MADANAPALLE INSTITUTE OF TECHNOLOGY & SCIENCE
(Deemed to be University)
MADANAPALLE
www.mits.ac.in

Estd 1998

Department of Computer Applications
Course Structure
&
Detailed Syllabi (R25)

For the students admitted to

Master of Computer Applications from the Academic Year 2025 – 26 Batch onwards

MCA Regular Two Year P.G. Degree Course

MADANAPALLE INSTITUTE OF TECHNOLOGY & SCIENCE

(Deemed to be University)

MADANAPALLE

MCA Two Year Curriculum Structure

Total Credits	86 Credits for 2025 Admitted Batch onwards

R25 - Curriculum Structure
I Year I Semester

S.No	Course Code	Course Title	L	T	P	C
1.	25MDMATTC01	Computational Mathematics	3	0	0	3
2.	25MDMCAEC01	Relational Database Management Systems	3	0	2	4
3.	25MDMCAEC02	Data Structures and Algorithms	3	0	2	4
4.	25MDMCAEC03	Advanced JAVA Programming	3	0	2	4
5.	25MDMCAEC04	Operating Systems	3	0	2	4
6.	25MDENGLC01	Corporate Communication Laboratory	0	0	4	2
7.	25MDMCASC01	Frontend Web Development	1	0	2	2
8.		**NSDC - Certification Course – I** * *(Refer ANNEXURE – III)*	0	0	2	0
Total			**16**	**0**	**16**	**23**

I Year II Semester

S.No	Course Code	Course Title	L	T	P	C
1.	25MDMCAEC05	Software Engineering and Design Patterns	3	0	2	4
2.	25MDMCAEC06	Artificial Intelligence	3	0	2	4
3.	25MDMCAEC07	Advanced Web Technologies and MERN Stack	3	0	2	4
4.		**Discipline Elective – I** *(Refer ANNEXURE – I)*	3	0	2	4
5.		**Discipline Elective – II** *(Refer ANNEXURE – I)*	3	0	2	4
6.	25MDMCALC02	Competitive Coding Laboratory	0	0	2	1
7.	25MDMCASC02	Mobile Application Development	1	0	2	2
8.		**NSDC - Certification Course – II** * *(Refer ANNEXURE – III)*	0	0	2	0
Total			**16**	**0**	**16**	**23**

(L = Lecture, T = Tutorial, P = Practical, C = Credit)

* NSDC – Certification Course I & II applicable only from 2026 admitted batch onwards

II Year I Semester

S. No	Course Code	Course Title	L	T	P	C
1.	25MDMCATC01	Internet of things	3	0	0	3
2.	25MDMCATC02	Generative AI	3	0	0	3
3.		**Discipline Elective – III** *(Refer ANNEXURE – I)*	3	0	2	4
4.		**Discipline Elective – IV** *(Refer ANNEXURE – I)*	3	0	2	4
5.		**Open Elective** *(Refer ANNEXURE – II)*	3	0	0	3
6.	25MDCOMTC01	Research Methodology and IPR	2	0	0	2
7.	25MDMCASC03	AI-Based Software Development	1	0	2	2
8.	25MDMCAIC01	Summer Internship	0	0	6	3
Total			**18**	**0**	**12**	**24**

II Year II Semester

S.No	Course Code	Course Title	L	T	P	C
1	25MDMCAPC01	Capstone Project	0	0	32	16
Total			**0**	**0**	**32**	**16**

(L = Lecture, T = Tutorial, P = Practical, C = Credit)

ANNEXURE – II

LIST OF DISCIPLINE ELECTIVE COURSES

Discipline Elective (AI and ML Stream)

S. No	DE No	Course Code	Course Title	L	T	P	C
1	DE1	25MDMCADC01	Data Analytics and Visualization	3	0	2	4
2	DE2	25MDMCADC02	Machine Learning	3	0	2	4
3	DE3	25MDMCADC03	Natural Language Processing	3	0	2	4
4	DE4	25MDMCADC04	Deep Learning for Computer Vision	3	0	2	4

Discipline Elective (Cloud Computing Stream)

S. No	DE No	Course Code	Course Title	L	T	P	C
1.	DE1	25MDMCADC05	Cloud Computing	3	0	2	4
2.	DE2	25MDMCADC06	Edge and Fog Computing	3	0	2	4
3.	DE3	25MDMCADC07	Cloud Web Services	3	0	2	4
4.	DE4	25MDMCADC08	Cloud Security	3	0	2	4

Discipline Elective (Cyber Security Stream)

S. No	DE No	Course Code	Course Title	L	T	P	C
1.	DE1	25MDMCADC09	Cyber Security	3	0	2	4
2.	DE2	25MDMCADC10	Applied Cryptography	3	0	2	4
3.	DE3	25MDMCADC11	Digital Forensics	3	0	2	4
4.	DE4	25MDMCADC12	Ethical Hacking	3	0	2	4

Discipline Elective (Software Engineering Stream)

S. No	DE No	Course Code	Course Title	L	T	P	C
1.	DE1	25MDMCADC13	Agile Methodologies	3	0	2	4
2.	DE2	25MDMCADC14	DevOps and Microservices	3	0	2	4
3.	DE3	25MDMCADC15	Software Testing	3	0	2	4
4.	DE4	25MDMCADC16	Software Project Management	3	0	2	4

ANNEXURE – II

LIST OF OPEN ELECTIVE COURSES

Open Elective **(To be offered under MOOC's Category from SWAYAM – NPTEL)**			
S.No	**Course Code**	**Course Title**	**Course Offered by the Department of**
1	25MDMBAOM01	Entrepreneurship	Management Studies
2	25MDMBAOM02	Management Accounting	Management Studies
3	25MDMBAOM03	Managerial Skills for Interpersonal Dynamics	Management Studies
4	25MDMBAOM04	Innovation, Business Models and Entrepreneurship	Management Studies
5	25MDMBAOM05	Management Information System	Management Studies
6	25MDMBAOM06	Ethics in Engineering Practice	Management Studies
Any new Interdisciplinary Course offered by SWAYAM NPTEL can be appended in future.			

MCA I Year I Semester

MCA I Year I Semester

25MDMATTC01 COMPUTATIONAL MATHEMATICS

L	T	P	C
3	0	0	3

Pre-requisite: None

Course Description:

Computational Mathematics is a foundational course that equips students with essential mathematical tools used in computer science, data analysis, and problem solving. It covers core topics including logic and proofs, graph theory, descriptive statistics, probability, and hypothesis testing. Students will learn to construct valid logical arguments, analyze networks using graph models, summarize and visualize data, understand and apply probability distributions, and perform statistical tests to make informed decisions. The course emphasizes both theoretical understanding and practical application, preparing students for advanced studies and real-world computational challenges.

Course Objectives:

This course enables students to :

1. Analyze and construct logical arguments using propositional and predicate logic.
2. Model real-world problems using graph theoretic structures and properties.
3. Compute descriptive statistics and create visualizations to interpret data sets.
4. Apply probability distributions to calculate the likelihood of events and moments of random variables.
5. Evaluate claims about population parameters by performing appropriate statistical hypothesis tests

UNIT I LOGIC AND PROOFS 12 Hours

Basic terminologies and logic connectives, Propositional logic- Logical equivalence -Tautologies, Rules of inference for propositional logic, Normal forms- Predicate (First-Order) Logic- Quantifiers: ∀ (for all), ∃ (there exists)-Models and interpretations.

UNIT II GRAPH THEORY 12 Hours

Basic definitions and terminology, Matrix Representation of Graphs, Subgraphs, walks, trails, paths, cycles and graph connectivity, Isomorphism of simple graphs, Euler graph, Hamiltonian circuits and Hamiltonian path, Graph coloring and chromatic number-Applications: scheduling, map coloring.

UNIT III DESCRIPTIVE STATISTICS 12 Hours

Measures of central tendencies and Dispersion, Coefficient of variation- Data visualization, Grouped data, Histograms, Ogives, Percentiles, Box-Plot, Correlation, Scatter diagram, Rank correlation and Linear Regression.

