION section to handle exceptions and display error messages.
- Procedures and Functions
    - Create procedures with IN, OUT, and IN OUT parameters.
    - Write functions that return values.
    - Call procedures and functions from PL/SQL blocks and SQL statements.
- Triggers
    - Create BEFORE and AFTER triggers for INSERT, UPDATE, and DELETE operations.
    - Write row-level and statement-level triggers.
    - Use :NEW and :OLD qualifiers to access data in triggers.

2

MongoDB Basics and CRUD Operations

MSC (CA&IT) - Semester: V
(Effective from year 2025-26)

Course Code:	**CAIT 504**		**Course Title:**	**Software Engineering and Project Management**
Course Credits:	**4**		**Hour of Teaching/Week:**	**4**
Internal Assessment Marks:	**50**		**External Exam Marks:**	**50**
Exam Duration	**2 hours**			

- CO1: Students will gain a comprehensive understanding of the SDLC, including various models and their practical applications.
- CO2: Students will become proficient in eliciting, analyzing, and documenting software requirements, including the ability to create Software Requirements Specifications (SRS) and use UML diagrams for representation.
- CO3: Students will develop skills in software design principles, architectural design, and project planning, enabling them to create software architectures, project plans, and manage project-related aspects like timelines, budgets, and risk assessments.
- CO4: Students will acquire a strong foundation in software testing methodologies and techniques, including black-box and white-box testing, test case design, test management, and an introduction to automated testing.
- CO5: Students will learn the principles of dissertation writing, including structure, style, and ethical considerations, and will develop the ability to effectively document software engineering projects.

Unit	**Description**
I	**Introduction to Software Engineering and Requirement specification:** Definition, Importance, and Ethics**.** Software Process: SDLC, SDLC Models. Introduction to Project Proposals, Types of project proposals (e.g., research, business, development). Requirements Elicitation: Techniques for Gathering User Needs. Requirements Analysis: Use Cases(UML), Functional/Non-Functional Requirements, SRS. Literature Review. **Practical:** Give Project Idea Summary, Initial Literature Review, Draft SRS. Project proposal introduction, objectives and scope.
II	**Software Design and Software Project Planning:** Software Design, Principles & Architecture. Design Concepts: Coupling, Cohesion. Project methodology, Introduction to UML: Class Diagram, Component Diagram, Use case Diagram, Activity Diagram, Introduction to DFD .Introduction to data dictionary, ER Diagram. Project Planning: Timeline, Budget, Risk Assessment, cost estimation: COCOMO Model. **Practical:** Prepare Project Architecture Diagram, Project Plan, Draft Methodology, Timeline and Budget sections of proposal.

III	**Software Testing Fundamentals:** Unit testing, Integration testing,System.Testing.Software Testing Techniques: Black-box, White-box. Formal Verification. Defect Tracking and Management. Introduction to Automation Testing. Introduction to Testing tools for web application and mobile application. Emerging Trends in Software Testing. Validation vs. Verification. Evaluation Metrics and Criteria. **Practical:** Case studies on real-world testing scenarios. Creating simple test plans and test cases for sample applications.
IV	**Dissertation Documentation and Project Status:** Introduction to Dissertation Writing: Structure, Style, and Format. Types of Dissertations. Dissertation structure and ethics. Peer Review and Feedback. Referencing and Citation Techniques. Draft Dissertation Chapters (Requirements, Design, Implementation, Testing, Results, Conclusion). **Practical:** Create one sample dissertation Document.

Basic Text & Reference Books :-	
1.	Roger Pressman, ―Software Engineering: A Practitioner‘s Approach‖, McGraw Hill, ISBN 0– 07–337597–7
2.	Rajib Mall, ―Fundamentals of Software Engineering‖, Prentice Hall India, ISBN-13: 978-8120348981
3.	"The Craft of Research" by Wayne C. Booth, Gregory G. Colomb, and Joseph M. Williams

MSC (CA&IT) - Semester: V
(Effective from year 2025-26)

Course Code:	CAIT-505		Course Title:	Data Warehousing
Course Credits:	4		Hour of Teaching/Week:	4
Internal Assessment Marks:	50		External Exam Marks:	50
Exam Duration	2 Hours			

- CO1 : Understand the fundamentals of mobile application development, including the Android platform and the Dart programming language.
- CO2 : Develop user interfaces using Flutter widgets, manage user interactions, implement navigation, and apply styling and themes for visually appealing apps.
- CO3 : Apply different state management techniques in Flutter and integrate Firebase services, specifically Firestore, for data storage and real-time updates.
- CO4 : Implement advanced UI features, including custom animations, gesture recognition, and local data storage in Flutter applications.
- CO5 : Gain knowledge of the process for building and deploying Flutter applications on Android platforms and understand the basics of app store deployment.