UNIT IV PROBABILITY AND RANDOM VARIABLES 12 Hours

Probability – Axioms of probability – Conditional probability – Bayes theorem – Random variables – Probability function – Moments – Moment generating functions and their properties – Distributions: Binomial, Poisson, Geometric, Uniform, Exponential, Gamma and Normal.

UNIT V TESTING OF HYPOTHESIS 12 Hours

Sampling distributions – Type I and Type II errors – Small and Large samples – Testing of Hypothesis: Normal, t, Chi square and F distributions–Independence of attributes - goodness of fit.

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Apply logic to construct and validate arguments.
CO2: Solve problems using graph theory concepts.
CO3: Summarize and interpret data statistically and visually.
CO4: Apply probability distributions to model uncertainty.
CO5: Conduct hypothesis tests to draw conclusions.

Text Books

1. Kenneth H. Rosen, *Discrete Mathematics and Its Applications*, 7th Edition, McGraw-Hill Education, 2012.
2. Gupta, S. C., and V. K. Kapoor. *Fundamentals of mathematical statistics*. Sultan Chand & Sons, 2020.

Reference Books

1. Huth, Michael, and Mark Ryan. *Logic in Computer Science: Modelling and reasoning about systems*. Cambridge university press, 2004.

2 Diestel, Reinhard. *Graph theory*. Vol. 173. Springer Nature, 2025.

3 Chan, Joshua CC, and Dirk P. Kroese. *Statistical modeling and computation*. Springer, 2025.

4. Freedman, D., Pisani, R., & Purves, R.Statistics (4th ed.). W. W. Norton & Company, 2007.

Mode of Evaluation: Assignments, Mid Term Tests and End Semester Examination.

MCA I Year I Semester

25MDMCAEC01 RELATIONAL DATABASE MANAGEMENT SYSTEMS

L	T	P	C
3	0	2	4

Pre-requisite None

Course Description:

This course introduces the fundamental concepts of Database Management Systems. It covers data models, relational algebra, Structured Query Language (SQL), normalization, transaction management, and database recovery. It emphasizes database design, query processing, concurrency control, and introduces students to modern database applications.

Course Objectives:

This course enables students to:

1. Differentiate database systems from file systems and model data with ER diagrams.
2. Apply relational algebra and construct SQL queries.
3. Normalize database schemas to eliminate redundancy.
4. Analyze transactions and implement concurrency control.
5. Develop PL/SQL programs and compare advanced database systems.

UNIT I INTRODUCTION TO DATABASES 9 Hours

Introduction to data bases, File system vs. Database system. Database system architecture- Data models- Entities, Attributes, Entity sets, Relationships and Relationship sets, Database design and ER diagrams, Specialization and Generalization.

UNIT II RELATIONAL MODEL & SQL 9 Hours

Relational model concepts– Relational algebra & relational calculus– SQL – Basic Queries DDL, DML, DCL, TCL, subqueries, joins, views, Integrity constraints.

UNIT III DATABASE DESIGN 9 Hours

Functional Dependencies-Normalization: 1NF, 2NF, 3NF, BCNF- Decomposition & dependency preservation- File operations-B+ Tree- Indexing & hashing.

UNIT IV TRANSACTION MANAGEMENT & CONCURRENCY 9 Hours

ACID properties-Transactions and Schedules, Concurrent Execution of transactions, Serializability - Concurrency control techniques: Locking, timestamp ordering-Deadlock Handling-Recovery techniques & log-based recovery

UNIT V PL/SQL & ADVANCED DATABASES 9 Hours

PL/SQL: Functions, procedures, triggers, cursors and exceptional handling, Package in PL/SQL. No SQL Distributed Databases-spatial databases- Temporal databases, Cloud databases, Object Oriented Databases- Object-Relational Mapping (ORM) Frameworks-SQL Alchemy, Dapper**,** Apache Cassandra.

List of Experiments **30 Hours**

1. Draw a complete Entity-Relationship (ER) diagram for the case study using modeling software.
2. Write SQL DDL commands to create the tables, alter them (add/drop columns), create indexes, and finally drop them.
3. Draw ER diagram for any application (i.e., library management) and convert the final ER diagram into a set of relational tables, specifying primary keys and foreign keys.
4. Create two tables: employees(emp_id, name, dept_id) and departments(dept_id, dept_name).

Write queries to:

   a. Display employee names with their department names (using JOIN).
   b. Find employees who work in the 'HR' department.
   c. Use a subquery to list employees whose salary is above the average salary.

5. Create a table called students with the following columns: *student_id (INT, primary key), name (VARCHAR(100)), age (INT), major (VARCHAR(50))*. Insert at least 5 student records into the table.
   a. Write a query to find all students majoring in Computer Science.
   b. Write a query to update the major of a student using student_id.
   c. Alter the students table to add enrollment date and update enrollment date for existing records.
   d. Find maximum and minimum age.
   e. Find students who enrolled after Jan 1, 2023
   f. Find students enrolled in the current year

6. Create tables: ***Orders*** (*OrderID, CustomerID, OrderDate, Amount*) ,***Customers*** (*CustomerID, CustomerName, Country*) and perform the following queries.
   a. Write a nested query to find the names of customers who have placed orders with an amount greater than $1000.
   b. Create a view that shows CustomerName, OrderID, and Amount for all orders placed by customers from the country 'USA'.
7. Write a SQL transaction that transfers $500 from account 'A1001' to account 'A1002'. Include BEGIN, COMMIT, and a check to ROLLBACK if any account's balance falls below zero.
8. Write a PL/SQL function called ***get_employee_salary*** that takes an ***emp_id*** and returns the salary of that employee from the Employees (*emp_id, name, salary, dept_id*) table.
9. Write a PL/SQL procedure called ***add_new_employee*** that takes parameters for *name, salary*, and *dept_id,* and inserts a new record into the Employees table. Generate the new emp_id automatically as the current maximum emp_id + 1.
10. Write a PL/SQL block that uses a cursor to loop through the Employees table and increases the salary of each employee by 5% if their current salary is less than 60000.
11. Create a BEFORE INSERT trigger on the ***Works_On***(*emp_id*, *project_id*, Hours_worked) table. The trigger should check if the Hours worked value being inserted is greater than 40. If it is, automatically set it to 40.

Mini Projects:

Each student has to implement any one of the following systems.

1. Hotel Reservation System
2. Bank Management system
3. Library information system

4. Hospital Management System
5. Railway reservation system
6. Inventory Management System

Software Requirements: Oracle Database 21c Express Edition (XE)
Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Design database schemas using conceptual data models and ER diagrams.
CO2: Construct and execute queries using SQL and relational algebra.
CO3: Normalize relational database schemas to eliminate anomalies.
CO4: Implement transaction management and concurrency control techniques.
CO5: Develop database programs using PL/SQL and evaluate advanced database systems.

Text Books

1. Abraham Silberschatz, Henry F. Korth, S. Sudarshan, *Database System Concepts*, 7th Edition, McGraw Hill 2019.
2. C.J. Date, *An Introduction to Database Systems.2006.*

Reference Books

1. Ramez Elmasri, Shamkant B. Navathe, *Fundamentals of Database Systems*, Pearson. 2015.
2. Raghu, Ramakrishnan, and Gehrke Johannes. "Database Management System 2nd edition.", 2018

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

MCA I Year I Semester

25MDMCAEC02 DATA STRUCTURES AND ALGORITHMS

L	T	P	C
3	0	2	4

Pre-requisite None

Course Description:
This course provides a strong foundation in Data Structures and Algorithms, blending theory with practical implementation. Students will learn to design, analyze, and implement efficient algorithms with a focus on time and space complexity. The course also introduces computational complexity classes like NP, enhancing students' understanding of problem-solving limits. By the end, learners will be equipped to handle complex algorithmic challenges and apply their skills to real-world problems.

Course Objectives:
This course enables students to:

1. Understand the fundamental data structures and their practical applications in problem-solving.
2. Construct advanced data structures for complex scenarios.
3. Analyze algorithm efficiency using complexity analysis.
4. Apply and optimize sorting and graph algorithms.
5. Design solutions using dynamic programming and backtracking.

UNIT I LINEAR DATA STRUCTURES

Importance of good data structures and algorithms. Arrays, Stacks, Evaluation of expressions: infix to postfix, evaluating postfix expressions. Queue, Priority Queue singly linked list, circular linked list, doubly linked list.

UNIT II NON-LINEAR DATA STRUCTURES 9 Hours

Binary tree, binary search tree, AVL trees, Splay tree, Red-black trees, B-Trees and B+ Trees, binary heap, skip list. Graph terminology, graph representations, graph traversals,

UNIT III ALGORITHMS COMPLEXITY AND ANALYSIS 9 Hours

The Role of Algorithms in problem-solving and computing, time complexity, space complexity, **Growth of functions, Asymptotic notation**s. **Recurrence relations: Substitution method, recursion tree method**, **Master theorem, Analysis of searching techniques: binary search,** hashing techniques.

UNIT IV ALGORITHM DESIGN PRINCIPLES 9 Hours

Divide and conquer: Quick sort, **Merge sort**, **Greedy Methods**: Minimum spanning trees (MST), Prim's algorithm, Kruskal's algorithm, applications of MST, Huffman coding, Single source shortest-path algorithms: Dijkstra's algorithm, Floyd-Warshall algorithm.

UNIT V PROBLEM-SOLVING METHODS AND APPROXIMATION ALGORITHMS 9 Hours

Dynamic programming (DP): Sum of subsets, travelling salesman problem **Backtracking:** 0/1 Knapsack problem, **Branch and Bound**: N-queen problem. Introduction to P, NP, NP-Hard, and NP-complete, Clique problem, **reducibility**, Cook's Theorem (without proof).

LIST OF EXPERIMENTS 30 Hours

1.Create a Stack and perform its operations.
2. Create a queue and perform its operations.
3. Write a program to convert a given infix arithmetic expression into its equivalent postfix notation.
4. Implement a singly linked list and perform the following operations:
   a. Insertion b. Deletion c. Traversal.
5. Implement the operations (insert, delete, search) on Binary Search tree.
6. Write a program on Tree Traversal
   a. Pre order b. In order c. Post order
7. Write a program to implement the **AVL** Tree
8. Implement the following Divide and Conquer techniques.
   a. Quick sort c. Merge sort
9. Write a program to implement Heap sort for the given list of integer values.
10. Implement Graph traversals. a. BFS b. DFS
11. Write programs to find the Minimum Spanning Tree of a given weighted, connected graph using:
    a. Prim's method b. Kruskal's method.
12. Implement Dijkstra's algorithm to solve the single-source shortest path problem
13. Implement N-queen's problem using backtracking.
14. Write a program to find the solution for a knapsack problem using Branch and Bound

Software requirements: Geany, Notepad++

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Implement basic linear and non-linear data structures.
CO2: Analyze algorithm efficiency using asymptotic notation and recurrence.
CO3: Compare sorting, searching, and hashing techniques.
CO4: Construct solutions using divide & conquer, greedy, and dynamic programming.
CO5: Classify problems by complexity and propose heuristic strategies.

Text Books

1. **Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to algorithms.* MIT press, 2022**
2. Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman. *Data Structures and Algorithms,* Addison-Wesley, 2007

Reference Books

1. Sanjoy Dasgupta, Christos Papadimitriou, Umesh Vazirani. *Algorithms*. McGraw Hill, 2023

2. Jon Kleinberg, Éva Tardos. *Algorithm Design,* Pearson Education India, 2006

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

MCA I Year I Semester

25MDMCAEC03 ADVANCED JAVA PROGRAMMING

L	T	P	C
3	0	2	4

Pre-requisite: None

Course Description:
This course introduces the core and advanced concepts of Java. It covers the core concepts of JVM architecture and OOPS concepts for building platform independent java applications. This course also deals with Implementation of Exception Handling, multi-threading in java programs. It also introduces the advanced concepts like Collection Framework and designing GUI Components using Swings.

Course Objectives:
Student will be able to:

1. Understand Java basics, primitive types, control structures, and JVM architecture.
2. Explore object-oriented principles using classes, inheritance, and polymorphism.
3. Apply interfaces, packages, and exception handling in Java applications.
4. Implement multithreading and manage data using the Collection Framework.
5. Develop GUI applications using AWT, Swing, and event handling mechanisms.

UNIT I INTRODUCTION TO JAVA 9 Hours

The History of Java, Java Primitive Data types, Variables, Type Casting: Widening vs Narrowing. Using Command-Line Arguments, input through Scanner Class. **Programming Constructs:** Selection, Iteration and Jump Statements. Java Virtual Machine Architecture-Byte Code.

UNIT II CLASSES, POLYMORPHISM AND INHERITANCE 9 Hours

Classes and Objects: Object Oriented Programming and its principles, Class, Objects, Methods, Constructors, keywords (this, super, Static and Final). Abstract Classes. Polymorphism: Method Overloading and Method Overriding, Early Binding Vs Late Binding. Inheritance: Extends Keyword, Types of Inheritances, Develop a case study on Multiple Inheritance Issues.

UNIT III INTERFACES, PACKAGES & EXCEPTION HANDLING 9 Hours

Interfaces, API packages and CLASSPATH, Access protection Levels, User Defined packages. **Exceptions:** Exception handlers, try and catch, Multiple catch clauses, nested try statements, throw, throws, finally block, Types of Exceptions.

UNIT IV MULTI-THREADING & COLLECTION FRAMEWORK 9 Hours

Multithreading: Thread Life Cycle, Creating Threads, Thread class Methods, Deamon Thread.
Collection Framework: Collection Hierarchy, Iterable interface, Cursors of Collection Framework, Interfaces: List, Queue, Set and Map, Implementations: Array List, LinkedList, Vector, Stack, Deque and HashSet.

UNIT V AWT, EVENT HANDLING & SWING 9 Hours

Applet, Applet Life Cycle, AWT, GUI Components, Event Handling: Event Delegation Model, Sources of Events, Event Listeners, Adapter Classes-Introduction to Swings, JLabel, JImageIcon, JTextField, JButton, JToggleButton Case Study: Develop a GUI Application using JavaFX.

List of Experiments **30 Hours**

1. a) Write a JAVA program to display default value of all primitive data types of JAVA
   b) Write a JAVA program to display the Fibonacci sequence
2. Ten Bikers Compete in a race such that they drive at a constant speed which may or may not be the same as the other. To qualify the race, the speed of a racer must be more than the average speed of all 10 racers. Take as input the speed of each racer and print back the speed of qualifying racers.
3. Write a Java program for creating one base class for student personal details and inherit those details into the sub class of student educational details to display complete student information.
4. Write a java program to demonstrate Runtime polymorphism, using subclass and super class hierarchy override superclass methods using parent child references.
5. Write a JAVA program to create an abstract class named Shape that contains two integers and an empty method named printArea(). Provide three classes named Rectangle, Triangle and Circle such that each one of the classes extends the class Shape. Each one of the classes contains only the method printArea() that prints the area of the given Shape.
6. Given two strings, **S1** and **S2**, determine if they are anagrams of each other. Two strings are considered anagrams if they contain the same characters with the same frequencies, regardless of the character order. The comparison should be case-insensitive and ignore any whitespace. For example, "Listen" and "Silent" are anagrams.
7. Given an input string, **S**, the task is to produce a new string that contains the characters of **S** in reverse order. This operation must be performed without utilizing any pre-existing Java methods specifically designed for string reversal
8. Demonstrate how to handle multiple Exceptions in a java application-using try with Multiple Catch Blocks.
9. Write a java program that implements a multi-thread application that has three threads. First thread generates random integer every 1 second and if the value is even, second thread computes the square of the number and prints. If the value is odd, the third thread will print the value of cube of the number.
10. Implement a dynamic collection of objects using Array List and perform various operations where elements can be added, removed, accessed, and modified efficiently.
11. Write a Java Program a simple user form which reads the name of a user and mail id in Text fields, select gender with radio buttons, selects some Known languages using checkboxes, and also enters an address in a text area. After filling details whenever a user presses the “submit” button, then displays all the information about the user input.
12. Write a Java Program to create a frame using swing in which create a push button with a label and image. When the button is clicked an image is displayed in the Frame?