Unit	Contents
1	**Introduction to Data Warehousing** Introduction: What is a Data Warehouse? Need for Data Warehousing, Operational Database Systems vs. Data Warehouses, Data Warehouse Characteristics (Subject-Oriented, Integrated, Time-Variant, Non-Volatile). • Data Warehouse Architecture: Components of a Data Warehouse (Source Systems, Data Staging Area, Data Warehouse Database, Front-end Tools), Three-Tier Architecture, Data Warehouse Models (Enterprise Warehouse, Data Mart, Virtual Warehouse). • Data Warehouse Development Life Cycle: Planning, Requirements Gathering, Design (Conceptual, Logical, Physical), Implementation, Deployment, Growth and Maintenance. • Data Warehousing Applications: Business Intelligence, Decision Support Systems, Competitive Advantage, Case Studies.

2	**Dimensional Modeling** • Concepts of Dimensions, Facts, Measures. • Star Schema: Structure, Advantages, Disadvantages, Examples. • Snowflake Schema: Structure, Advantages, Disadvantages, Comparison with Star Schema, Examples. • Fact Table Design: Additive, Semi-Additive, Non-Additive Facts, Types of Fact Tables (Transactional, Snapshot, Accumulating Snapshot). • Dimension Table Design: Slowly Changing Dimensions (SCDs) - Type 1, Type 2 (with history), Type 3 (current and previous), Type 4 (history table), Type 6 (hybrid), and Surrogate Keys. • Data Granularity: Levels of Granularity, Impact on Design. • Conformed Dimensions and Facts.
3	**Data Extraction, Transformation, and Loading (ETL)** • ETL Overview: Importance of ETL, ETL Process Steps (Extraction, Transformation, Loading). • Data Extraction: Sources of Data (Operational Databases, External Sources, Legacy Systems), Data Extraction Techniques (Full Extraction, Incremental Extraction). • Data Transformation: Data Cleaning (Handling Missing Values, Noise Reduction, Data Type Conversion), Data Integration (Schema Integration, Data Aggregation), Data Transformation Techniques (Normalization, Discretization , Attribute Construction, Data Filtering). • Data Loading: Loading into Data Warehouse, Initial Load vs. Incremental Load, Refresh Strategies (Full Refresh, Incremental Refresh), Data Quality Checks during Loading.

4	**Online Analytical Processing (OLAP)**
	• OLAP Concepts: Multidimensional Data Model, OLAP Operations (Drill-down, Roll-up, Slice, Dice, Pivot).
	• Types of OLAP Systems:
	  ○ Multidimensional OLAP (MOLAP): Architecture, Advantages, Disadvantages.
	  ○ Relational OLAP (ROLAP): Architecture, Advantages, Disadvantages.
	  ○ Hybrid OLAP (HOLAP): Architecture, Advantages, Disadvantages.
	• OLAP Architectures and Tools.

Text Books & References

 Ralph Kimball and Margy Ross, "The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling," Wiley.

 W.H. Inmon, "Building the Data Warehouse," Wiley.

 Paulraj Ponniah, "Data Warehousing Fundamentals: A Comprehensive Guide for IT Professionals," Wiley.

 Sam Anahory and Dennis Murray, "Data Warehousing in the Real World: A Practical Guide for Building Decision Support Systems," Addison-Wesley.

 Bill Inmon and Claudia Imhoff, "Tapping into the Data Warehouse," Wiley.

MSC (CA&IT) - Semester: V

(Effective from year 2025-26)

Course Code:	**CAIT 506P**		**Course Title:**	**Practical Applications of IoT**
Course Credits:	**2**		**Hour of Teaching/Week:**	**4**
Internal Assessment Marks:	**25**		**External Exam Marks:**	**25**
Exam Duration	**1 Hours**			

Practical Based on the Following Topics

1. Introduction to Arduino and the Development Environment:

- Introduction to Arduino : Concepts and Applications
- Arduino Simulation Environment and IDE Overview
- Setting up the Arduino Board
- Arduino Sketch Structure
- Uploading and Running Sketches
- Creating and Saving Sketches

2. Arduino Programming Fundamentals:

- Primitive Data Types (int, float, char, etc.)
- Variables and Constants
- Operators (Arithmetic, Logical, Comparison)
- Control Flow:
    - Conditional Statements (if, else, switch)
    - Looping Structures (for, while, do-while)
- Functional Blocks (Functions)

3. Basic Input / Output and Communication:

- LED Blinking and Control using Logical Code
- Working with the Serial Monitor:
    - Introduction to Serial Communication
    - Debugging and Data Output
- Introduction to Analog Communication (Analog Read , Analog Write)

4. Working with Basic Input Components:

- Push Button: Digital Input
- LDR Sensor: Analog Input (Light Detection)
- Rotary Sensor: Analog Input (Potentiometer)
- Joystick: Analog Input (X, Y axes)

5. Working with Output and Actuators:

- Buzzer: Digital Output (Sound)
- RGB LED: Digital and Analog Output (Colour Control)
- Servo Motor: Position Control
- DC Motor: Speed and Direction Control
- Relay: Switching Higher Voltage Devices

6. Advanced Sensors and Applications:

- Ultrasonic Sensor: Distance Measurement
- IR Sensor: Object Detection
- Flame Detector: Flame Detection
- Sound Detector: Sound Level Detection
- Temperature and Humidity Sensor: Environmental Monitoring
- Smoke Detector: Air Quality Monitoring

MSC (CA&IT) - Semester: VI

(Effective from year 2025-26)

Course Code:	**CAIT-601**		**Course Title:**	**Mobile Computing**
Course Credits:	**4**		**Hour of Teaching/Week:**	**4**
Internal Assessment Marks:	**50**		**External Exam Marks:**	**50**
Exam Duration	**2 Hours**			

- CO1 : Understand the fundamentals of mobile application development, including the Android platform and the Dart programming language.
- CO2 : Develop user interfaces using Flutter widgets, manage user interactions, implement navigation, and apply styling and themes for visually appealing apps.
- CO3 : Apply different state management techniques in Flutter and integrate Firebase services, specifically Firestore, for data storage and real-time updates.
- CO4 : Implement advanced UI features, including custom animations, gesture recognition, and local data storage in Flutter applications.
- CO5 : Gain knowledge of the process for building and deploying Flutter applications on Android platforms and understand the basics of app store deployment.

Unit	Contents
1	**Introduction To Mobile Apps:** Why We Need Mobile Apps, Different Kinds of Mobile Apps, Briefly about Android , Introduction Android: History Behind Android Development, What is Android?, Prerequisites to learn Android **Android Architecture:** Overview of Android Stack, Android Features, Introduction to OS layers , Introduction to AVD **Dart Programming Fundamentals:** Dart syntax and data types, Variables, operators, and control flow , Functions and classes , Asynchronous programming (Futures, async/await) , Object-oriented programming in Dart.
2	**Flutter Widgets Basics** Understanding the concept of widgets , Stateless and stateful widgets , Basic layout widgets (Container, Row, Column, Stack) , Text, Image, and Icon widgets , Building simple UI layouts. **Building Interactive UI** Handling user input , Gestures and touch events , Form validation ,Using lists and grids,Navigation concepts (routes, pages) , Passing data between screens , Named routes and route arguments. **Styling and Themes** Applying styles to widgets ,Creating custom themes , Using Material Design and Cupertino styles , Adapting UI for different screen sizes.

3	**State Management Techniques** Understanding application state and its importance , setState() for simple state management , Introduction to state management patterns (Provider, Bloc, Riverpod - choose one or two) , Exploring reactive programming concepts. **Firebase Overview:** What is Firebase? Understanding its services and benefits , Setting up a Firebase project.,Firebase console overview **Firebase Firestore** Firestore Basics (Understanding NoSQL databases and Firestore's data model , Collections and documents , Reading and writing data to Firestore , Real-time data updates , Firestore Queries , Filtering and sorting data) using Flutter
4	**Advanced UI and Animations:** Creating custom animations and transitions,Implementing gesture recognition, Using advanced layout techniques,Working with Canvas , Local data storage using SharedPreferences **Deployment:** Building and deploying Flutter applications for Android (APK, App Bundle),Introduction to app store deployment (Google Play Store, Apple App Store).

Text Books and References:

1. Google Flutter Mobile Development Quick Start Guide: Get up and running with iOS and Android mobile app development Paperback – 30 March 2019
   by Prajyot Mainkar (Author), Salvatore Giordano (Author)
2. Beginning App Development with Flutter. by Rap Payne

MSC (CA&IT) - Semester: VI

(Effective from year 2025-26)

Course Code:	CAIT-602		Course Title:	Data mining and Data Visualization
Course Credits:	4		Hour of Teaching/Week:	4
Internal Assessment Marks:	50		External Exam Marks:	50
Exam Duration	2 Hours			

- **CO1:** Explain the fundamental concepts and processes of data mining and visual data exploration.
- **CO2:** Apply basic data mining techniques such as classification, clustering, and association rule mining, and visualize their initial data and results using fundamental visualization methods.
- **CO3:** Select and interpret appropriate visualization techniques for different data structures, including hierarchical, network, and high-dimensional data.
- **CO4:** Understand the importance of interactive visualization for gaining deeper insights from data mining outcomes and apply basic interaction techniques.
- **CO5:** Identify and describe various data mining and visualization tools, differentiating between integrated environments and specialized visualization platforms.

Unit	Contents
1	**Foundations of Data Mining and Visual Data Exploration** • **Introduction to Data Mining:** What is Data Mining? Knowledge Discovery in Databases (KDD), Data Mining Process, Data Mining Tasks (Classification, Clustering, Association Rule Mining - overview). • **Introduction to Data Visualization:** Importance of Visualization in Data Analysis and Data Mining, History, Principles of Effective Visualization (Clarity, Accuracy, Efficiency, Aesthetics), the Visualization Process. • **Types of Data for Mining and Visualization:** Categorical, Numerical, Ordinal, And Temporal. • **Visual Data Exploration:** Using basic plots (histograms, scatter plots, box plots) to understand data distributions, identify outliers, and discover initial patterns. • **Data Preprocessing for Mining and Visualization:** Data Cleaning, Data Integration, Data Reduction, Data Transformation (brief overview).