Software Requirements: Notepad++ and Geany

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Understand java program using basic constructs of java.
CO2: Analise Object Oriented Principles.
CO3: Implement interfaces and Exception handling concepts in java applications.
CO4: Apply multithreaded and Collection concepts in developing applications
CO5: Design and Implement Graphical User Interfaces (GUIs).

Text Books

1. Java: A Beginner's Guide, Herbert Schildt and Danny Coward,10th Edition, 2024
2. Advanced Java Dr. C. Muthu-Shalom Infotech, 2015

Reference Books

1. Horstmann, Cay S. *Core java, volume I: fundamentals*. Pearson Education, 2024.
2. Buyya, Rajkumar, S. Thamarai Selvi, and Xingchen Chu. *Object-oriented programming with Java: essentials and applications*. Tata McGraw-Hill, 2009.
3. Java—One Step Ahead, Anita Seth & B.L. Juneja, First Edition. 2017.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

MCA I Year I Semester

25MDMCAEC04 OPERATING SYSTEMS

L	T	P	C
3	0	2	4

Pre-requisite: None

Course Description:

This course offers a comprehensive study of operating system concepts, design, and implementation. It covers the fundamental components of operating systems, including process management, memory management, file systems, and input/output systems. The curriculum also explores real-time operating systems, with an emphasis on scheduling algorithms and resource management techniques. In addition, students will gain hands-on experience through practical exercises in shell scripting and system administration tasks.

Course Objectives:

This course enables students to:

1. Understand the basic functions, types, and architecture of operating systems.
2. Explore process synchronization, memory management techniques, and deadlock handling.
3. Explain file systems, storage structures, and disk scheduling algorithms.
4. Examine real-time operating systems and compare Android, iOS, Linux, and Windows.
5. Apply shell programming concepts and system administration tasks in Unix/Linux.

UNIT I INTRODUCTION TO OPERATING SYSTEMS 9 Hours

Definition, Functions of Operating Systems- Operating System Architecture -Types of Operating Systems- Operating System Components- System Calls. Process States, Context Switching, CPU Scheduling (FCFS, SJF, Priority Scheduling, Round Robin) and Evaluation.

UNIT II PROCESS SYNCHRONIZATION AND MEMORY MANAGEMENT 9 Hours

Process Synchronization: Critical Section Problem, Semaphores, Monitors. Classical Synchronization Problems, Deadlocks management**,** Banker's Algorithm. Contiguous Memory Allocation - Paging, Segmentation- Demand Paging, Page Replacement Algorithms (FIFO, Optimal, LRU), Thrashing.

UNIT III FILES AND STORAGE MANAGEMENT 9 Hours

Storage Management: File System Interface and Implementation-**File Allocation Methods:** Contiguous, Linked, Indexed, Directory Structure, File Protection, File Access Methods - Disk Scheduling Algorithms (FCFS, SSTF, SCAN, C-SCAN, LOOK, C-LOOK).

UNIT IV REAL-TIME OPERATING SYSTEMS (RTOS) 9 Hours

Characteristics and Applications of RTOS-Types of Real-Time Systems: Hard vs Soft Real-Time Systems- Mobile OS – Android Vs IOS layered Architecture-RTOS Kernel: Process Management, Scheduling in RTOS - Rate Monotonic Scheduling, Earliest Deadline First (EDF)- Inter-process Communication in RTOS-Memory Management in RTOS. **Case studies**: Analysing Linux and Windows Operating systems.

UNIT V SHELL PROGRAMMING AND SYSTEM ADMINISTRATION 9 Hours

Shell Types: Bourne Shell, C Shell, Bash. Shell Script Basics: Variables, Operators, Control Statements (if, case, for, while). Functions and User-defined Commands. **Filters and Utilities**: grep, sed, awk, sort, uniq, wc, etc.- user Management, File Permissions, Disk Management, Process Monitoring.

List of Experiments 30 Hours

1. Understand the basic functions, types, and architecture of operating systems.
2. Explore process synchronization, memory management techniques, and deadlock handling.
3. Explain file systems, storage structures, and disk scheduling algorithms.
4. Examine real-time operating systems and compare Android, iOS, Linux, and Windows.
5. Apply shell programming concepts and system administration tasks in Unix/Linux.
   Simulate the following CPU scheduling algorithms
   a) FCFS b) SJF c) Priority d) Round Robin
6. Control the number of ports opened by the operating system with
   a) Semaphore b) Monitors.
7. Write a program to illustrate concurrent execution of threads using pthreads library.
8. Write a program to solve producer-consumer problem using Semaphores.
9. Implement the following memory allocation methods for fixed partition
   a) First fit b) Worst fit c) Best fit
10. Simulate the following page replacement algorithms
    a) FIFO b) LRU c) LFU
11. Implement Bankers Algorithm for Dead Lock avoidance and prevention
12. Simulate Paging Technique of memory management.
13. Simulate the following file allocation strategies
    a) Sequential b) Indexed c) Linked
14. Simulate the following Disk scheduling algorithms
    a) FCFS, b) SSTF, c) SCAN, d)LOOK
15. Write programs using the following UNIX operating system calls. fork, exec, getpid, exit, wait, close, stat, opendir and readdir
16. Simulate UNIX commands like cp, ls, grep, etc.,

Software requirements: Geany, Ubuntu,GCC Compiler and Vim

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Explain the fundamental concepts, structures, and functions of operating systems.

CO2: Apply CPU scheduling and process synchronization techniques, and analyze deadlock handling strategies.

CO3: Implement memory management and storage management techniques

CO4: Differentiate between general-purpose operating systems and real-time operating systems, and evaluate real-time scheduling strategies.

CO5: Develop shell scripts and perform basic system administration tasks using Unix/Linux filters and utilities.

Text Books

1. Abraham Silberschatz, Peter B. Galvin, and Greg Gagne. *Operating System Concepts*. 10th ed., Wiley, 2018
2. William Stallings. *Operating Systems: Internals and Design Principles*. 9th ed., Pearson Education, 2018.

Reference Books

1. Andrew S. Tanenbaum and Herbert Bos. *Modern Operating Systems*. 5th ed., Pearson, 2022.
2. Richard Blum and Christine Bresnahan. *Linux Command Line and Shell Scripting Bible*. 4th ed., John Wiley & Sons, Inc., 2021

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

MCA I Year I Semester

25MDENGLC01 CORPORATE COMMUNICATION LABORATORY

L	T	P	C
0	0	4	2

Pre-requisite: None

Course Description:

English is practical and it is a must for any institution to provide students with opportunities to indulge in actively applying their language skills. Thus, the Communication Skills Lab facilitates students with adequate opportunities to put their communication skills in use. It also accommodates peer learning by engaging students in various interactive sessions. This lab will be accompanied by a practical lab component.

Course Objectives:

This course enables students able to

1. Comprehend listening tasks and enhance formal conversations and presentations skills
2. Read and write various genres of texts encountered in corporate scenario
3. Prepare and succeed in verbal ability round of job interviews and competitive exams
4. Prepare and succeed in HR interview round of the job selection process
5. Enhance pronunciation and neutralize accent

UNIT I LISTENING AND SPEAKING SKILLS 9 Hours

Listening/watching interviews, conversations, documentaries, etc.; Listening to lectures, discussions from TV/Radio/Podcast. Conversational skills (Formal and Informal); Group Discussion; Making effective Power Point presentations.