2	**Basic Data Mining & Visualization** • **Classification:** Concepts (supervised learning), algorithms (Decision Trees, Naive Bayes - conceptual), visualizing data and basic results (feature/class distributions, confusion matrix intro, decision boundary concept). • **Clustering:** Concepts (unsupervised learning), algorithm (K-Means - conceptual), visualizing data (scatter plots), visualizing basic results (scatter with clusters, basic profiles). • Association **Rule Mining :** Concepts (market basket), algorithm (Apriori - basic idea), visualizing data (item frequencies), visualizing basic rules (support/confidence scatter plot intro).
3	• **Hierarchical Data:** Tree-like data (e.g., org charts) can be visualized using node-link diagrams (indented lists, radial trees), which are clear for small hierarchies but can become cluttered. • **Network Data:** Relationship-based data (e.g., social networks) is visualized with node-link diagrams (nodes and edges), where layout algorithms and visual encoding enhance readability. Adjacency matrices offer an alternative matrix-based representation. • **High-Dimensional Data:** Visualizing many variables is challenging. Scatter plot matrices (SPLOMs) display pair wise relationships but can be overwhelming with numerous dimensions. • **Interactive Visualization:** Interaction (filtering, zooming/panning) is crucial for dynamically exploring data mining results and gaining deeper insights.

4	**Overview of Data Mining and Visualization Tools**
	• **Introduction to Integrated Environments:** Discuss platforms that offer both data mining and visualization capabilities within a single interface. Examples include:
	  o **KNIME:** Open-source data analytics platform with a visual workflow-based approach for data mining, ETL, and basic visualization.
	  o **RapidMiner:** Another open-source platform offering a comprehensive suite of data science tools, including data mining and visualization.
	  o **SAS Enterprise Miner/Visual Analytics:** Commercial tools used in industry for advanced analytics and visualization. (Focus on conceptual understanding if access is limited).
	• **Introduction to Specialized Visualization Tools:** Discuss tools primarily focused on creating interactive and insightful visualizations:
	  o **Tableau:** Popular business intelligence tool known for its ease of use in creating a wide range of interactive charts and dashboards.
	  o **Power BI:** Microsoft's business analytics service for creating interactive visualizations and business intelligence capabilities

Text Books & References

- "Data Mining and Data Visualization" by Richard S. Gallagher (Elsevier)
- "Information Visualization: Perception for Design" by Colin Ware (Morgan Kaufmann)
- "Data Mining: Practical Machine Learning Tools and Techniques" by Ian H. Witten, Eibe Frank, Mark A. Hall, and Christopher J. Pal (Morgan Kaufmann)
- Sam Anahory and Dennis Murray, "Data Warehousing in the Real World: A Practical Guide for Building Decision Support Systems," Addison-Wesley.
- Bill Inmon and Claudia Imhoff, "Tapping into the Data Warehouse," Wiley.

MSC (CA&IT) - Semester: VI

(Effective from year 2025-26)

Course Code:	**CAIT-603P**		**Course Title:**	**Lab: Practical based on CAIT-601 and CAIT-602**
Course Credits:	**4**		**Hour of Teaching/Week:**	**4**
Internal Assessment Marks:	**50**		**External Exam Marks:**	**50**
Exam Duration	**2 Hours**			

Unit	**Content**
1	1. **Exploring Different Kinds of Mobile Apps:** Research and present examples of Native, Hybrid, and Web Apps, highlighting their differences and use cases. 2. **Setting up the Android Development Environment:** Install Android Studio and SDK (if focusing on the Android-specific introduction). 3. **Creating and Running a Simple "Hello World" Android App:** Familiarize yourself with the basic project structure in Android Studio (if applicable). 4. **Introduction to Android Virtual Device (AVD):** Create and configure an AVD for testing mobile apps. 5. **Dart Syntax and Data Types:** Write simple Dart programs to practice declaring variables of different data types (int, double, String, bool, List, Map, Set). 6. **Operators and Control Flow:** Implement programs using arithmetic, logical, relational operators, and control flow statements (if, else, for, while, do-while). 7. **Functions and Classes:** Define and use functions with different parameter types. Create and instantiate classes with properties and methods. 8. **Asynchronous Programming:** Write Dart code using Future to simulate asynchronous operations and use async/await to handle results. 9. **Object-Oriented Programming:** Implement inheritance, polymorphism, and understand the concepts of abstract classes and interfaces in Dart. 10. **Understanding Stateless Widgets:** Create simple UIs using StatelessWidget to display static information (e.g., a greeting, a fixed image). 11. **Understanding Stateful Widgets:** Create interactive UIs using StatefulWidget that can change over time (e.g., a counter app). 12. **Basic Layout Widgets:** Build layouts using Container for styling and positioning, Row and Column for arranging widgets linearly, and Stack for overlapping widgets. 13. **Text, Image, and Icon Widgets:** Display text with different styles, load and display images from assets and network, and use built-in and custom icons. 14. **Building Simple UI Layouts:** Combine basic layout widgets and content