UNIT II READING AND WRITING SKILLS 9 Hours

Reading different genres of texts including newspapers Magazines: creative writing; Writing job applications and resume; Emails; Letters; Memorandum; Reports; Writing abstracts and summaries; Interpreting visual texts.

UNIT III ACCLIMATIZING STUDENTS TO OTHER EXAMS 9 Hours

Test of English as a Foreign Language (TOEFL); International English language Testing System (IELTS); Civil Service Examinations; Verbal-ability.

UNIT IV INTERVIEW SKILLS 9 Hours

Different types of interviews: Answering questions and offering information; Mock interviews; Body Language.

UNIT V PHONETICS 9 Hours

Vowels, Consonants, Articulation of sounds, Neutralization of Accent; Word Stress, Sentence Stress and Intonation.

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Comprehend audio-visual content in English and improve their conversational and presentation skills
CO2: Understand various genres of texts and write texts used as a part of corporate communication
CO3: Apply verbal ability and reasoning skills in job selections and competitive exams
CO4: Analyze the expectations of recruiters and answer common HR interview questions
CO5: Apply knowledge of phonetics, stress and intonation of English in everyday usage

Text Books

1. Sanjay Kumar and PushpLata; Communication Skills; Oxford University Press, 2012.
2. Sabina Pillai and Agna Fernandez; Soft Skills and Employability Skills;Cambridge University Press, 2018.
3. S.P. Dhanavel; English and Communication Skills for Students of Science and Engineering; Orient Blackswan, 2009
4. M. Ashraf Rizvi; Effective Technical Communication; Tata Mc Graw Hill Co. ltd, 2005.

Reference Books

1. Dr.M.Adithan; *Study Skills for Professional Students in Higher Education*; S.Chand& Co. Pvt., 2014.
2. Guy Brook Hart & Vanessa Jakeman; *Complete IELTS*: Cambridge University Press, 2014.
3. Vanessa Jakeman& Clare Mcdowell; *Action Plan for IELTS:* Cambridge University Press, 2006.
4. Guy Brook Hart; *Instant IELTS*;Cambridge University Press, 2004
5. Leo Jones; *Let's Talk 3*(second edition); Cambridge University Press.
6. Steven Gershon; *Present Yourself 2* (second edition); Cambridge University Press.
7. Charles Browne, Brent Culligan 7 Joseph Phillips; In Focus (level 2); Cambridge University Press.
8. S.P.Bakshi&Richa Sharma; *Descriptive General English;*Arihant Publications, 2012
9. Nutall J. C.; *Reading Comprehension*; Orient Blackswan
10. *www.cambridgeenglish.org/in/*
11. https://learnenglish.britishcouncil.org/en/english-grammar
12. https://www.rong-chang.com/

Mode of Evaluation: Continuous Internal Evaluation and End Semester Examination.

MCA I Year I Semester

25MDMCASC01 FRONTEND WEB DEVELOPMENT

L	T	P	C
1	0	2	2

Pre-requisite: None

Course Description:
To understand the basics of frontend web development including HTML, CSS and JavaScript. the students can update their knowledge on technologies. This will help the students to learn the complete set of process-like designing, development and deployment.

Course Objectives:
This course enables students to:

1. Understand the Fundamentals of HTML web pages.
2. Apply CSS Styling Techniques, and properties to style web pages effectively, including color manipulation, font styling, and element positioning.
3. Familiarize web page with validation using JavaScript
4. Understand different DOMs and event handling mechanisms.
5. Understand and practice embedded dynamic scripting on client-side Internet

UNIT I UI DESIGN 6 Hours

HTML Introduction - Basics- Elements- Attributes- Headings- Paragraph- Tables- Formatting- Links and Images - Lists- Blocks- Layout -Responsive - IFrames -Forms – Form Elements- Introduction to HTML5

1. Write an HTML program to format text using different fonts, colors, headings, and set a background color for the page.
2. Create an HTML table with at least 3 rows and 4 columns, including table headers and cell borders.
3. Design a Registration Form using HTML form elements such as text boxes, radio buttons, checkboxes, drop-down menus, and submit/reset buttons.

UNIT II CASCADING STYLE SHEET 6 Hours

Cascading Style Sheet (CSS3): The need for CSS – Basic syntax and structure Inline Styles – Embedding Style Sheets - Linking External Style Sheets - Introduction to CSS3 – Backgrounds - Manipulating text - Margins and Padding - Positioning using CSS - Responsive Web Design - Introduction to LESS/SASS

1. Write CSS rules to change fonts, background colors, and text colors of elements in an HTML page.
2. Create an HTML page containing a drop-down list of 5 countries. When a country is selected, display its capital beside the list styled with CSS (bold, color, font size). Use JavaScript to handle the selection.
3. Design a resume page and apply CSS3 properties for styling text, backgrounds, margins, and paddings.

UNIT III OVERVIEW OF JAVASCRIPT 6 Hours

Introduction - Core features - Data types and Variables - Operators, Expressions, and Conditional Statements (if, if-else, if-else-if-else), switch-case, loops, (while, for, do-while), Functions, Objects - Array, Date and Math Related Objects.

1. Write a JavaScript program to generate and display a random number between 1 and 100.

2. Write JavaScript programs to demonstrate the use of loops: while, for, and do-while. For example, calculate the sum of first N natural numbers.
3. Create a CGPA calculator using HTML, CSS, and JavaScript functions. The user should enter marks, and the program calculates and displays the CGPA.

UNIT IV DOM AND EVENT HANDLING 6 Hours

Document Object Model: getElementByid(), getElementByClassName(), getElementByName(), getElementByTagName(), JS innerHTML Propery - Event Handling - Controlling Windows & Frames and Documents - Form validations- Exception Handling.

1. Write JavaScript code to validate form inputs on a registration page. Validate fields like name (only alphabets), email (valid format), phone number (digits only), and password (minimum length).
2. Write a JavaScript program to demonstrate event handling: change text on button click, change color on mouse hover, and show an alert on form submission.
3. Write a JavaScript program to demonstrate exception handling using try-catch-finally blocks. For example, handle division by zero or invalid input errors gracefully

UNIT V ADVANCED FEATURES OF JAVASCRIPT 6 Hours

Browser Management and Media Management – Classes – Constructors – Object-Oriented Techniques in JavaScript – Object constructor and Prototyping - Sub classes and Super classes – Introduction to JSON – JSON Structure –Introduction to jQuery –Introduction to AJAX-Bootstrap - Bootstrap components.

1. Create a PAN Card validation form using Object-Oriented JavaScript. Ensure the last character of the PAN number is an alphabet. Also, take user's first name, last name, and other details as input.
2. Create a JSON structure representing a bookstore with categories and books. Validate this JSON using an online validator (e.g., jsonlint.com). Write JavaScript code to parse this JSON and display all books under the "Fiction" category.
3. Build a Single Page Application (SPA) for movie search:
   - Admin login page to upload movie trailer, poster, keywords, and details.
   - Use Bootstrap and jQuery for UI design.
   - Implement form submission using AJAX to update the movie list dynamically.

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Design a complete, well-organized website using HTML.

CO2: Design interactive websites and dynamically use CSS to optimize user experience

CO3: Develop the dynamic web pages using JAVA script

CO4: Create Dynamic webpages using DOMs and Form Validation.

CO5: Employ object-oriented programming principles.

Text Books

1. Deitel and Deitel and Nieto, Internet and World Wide Web - How to Program, Prentice Hall, 5th Edition, 2011.

2. Java Script for Programmers Paul J. Deitel, Deitel & Associates, Inc.Harvey M. Deitel, Deitel& Associates, Inc.

Reference Books

1. Stephen Wynkoop and John Burke "Running a Perfect Website", QUE, 2nd Edition,1999

2 Web Coding Bible, An Accelerated Course, Chong Lip Phang, 2015

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

MCA I Year II Semester

MCA I Year II Semester

25MDMCAEC05 SOFTWARE ENGINEERING AND DESIGN PATTERNS

L	T	P	C
3	0	2	4

Pre-requisite: None

Course Description:

This course introduces students to the principles of software engineering, design methodologies, and reusable design patterns. It covers SDLC models, requirements analysis, UML design, and software project management. A major emphasis is placed on the practical application of design patterns through lab exercises, mini-projects, and the use of CASE tools.