	widgets to create simple screen designs (e.g., a profile screen, a basic list). 15. **Handling User Input:** Implement button presses and update the UI in response using onPressed callbacks. 16. **Gestures and Touch Events:** Detect and respond to different gestures like taps, double-taps, swipes using GestureDetector. 17. **Form Validation:** Create simple forms using TextFormField and implement basic validation rules. 18. **Using Lists and Grids:** Display dynamic data using ListView (for scrolling lists) and GridView (for displaying items in a grid). 19. **Navigation Concepts (Routes, Pages):** Implement basic navigation between different screens using Navigator.push and Navigator.pop. 20. **Passing Data Between Screens:** Pass simple data between different routes during navigation. 21. **Named Routes and Route Arguments:** Implement navigation using named routes and pass arguments to the destination screen. 22. **Applying Styles to Widgets:** Apply inline styles and use TextStyle and other style-related properties to customize the appearance of widgets. 23. **Creating Custom Themes:** Define a custom ThemeData to apply consistent styling across the application. 24. **Using Material Design and Cupertino Styles:** Build UIs that adhere to Material Design (Android-like) and Cupertino (iOS-like) visual guidelines using provided widgets and themes. 25. **Adapting UI for Different Screen Sizes:** Explore basic techniques for creating responsive layouts using MediaQuery or layout widgets like Expanded and Flexible. 26. **Simple State Management with setState():** Implement simple state changes and UI updates using setState() in StatefulWidget. 27. **Introduction to State Management (Choose one or two):** o **Provider:** Implement a simple counter or data sharing mechanism using the provider package. o **Bloc/Cubit:** Implement a basic feature with state management using the flutter_bloc package. o **Riverpod:** Implement a simple state management scenario using the flutter_riverpod package. 28. **Exploring Reactive Programming Concepts (using the chosen state management):** Observe how data changes propagate and update the UI automatically within the chosen state management pattern. 29. **Setting up a Firebase Project:** Create a new project in the Firebase console. 30. **Firebase Console Overview:** Explore the different services offered by Firebase in the console. 31. **Firestore Basics (Collections and Documents):** Create collections and documents in the Firestore database via the Firebase console. 32. **Reading and Writing Data to Firestore (Flutter):** Implement Flutter code

	to add, read, update, and delete data in Firestore. 33. **Real-time Data Updates (Flutter):** Implement Flutter code to listen for real-time changes in Firestore and update the UI accordingly using StreamBuilder. 34. **Firestore Queries (Filtering and Sorting):** Implement Flutter code to perform basic filtering (e.g., where clauses) and sorting of data retrieved from Firestore. 35. **Creating Custom Animations and Transitions:** Implement basic animations using AnimatedContainer, AnimatedOpacity, or TweenAnimationBuilder. 36. **Implementing Gesture Recognition:** Implement more complex gesture interactions like dragging or scaling using GestureDetector. 37. **Using Advanced Layout Techniques:** Explore widgets like CustomScrollView, SliverAppBar for more complex scrolling effects. 38. **Working with Canvas (Basic):** Draw simple shapes or text directly on the screen using the CustomPaint widget and Canvas. 39. **Local Data Storage using SharedPreferences:** Implement functionality to save and retrieve simple key-value data locally using SharedPreferences. 40. **Building Flutter Applications for Android:** Build an APK and App Bundle for an Android application. 40. **Introduction to App Store Deployment:** Research the basic steps involved in publishing an app to the Google Play Store and Apple App Store (account creation, app details, screenshots, etc.). (This might be more theoretical due to the requirements for actual deployment).
2	**Unit 1: Foundations of Data Mining and Visual Data Exploration** 1. **Data Exploration with Basic Plots:** o Load a sample dataset (e.g., Iris, Titanic). o Create histograms to visualize the distribution of numerical features. o Generate scatter plots to explore relationships between pairs of numerical features. o Use box plots to identify outliers and compare distributions of categorical groups. 2. **Data Preprocessing:** o Handle missing values in a dataset using different techniques (e.g., imputation). o Perform data integration by merging two or more datasets. o Apply data reduction techniques like sampling or feature selection (basic methods). o Implement data transformation methods such as normalization or standardization. **Unit 2: Basic Data Mining & Visualization**

3. **Classification and Visualization:**
    - Implement a basic Decision Tree classifier on a small dataset.
    - Implement a basic Naive Bayes classifier on a small dataset.
    - Visualize feature distributions for different classes using histograms or density plots.
    - Visualize class distributions using bar charts or pie charts.
    - Generate a confusion matrix (using a library) for a classification model and interpret the results.
    - (Conceptual) Visualize decision boundaries for a simple 2D classification problem using scatter plots with color-coded regions.
4. **Clustering and Visualization:**
    - Implement the K-Means algorithm on a sample dataset.
    - Visualize the clustering data using scatter plots before and after clustering.
    - Visualize the cluster assignments by coloring data points in a scatter plot based on their cluster.
    - Generate basic cluster profiles by calculating and visualizing the mean or median feature values for each cluster (e.g., using bar charts).
5. **Association Rule Mining and Visualization:**
    - Apply the Apriori algorithm (using a library) to a transactional dataset (e.g., a simplified market basket dataset).
    - Visualize item frequencies using bar charts.
    - Generate a scatter plot of association rules, plotting support against confidence (introduction).

Unit 3: Visualizing Hierarchical, Network, and High-Dimensional Data & Interactive Visualization

6. **Visualizing Hierarchical Data:**
    - Represent a simple organizational chart or file system structure using an indented list (text-based or using basic HTML).
    - (Using a visualization library) Create a basic node-link tree diagram for a small hierarchical dataset.
    - (Using a visualization library) Explore radial tree layouts for the same data and compare their effectiveness.
7. **Visualizing Network Data:**
    - Represent a small social network or web link structure using a node-link diagram (using a visualization library).
    - Experiment with different layout algorithms (e.g., spring layout) and observe their impact on the visualization.
    - Apply visual encoding (size, color) to nodes based on a node attribute (e.g., degree, category).
    - Create an adjacency matrix representation of a small network (e.g., using a spreadsheet or basic programming).

8. **Visualizing High-Dimensional Data:**
    - Generate a scatter plot matrix (SPLOM) for a dataset with several numerical features (using a visualization library).
    - Interpret the pairwise relationships shown in the SPLOM.
    - Discuss the limitations of SPLOMs as the number of dimensions increases.
9. **Interactive Visualization:**
    - (Using a visualization library like Matplotlib with interactive backends or a dedicated interactive library like Plotly or Bokeh) Create scatter plots or bar charts with basic interactive features:
        - Implement filtering based on a selected range of values.
        - Enable zooming and panning on the visualization.
        - Implement basic tooltips to display data point information on hover.
    - (Conceptual) Design a linked view visualization where selecting points in one plot highlights them in another related plot.

Unit 4: Overview of Data Mining and Visualization Tools

10. **Exploring KNIME or RapidMiner (or similar visual workflow tool):**
    - Build a simple data mining workflow for data loading, preprocessing, and basic visualization within the tool's interface.
    - Experiment with different data mining nodes (e.g., for filtering, sampling, basic modeling).
    - Create basic charts and graphs available within the tool.
11. **Working with Tableau or Power BI (or similar BI tool):**
    - Connect to a sample dataset.
    - Create various types of charts (bar charts, scatter plots, line charts).
    - Build a simple interactive dashboard with multiple linked visualizations and filters.

MSC (CA&IT) - Semester: VI
(Effective from year 2025-26)

Course Code:	CAIT-604		Course Title:	Digital Forensics and Cyber Law
Course Credits:	4		Hour of Teaching/Week:	4
Internal Assessment Marks:	50		External Exam Marks:	50
Exam Duration	2 Hours			

- CO1 : Understand the fundamentals of mobile application development, including the Android platform and the Dart programming language.
- CO2 : Develop user interfaces using Flutter widgets, manage user interactions, implement navigation, and apply styling and themes for visually appealing apps.
- CO3 : Apply different state management techniques in Flutter and integrate Firebase services, specifically Firestore, for data storage and real-time updates.
- CO4 : Implement advanced UI features, including custom animations, gesture recognition, and local data storage in Flutter applications.
- CO5 : Gain knowledge of the process for building and deploying Flutter applications on Android platforms and understand the basics of app store deployment.

Unit	Contents
1	**Foundations of Cyber Crime and Digital Evidence** • **Introduction to Cyber Crime Investigation:** Definition, scope, types of cyber crimes, challenges, law enforcement roles, first responder procedures. • **Cyber Warfare, Terrorism & Social Networking:** Understanding cyber warfare, cyber terrorism, cyber threats from social media, investigating social media crimes. • **Cyber Forensics and Incident Handling:** Definition and scope of cyber forensics, the forensics process, incident handling phases, their relationship. • **Cyber Forensic Basics:** Digital evidence (definition, characteristics, admissibility), sources of digital evidence, fundamental principles (chain of custody, integrity, avoiding contamination, sound tools). • **Storage Fundamentals - File System Concepts:** Basic storage concepts, understanding file systems (purpose, organization), common file systems (Windows, Linux, macOS - basics), file system metadata (MAC times and significance), file extensions.

2	**Investigating Real-World Cyber Crimes**
	• **Investigating Specific Cyber Crime Types:** In-depth look at investigating:
	  o Social Media Profile Impersonation
	  o Phishing Cases
	  o Data Theft Cases
	  o Corporate Espionage Cases
	  o Email Fraud Cases
	  o Credit Card Fraud Cases
	  o Cyber Pornography Cases
	  o Denial of Service Attacks Cases
	  o Cyber Defamation Cases [1]
	• For each crime type, the summary will cover identification, evidence gathering, legal aspects, and basic tracing techniques.

3	**Data Recovery, Preservation, and Legal/Ethical Considerations** • **Gathering Evidence:** Identifying sources, systematic approach, legal framework (warrants subpoenas). • **Precautions:** Maintaining a forensically sound environment, preventing contamination, proper device handling, and documentation. • **Preserving and Safely Handling Original Media:** Forensic imaging (concepts and tools like FTK Imager, EnCase Imager), verifying integrity (hashing), secure storage and transportation. • **Documenting Chain of Custody:** Definition, purpose, maintaining accurate records, legal implications of a broken chain, best practices. • **Complete Timeline Analysis:** Understanding MAC times, tools and techniques for extraction and analysis, identifying discrepancies, correlating with logs. • **Data Protection and Privacy:** Ethical considerations, legal frameworks (India's IT Act, IPC, GDPR implications), data minimization, secure handling of sensitive information.