Course Objectives:

This course enables students able to

1. Understand the software development life cycle and project management fundamentals.
2. Model requirements and system design using UML and CASE tools.
3. Learn the classification and application of design patterns.
4. Implement creational, structural, and behavioral patterns in real-world projects.
5. Integrate software engineering processes with Agile methodologies and iterative development.

UNIT I SOFTWARE ENGINEERING FUNDAMENTALS 9 Hours

Software nature and principles, SDLC Models: Waterfall, Spiral, Incremental, Agile. Project management basics: Estimation, Scheduling, Risk Management.

UNIT II REQUIREMENTS & MODELING 9 Hours

Requirements Engineering: Elicitation, Analysis, SRS. Design Concepts: Modularity, Abstraction, Cohesion, Coupling. UML Diagrams: Class, Use case sequence collaboration activity, state, Component and deployment.

UNIT III INTRODUCTION TO DESIGN PATTERNS 9 Hours

Introduction to Design Patterns-,Classification of Design Patterns- catalogues of design patterns. Classification: Creational, Structural, Behavioral. Guidelines for selecting and applying patterns.

UNIT IV CREATIONAL & STRUCTURAL PATTERNS 9 Hours

Creational: Singleton, Factory, Builder, Prototype, Abstract Factory. Structural: Adapter, Composite, Decorator, Proxy, Facade, Bridge.

UNIT V BEHAVIORAL PATTERNS & CASE STUDIES 9 Hours

Behavioral: Observer, Strategy, Command, Template Method, Iterator. Refactoring and anti-patterns. **Case study**: E-commerce / Banking / Healthcare system with UML & Patterns.

List of Experiments **30 Hours**

1. Develop Software Requirement Specification (SRS) for a case study.
2. Draw UML diagrams: Use Case, Class, Sequence, State Chart, Activity.
3. Model interactions using StarUML or equivalent CASE tool.
4. Implement Creational Patterns (Singleton, Factory, Builder) in Java/Python.
5. Implement Structural Patterns (Adapter, Decorator, Composite).
6. Implement Behavioral Patterns (Observer, Strategy, Command).
7. Conduct Software Estimation & Scheduling using tools (e.g., COCOMO, Gantt charts).
8. Prepare test cases and demonstrate unit testing & integration testing.
9. Group activity: Refactor legacy code using patterns.
10. Mini Project: Develop a software system applying at least 3–4 design patterns, with UML modeling and testing reports.

Software requirements: StarUML,TestNG (Java), OpenProject, Online COCOMO calculators (web-based)

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Explain software engineering processes and project management.
CO2: Develop SRS and UML models for software systems.
CO3: Differentiate and classify design patterns.
CO4: Apply creational, structural, and behavioral patterns in software development.
CO5: Demonstrate testing, refactoring, and estimation skills in practical scenarios.

Text Books

1. Pressman, Roger S. *Software Engineering: A Practitioner's Approach*. 9th ed., McGraw-Hill Education, 2020.

2. Gamma, E., Helm, R., Johnson, R., Vlissides, J. – *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison Wesley. 1994.

Reference Books

1. Sommerville, Ian. *Software Engineering*. 10th ed., Pearson, 2016.

2. Larman, Craig. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*. 3rd ed., Pearson, 2005.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

MCA I Year II Semester

25MDMCAEC06 ARTIFICIAL INTELLIGENCE

L	T	P	C
3	0	2	4

Pre-requisite: None

Course Description:

The course aims to introduce the fundamentals of Artificial Intelligence, enable understanding and implementation of various AI problem-solving techniques, explore strategic decision-making through game theory and logical reasoning, and provide insights into the diverse real-world applications of AI across domains such as communication, automation, healthcare, and security.

Course Objectives:

This course enables students to

1. Understand the fundamentals of Artificial Intelligence.
2. Identify the problem-solving techniques of Artificial Intelligence.
3. Explain and implement various problem-solving techniques.
4. Explore strategic decision-making using game theory and logical reasoning techniques.
5. Gain insights into real-world applications of AI.

UNIT I BASICS OF AI 9 Hours

Human Intelligence, Artificial Intelligence – Human Intelligence vs AI - Evolution of AI – Types of AI - Components of Artificial Intelligence - Ethical and societal challenges of AI – **AI Tools:** Python, Jupyter Notebook, Google Colab - **AI Libraries**: Numpy, Pandas, Seaborn, Matplotlib.

UNIT II INTELLIGENT AGENTS 9 Hours

Intelligent agents: Agents and Environments – Rationality, Omniscience, learning, and autonomy- PEAS– The Structure of Agents - Properties of task environments -Types of agents – real world problems: toys problem, 8 puzzle, 8 – Queen problem.

UNIT III PROBLEM SOLVING IN AI AND GAME THEORY 9 Hours

Solving Problems by Searching: Breadth-first search, Depth-first search, Bidirectional search - Informed (Heuristic) Search Strategies: Best First Search, A* Search, AO* Search - Hill Climbing Search - Game Solving Algorithms -Min-Max algorithms, Alpha-Beta Pruning.

UNIT IV KNOWLEDGE REPRESENTATION 9 Hours

Knowledge Representation: Logical Agents - knowledge-based agents - Semantic Networks, Frames, and Ontologies - The wumpus world - Forward Chaining - Backward Chaining.

UNIT V AI APPLICATIONS & RESPONSIVE AI 9 Hours

Application of AI: Natural Language Processing: Text Classification, Sentiment Analysis - Computer Vision: Object Detection & object recognition - Chatbots and Virtual Assistants - Healthcare, Finance, Agriculture, Education. Responsible AI-Fairness-Transparency-Accountability-Privacy-Safety & Security-Inclusiveness-Human-Centered

Experiments **30 Hours**

1. Implement a simple reflex agent for a vacuum cleaner environment that cleans dirty tiles.
2. Represent the 8-puzzle problem and write a program to generate possible moves from a state
3. Implement Breadth-First Search (BFS) to traverse a given graph and print nodes in order. Implement Depth-First Search (DFS) and compare its traversal output with BFS on the same graph.
4. Write a program implementing A* search on a grid to find the shortest path from start to goal.
5. Optimize the Min-Max algorithm using Alpha-Beta pruning and compare the performance.
6. Write a backward chaining algorithm to answer queries based on given facts and rules.
7. Build a text classification program using Naive Bayes classifier on a movie review dataset.
8. Perform sentiment analysis on tweets using a Python NLP library and visualize results.
9. Use OpenCV to detect faces or objects in images and display bounding boxes.
10. Develop a simple chatbot that responds to greetings and FAQs based on predefined patterns.

Software Requirements: Required AI Tools: Jupyter Notebook, Google Colab.

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Describe the fundamental concepts, history, and scope of Artificial Intelligence.

CO2: Apply uninformed and informed search strategies to solve AI-based problems.

CO3: Implement game-playing algorithms and logic-based reasoning using propositional and first-order logic.

CO4: Analyze the architecture and functionalities of AI systems such as chatbots, expert systems, and recommendation engines.

CO5: Evaluate the role and impact of AI applications in domains like healthcare, autonomous driving, and cyber security.

Text Book(s)

1. **Stuart Russell and Peter Norvig**, *Artificial Intelligence: A Modern Approach*, 4th Edition, Pearson Education, 2021.

Reference Books

1. **Suresh Samudrala**, *Introduction to Artificial Intelligence*, Wiley, **2023**.
2. **Tom Taulli**, *Artificial Intelligence Basics: A Non-Technical Introduction*, Apress, **2020**.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

MCA I Year II Semester

25MDMCAEC07 ADVANCED WEB TECHNOLOGIES AND MERN STACK

L	T	P	C
3	0	2	4

Pre-requisite: None

Course Description:

This course introduces web designing and development methodologies using the front-end development languages such as JavaScript and Mern Stack, React Js, Node JS and Express. Also, this course gives the idea of web development process using Mongo DB

Course Objectives:

This course enables students to:

1. Understand JavaScript fundamentals and MERN architecture.
2. Apply React to build interactive UIs with components and state.
3. Analyze and develop REST APIs using Node.js and Express.
4. Implement MongoDB for data storage and retrieval.
5. Optimize full-stack application performance through best practices.