4	**Cyber Forensics Investigation Process and Tools** • **Introduction to Cyber Forensic Investigation:** Overview of the process (detailed phases), developing an investigation plan, team roles, report writing and presentation. • **Investigation Tools – eDiscovery:** Definition, relevance, eDiscovery process (Identification to Production), overview of tools (Relativity, Nuix, open-source), keyword searching and filtering. • **Digital Evidence Collection:** Forensically sound methods from various sources (live vs. dead acquisition, imaging, volatile data, network traffic, cloud), maintaining integrity. • **Evidence Preservation:** Detailed look at write blockers, secure storage, legal and regulatory requirements for retention. • **Email Investigation:** In-depth analysis of headers, content, attachments, recovering deleted emails, investigating webmail and enterprise systems. • **Encryption and Decryption Methods:** Introduction to cryptography, common encryption methods, challenges of encrypted evidence, legal and technical aspects of decryption (ethical and legal boundaries). • **Search and Seizure of Computers:** Legal procedures for warrants, best practices for physical seizure, documentation, on-site triage. • **Work on Open Source, Commercial Tools:** Hands-on introduction to open-source tools (Autopsy, SIFT, Volatility), overview of commercial tools (EnCase, FTK), comparison.

Text Books & References

- "Law of Information Technology" by Paintal D, Taxmann Publications Pvt.Ltd
- "International domain name law: ICANN and the UDRP" by Lindsay D., Oxford-Hart Publishing.
- "Cyber Laws" by Sharma J. P & Kanojia S. Ane Books Pvt. Ltd New delhi
- "Cyber Laws" by Pavan Duggal, Universal Publication
- "Law relating to computers, internet and e-commerce: A guide to Cyber Laws and the IT Act, 2000 with rules, regulations and notifications" by Kamath N, Universal Law Publishing, New delhi
- https://onlinecourses.swayam2.ac.in/nou24_cs05/preview

MSC (CA&IT) - Semester: VI

(Effective from year 2025-26)

Course Code:	CAIT-605		Course Title:	The Startup Sprint
Course Credits:	2		Hour of Teaching/Week:	2
Internal Assessment Marks:	25		External Exam Marks:	25
Exam Duration	1 Hours			

- CO1: To discuss the basic understanding of the relevant concepts and practice of entrepreneurship.
- CO2: To recognize the essential criteria in the decision to become an entrepreneur or opt for entrepreneurship.
- CO3: To demonstrate how the entrepreneurial process is of moving from just an idea to a viable and sustainable launch of an entrepreneurial firm/venture.
- CO4: To practice with the basic tools and tactics required to manage and grow an entrepreneurial firm

Unit	Contents
1	Unit 1. DISCOVERING THE ENTREPRENEUR IN ME • The Psychology of An Entrepreneur • Who is an entrepreneur? • The entrepreneurial mind-set: looking inside the black box • Entrepreneurial expertise. • Overview of Entrepreneurship • Evolving concept of Entrepreneurship & Entrepreneurial Environment • How to avoid the pitfalls of entrepreneurship – The Dark Side • Myths & Realities about Entrepreneurship • Creativity in Entrepreneurship • The Creative Process – Stages/Steps • Creative Thinking – Techniques, Sources for Generation of Ideas, and Role of • Imagination • Effectuation, Improvisation, Analogous Thinking, etc. • Problem Finding and Solving – Divergent /Convergent, Associative approaches and • Improvisation • The Innovation Dimension • Barriers to Innovation • Typology of Innovation • Innovation and New Product Development – An overview

2	APPLIED ENTREPRENEURSHIP! (The Hatchery Sprint – A Hands-on Process of Entrepreneurship) Stage 1: Ideation & Team Formation • The Founding team [their motivation/skills/etc.] • Idea generation Stage 2: Skills & Resources • Resources and skills required • Logistical/Legal requirements Stage 3: Opportunity Analysis • Macro-environment scanning • Sizing the opportunity • Stakeholder analysis • Competition analysis Stage 4: Customer analysis • Customer segmentation • Understand the customers' context/needs; personas Stage 5 –The Early Solution • Mock-ups & Prototyping • Customer validation/demo Stage 6 – Business Viability • Value Proposition Canvas • Cost/Revenue Model • Marketing plan Stage 7 - Getting ready to launch • Business Model Canvas • Pitching • Social Impact Measurement
Text Book: 1. Charanthimath, Poornima M. (2016). Entrepreneurship Development and Small Business Enterprises. Pearson Education. Reference Books/Learning Resources: 1. McGrath, R. G., & MacMillan, I. C. (2000). The entrepreneurial mind set: Strategies For continuously creating opportunity in an age of uncertainty (Vol. 284). Harvard Business Press. 2. Neck, H. M., Neck, C. P., & Murray, E. L. (2017). Entrepreneurship: the practice and mind set. SAGE Publications.	

MSC (CA&IT) - Semester: VI
(Effective from year 2025-26)

Course Code:	CAIT-606		Course Title:	Project
Course Credits:	4		Hour of Teaching/Week:	-
Internal Assessment Marks:	50		External Exam Marks:	50
Exam Duration				

Rules and Regulations for a 4-Credit Individual Software Project (UG Course): In-House and Internship Options

1. Project Eligibility and Registration:

- Students may be required to have cleared certain prerequisites like approval of the project proposal from competent authority.
- Project registration will be conducted within a specified timeframe announced by the department.
- Students can choose to pursue either:
    - **In-House Individual Software Project:** Proposing and developing a software project within the academic environment under the guidance of a faculty supervisor.
    - **Internship-Based Individual Software Project:** Undertaking a substantial software development project as part of a departmentally approved internship with an external organization.
- All students must individually register their project choice (in-house topic or internship details) and be assigned a faculty supervisor.
- For internship-based projects, the internship organization and the proposed project/tasks must be approved by the department to ensure academic rigor and relevance.

2. Project Proposal and Approval:

- **In-House Projects:** Each student must submit a detailed individual project proposal (as outlined in the previous in-house regulations).
- **Internship-Based Projects:** Each student must submit a project proposal that includes (as outlined in the previous internship regulations):
    - Internship organization details and contact person.
    - Description of the internship role and responsibilities.
    - Clear outline of the specific software project to be undertaken during the internship.
    - Expected learning outcomes and alignment with coursework.
    - Letter of confirmation/offer from the internship organization.

    - Name and contact details of the industry supervisor/mentor.
- All project proposals (both in-house and internship-based) will be reviewed and approved by a committee constituted by the department.
- Students may be required to present their project proposal individually to the committee for approval.
- Supervisors will be assigned based on the project topic/internship and faculty expertise. For internship-based projects, the faculty supervisor will liaise with the industry supervisor.

3. Roles and Responsibilities:

- **Student (Both Options):**
    - Take primary and sole responsibility for all aspects of the project (planning, design, development, testing, documentation, and for internships, active participation).
    - Regularly consult with their faculty supervisor and seek guidance. For internships, also communicate regularly with the industry supervisor.
    - Adhere to the project timeline and meet deadlines.
    - Maintain a detailed project logbook/diary documenting their work (including internship activities for internship-based projects).
    - Conduct all work ethically and with academic integrity.
    - Prepare and submit progress reports, a comprehensive final project report, and other required deliverables.
    - Present their project work and (for internships) internship experience.
    - Participate in the individual viva voce examination.
- **Faculty Supervisor (Both Options):**
    - Provide guidance and mentorship to the student throughout the project.
    - Help in defining the project scope and objectives (in collaboration with the industry supervisor for internships).
    - Provide feedback on proposals, progress reports, and final reports.
    - Monitor project progress and provide timely advice. For internships, maintain communication with the industry supervisor.
    - Assess the student's work and provide feedback for evaluation.
    - Ensure the project aligns with academic standards and ethical guidelines.
- **Industry Supervisor/Mentor (Internship-Based Only):**
    - Provide guidance and mentorship to the student during the internship.
    - Assign relevant software development project tasks and provide necessary resources.
    - Provide feedback on the student's performance and progress to both the student and the faculty supervisor (if required).
    - May contribute to the evaluation process in consultation with the faculty supervisor (e.g., providing a performance assessment).

4. Project Deliverables and Timeline:

- A clear timeline for project completion will be established and communicated. This will be adapted for internship-based projects to accommodate the internship duration. Key deliverables include:
    - Project Proposal (as per the chosen option).
    - Project Logbook/Internship Diary.
    - Progress Reports (frequency may vary based on the option).
    - Draft Final Report.
    - Final Project Report (content will vary slightly based on the option, with internship reports including details of the organization and the student's role).
    - Software Demonstration (for both options).
    - (For Internships) Letter of Completion/Performance Certificate from the internship organization.
- Specific guidelines for the format and content of the project report and software documentation will be provided by the department.
- Submission deadlines must be strictly adhered to. Late submissions may be subject to penalties.

5. Project Assessment and Evaluation:

- The individual project will be evaluated based on a combination of factors, with slight variations depending on the option:
    - **Both Options:** Proposal quality, progress reports, software design and implementation, functionality, testing, final report quality, software demonstration, presentation skills, viva voce performance, adherence to academic integrity.
    - **Internship-Based Specifics:** Relevance and academic rigor of the internship project, student's engagement and contribution during the internship (based on diary and industry supervisor feedback), integration of internship experience in the final report, feedback from the industry supervisor.
- The evaluation will be conducted by a panel of examiners.
- A detailed marking scheme outlining the weightage for each component will be provided.
- Plagiarism in code and documentation is strictly prohibited.
- For internship-based projects, the evaluation will consider both the academic merit of the software project and the student's learning and contribution during the internship.