UNIT I JAVASCRIPT AND BASICS OF MERN STACK 9 Hours

JavaScript Fundamentals - Objects - Generators, advanced iteration - Modules - DOM tree - Node properties - browser events - Event delegation - UI Events -Forms, controls - Document and resource loading-Event loop: microtasks and macrotasks - MERN Components- Need for MERN - Server-Less Hello World - Server Setup - nvm - Node.js - npm

UNIT II REACT JS 9 Hours

React Introduction - React ES6 - React Render HTML - React JSX - Components -React Classes - Composing Components - Passing Data - Dynamic Composition - React state - setting State - Async State Initialization - Event Handling Communicating from Child to Parent - Stateless Components - Designing components- React Forms - React CSS - React SaaS

UNIT III NODE.JS AND EXPRESS 9 Hours

Node.js basics - Local and Export Modules - Node Package Manager - Node.js web server - Node.js File system - Node Inspector - Node.js EventEmitter - Frameworks for Node.js - Express.js Web App - Serving static Resource - Node.js Data Access - Express REST APIs - REST - Resource Based - HTTP Methods as Actions - JSON- Express - Routing - Handler Function - Middleware - The List API

UNIT IV MONGODB 9 Hours

MongoDB - MongoDB Basics - Documents - Collections - Query Language - Installation - The mongo Shell - Schema Initialization - MongoDB Node.js Driver - Reading from MongoDB - Writing to MongoDB - CRUD operations - projections - Indexing - Aggregaton - Replication - Sharding - Creating backup – Deployment

UNIT V ADVANCED FEATURES 9 Hours

Modularization and Webpack - Routing with React Router - Forms - More Filters in the List API - UI Components - Update API - Delete API - React-Bootstrap - Bootstrap Installation - Navigation - Table and Panel - Forms - Alerts - Modals -Server Rendering - Basic Server Rendering - Handling State - Pagination - Session Handling

List of Experiments **30 Hours**

1. Write a program to create a simple webpage using HTML.
2. Write a program to build a Chat module using HTML CSS and JavaScript?
3. Build a simple form with validation (check if email is valid before submission).
4. Write a program to create an array of 5 cities and perform the following operations: Log the total number of cities. Add a new city at the end. Remove the first city. Find and log the index of a specific city.
5. Write a program to create a simple calculator Application using React JS
6. Write a program to create a Simple Login form using React JS
7. Write a program to create a counter using ReactJS
8. Create a NodeJS server that serves static HTML and CSS files to the user without using Express.
9. Create a NodeJS server using Express that stores data from a form as a JSON file and displays it in another page. The redirect page should be prepared using Handlebars.
10. Install the MongoDB driver for Node.js. Create a NodeJS server using Express that creates, reads, updates and deletes students' details and stores them in MongoDB database. The information about the user should be obtained from a HTML form
11. Create a simple Sign up and Login mechanism and authenticate the user using cookies. The user information can be stored in MongoDB and the server should be built using NodeJS
12. Create an Express API with a /products endpoint to fetch all products. Use fetch in React to call the /products endpoint and display the list of products. Add a POST /product
13. Write a program to create a voting application using ReactJS

Software requirements: Notepad, Node.js (includes npm), MongoDB Community Server (local) or MongoDB Atlas (cloud)

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Explain core JavaScript concepts and MERN stack architecture
CO2: Develop interactive web interfaces using React components and state management
CO3: Design and implement RESTful APIs using Node.js and Express
CO4: Perform database operations using MongoDB's document-based structure
CO5: Analyze and optimize performance in full-stack MERN applications

Text Books

1. Jon Duckett, "JavaScript & jQuery: Interactive Front-End Web Development", Wiley, 2014.
2. Vasan Subramanian, Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React, and Node. Apress, 2019.

Reference Books

1. Mastering Javascript, VedAntani, PACKT publishing, 2019

2. Node JS Web Development, David Herron, PACKT publishing, 2021

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

MCA I Year II Semester

25MDMCALC02 COMPETITIVE CODING LABORATORY

L	T	P	C
0	0	2	1

Pre-requisite: None

Course Description:
This course is designed to strengthen problem-solving skills and logical thinking through competitive programming practices. Students will learn to apply fundamental programming concepts, data structures, algorithms, and optimization techniques to solve computational problems effectively. With hands-on practice on coding platforms like HackerRank and LeetCode, the course builds the foundation for technical interviews, coding contests, and real-world problem solving.

Course Objectives:
This course enables students to:

1. Analyze algorithm complexity and apply mathematical & bit-level concepts for efficient solutions.
2. Implement array and string algorithms using key techniques
3. Utilize linear data structures and employ hashing for data analysis.
4. Design recursive algorithms and evaluate Heaps & Graphs for prioritization and cycles.
5. Evaluate problems to select and synthesize solutions using advanced paradigms

UNIT I FOUNDATIONS OF COMPETITIVE PROGRAMMING 12 Hours
Competitive Programming, Setting up a coding environment, Time and Space Complexity Analysis, Mathematics for Programming: Prime numbers, Modular arithmetic, Number theory basics (GCD - Euclidean algorithm, LCM, Combinatorics (nCr, nPr). Bit Manipulation: Basic operators (AND, OR, XOR, NOT, Left/Right Shift), Solving problems using bit masks. Gray Codes.

UNIT II ARRAYS AND STRING TECHNIQUES 12 Hours
Array: Sliding Window (Fixed and Variable size) problems, Two-Pointer technique (for sorting, pairing, and partitioning), Kadane's Algorithm**, String Manipulation:** String building, pattern matching, searching, Character encoding, palindrome problems**.**

UNIT III LINEAR DATA STRUCTURES & HASHING 12 Hours
Stack & Queue: Balanced Parenthesis, Stock span problem**,** generate numbers with given digits**, Linked Lists:** Fast and Slow Pointer technique (cycle detection, finding middle). reversal, reordering, and merging lists. **Hashing:** frequency counting, duplicate detection, subarray sum equals K.

UNIT IV RECURSION, TREES, HEAPS & GRAPHS TECHNIQUES 12 Hours
Recursion: Principles of recursion, factorial, Fibonacci, permutations, Tree Traversals, Subset Sum, **Heaps (Priority Queues**): Top K elements, Kth largest/smallest, merging sorted lists, Graphs: Cycle detection in directed and undirected graphs.

UNIT V ADVANCED ALGORITHMS & PROBLEM-SOLVING STRATEGIES 12 Hours
Greedy Algorithms: activity selection, Huffman coding, interval scheduling. **Dynamic Programming (DP):** Climbing Stair, Longest Common Subsequence, Decode Ways, Unique Paths, House Robber,

Backtracking: N-Queens, Sudoku solver, **Subsets, Brach and Bound:** Traveling Salesperson Problem (TSP). Job Assignment Problem

Course Outcomes:

Upon successful completion of the course, students will be able to
CO1: Analyze algorithm complexity and apply math/bit manipulation.
CO2: Implement array/string techniques to solve problems.
CO3: Utilize non-linear structures and employ hashing.
CO4: Design recursive algorithms and use heaps.
CO5: Evaluate problems and construct solutions with advanced algorithms

Text Books

1. Laakmann McDowell, Gayle. *Cracking the Coding Interview: 189 Programming Questions and Solutions.* 6th ed., CareerCup, 2015.

2. Arefin, Ahmed Shamsul. "Art of Programming Contest." *C Programming Tutorials, Data Struc-tures and Algorithms. ISBN* (2006): 984-32.

3. Halim, Steven, and Felix Halim. *Competitive Programming 4: The Lower Bound of Programming Contests in the 2020s.* 4th ed., Lulu.com, 2020.

Reference Books

1. Cormen, Thomas H., et al. *Introduction to algorithms*. MIT press, 2022.

2 Skiena, Steven S. *The Algorithm Design Manual.* 3rd ed., Springer, 2020.

3. Recommended online Platform:

1 **https://leetcode.com/**
2 **https://www.geeksforgeeks.org**
3 **https://www.hackerrank.com**
4 **https://codeforces.com**
5 **https://www.codechef.com**
6 **https://skillrack.com/**

Mode of Evaluation: Continuous Internal Evaluation (Record) and End Semester Examination.

MCA I Year II Semester

25MDMCASC02 MOBILE APPLICATION DEVELOPMENT

L	T	P	C
1	0	2	2

Pre-requisite: None

Course Description:

This course offers a thorough foundation in Android development, teaching core concepts like architecture, UI design, threading, and storage systems. Learners will create interactive apps using Recycler View, Material Design, and Intents while exploring advanced features like sensors, databases, and location services. Practical Android Studio exercises and real-world projects build hands-on skills for developing professional mobile applications.

Course Objectives:

This course enables students to

1. Understand the architecture, evolution, and life cycle of Android applications,
2. Design and implement responsive user interfaces using Android UI components
3. Develop robust Android applications by applying threading models
4. Utilize Android's storage mechanisms
5. Integrate device sensors and location-based services to build interactive and context-aware mobile applications.

UNIT I INTRODUCTION ABOUT ANDROID TOOLS 6 Hours

Android Overview – History – Android Versions - Android Stack: Linux, Native Layer and Hardware Abstraction Layer (HAL) –Installing the Android SDK - Anatomy of an Android Project.

1. Create a new Android Project and run a simple "Hello World" app on an emulator or physical device.
2. Design a layout with TextView, ImageView, Button, and EditText.

UNIT II BUILDING USER INTERFACE 6 Hours

Input Components – Text View – Image View – List View and Alert Dialogues – Menus: Popup, Options and Context Menus – Screen Navigation through App Bar – Recycler View –Usage of Intends – Creation of Indents with example program – Lists and Adapters – Types of Adapters.

1. Design a screen with ImageView showing an image resource.
2. Add an Alert Dialog with OK/Cancel buttons in response to a button click
3. Implement navigation between two activities using intents.

UNIT III APPLICATION DESIGN 6 Hours

Threading in Android – AsyncTask – Loaders – AsyncTask Loader –Broadcast Receivers: Custom Broadcasts - Boot Receiver - Alarms and system services –Services: Services Life Cycle – Intent Service – Implementing Intent Service – Notifications: Managing Notifications.

1. Implement an Intent Service and demonstrate its use case.
2. Implement AsyncTask to perform a background operation with UI update.

UNIT IV ANDROID FILE SYSTEM 6 Hours

Android File systems and Files - Action Bar: Preferences and Action Bar - Shared Preferences – App Settings - Databases on Android - SQLite - Content Providers: Overview – Role of Content Providers - Content Provider Example Program – Content Resolver .

1. Design a simple app to create, read, update, and delete records in SQLite database.
2. Build a custom Content Provider and query it via Content Resolver.

UNIT V APPLICATION DEVELOPMENT AND SENSOR 6 Hours

App Widgets: Creation of Application Widgets - Interaction and Animation: Live Wallpaper and Handlers - Sensors: Sensor API in Android - Motion Sensor, Position Sensor, Environmental Sensor, orientation Sensors, Sensor Examples

1. Create a simple App Widget that updates content periodically.
2. Implement a basic animation using Handlers.
3. Develop an app that listens to accelerometer sensor data and displays it in real-time.

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Understand the basics of mobile application development and Install android.
CO2: Develop a UI for mobile applications.
CO3: Work with Broadcast Receivers and Services.
CO4: Create Database in Android.
CO5: Build widgets, Wall papers for an android application using sensors

Text Books

1. *Dawn Griffiths, David Griffiths* **"Head First Android Development"** *2nd Edition, 2021,*

2. *Ian Darwin* **"Android Cookbook"** *2nd Edition, O'Reilly, 2022,*

Reference Books

1. *Zigurd Mednieks, Laird Dornin* **"Programming Android"** *2nd Edition, O'Reilly, 2022, , Blake Meike*

2. *Varun Nagpal* **"Android Sensor Programming By Example"** *Packt, 2023,*

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

MCA II Year I Semester

MCA II Year I Semester

25MDMCATC01 INTERNET OF THINGS

L	T	P	C
3	0	0	3

Pre-requisite: None

Course Description:

The course explores IoT communication protocols, cloud platforms, data collection, storage, and analytics used in IoT environments. Security and privacy issues in IoT systems are discussed along with methods to protect connected devices and networks. In addition, students learn about real-world IoT applications such as smart homes, smart cities, healthcare, agriculture, and industrial automation, enabling them to understand how IoT technologies are used to build intelligent and connected systems.

Course Objectives:

This course enables students

1. To understand the fundamental concepts, architecture, and components of the Internet of Things
2. To learn about IoT hardware such as sensors, actuators, and embedded systems used in connected devices.
3. To study IoT communication technologies and protocols used for device connectivity.
4. To explore IoT platforms, data management, and cloud integration for IoT applications.
5. To Analyze IoT security challenges and analyze real-world IoT applications in various domains

UNIT I INTRODUCTION TO IOT 9 Hours

Introduction to Internet of Things, definition and characteristics of IoT, evolution of IoT, IoT architecture, physical design of IoT, logical design of IoT, IoT enabling technologies, IoT communication models, IoT communication APIs.

UNIT II IOT HARDWARE AND COMMUNICATION TECHNOLOGIES 9 Hours

Overview of IoT hardware components, sensors and actuators, microcontrollers and embedded systems, introduction to Arduino and Raspberry Pi, wireless communication technologies including Wi-Fi, Bluetooth, Zigbee, RFID, NFC, and LoRa, networking technologies for IoT, introduction to IoT protocols such as IPv6, MQTT, and CoAP.

UNIT III IOT PLATFORMS AND DATA MANAGEMENT 9 Hours

IoT platforms and cloud integration, IoT middleware, data collection and storage in IoT systems, cloud computing for IoT, edge computing, data analytics in IoT, real-time monitoring systems, and data visualization techniques for IoT applications.

UNIT IV IOT SECURITY AND PRIVACY 9 Hours

Security issues in IoT systems, IoT threat models, and authentication and authorization mechanisms, encryption techniques for IoT devices, Zero trust Architecture for IoT,privacy issues in IoT, secure communication protocols, device security, IoT vulnerabilities and threat analysis

UNIT V IOT APPLICATIONS AND CASE STUDIES 9 Hours

Design and development of IoT applications, smart home automation systems, smart healthcare systems, smart wearables, smart agriculture and environmental monitoring, smart transportation systems, smart cities, industrial IoT (IIoT), real-world IoT case studies, and future trends and challenges in IoT.

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Understand the fundamental concepts, architecture, and enabling technologies of the Internet of Things (IoT).
CO2: Identify and use IoT hardware components such as sensors, actuators, and microcontrollers.
CO3: Analyze IoT communication technologies and protocols used for device connectivity.
CO4: Apply IoT platforms and data management techniques for developing IoT applications.
CO5: Evaluate security challenges in IoT systems and understand real-world IoT.

Text Books

1. Ahmed Banafa, Introduction to Internet of Things (IoT), River Publishers, 2023.
2. Raj Kamal, Internet of Things: Architecture and Design Principles, 2nd Edition, McGraw-Hill Education, 2022
3. Arshdeep Bahga and Vijay Madisetti, Internet of Things: A Hands-On Approach, Universities Press, 2015.

Reference Books

1. Preeti Mishra et al., Internet of Things Security: Attacks, Tools, Techniques and Challenges, Elsevier, 2025.
2 Srinivasa K. G., Siddesh G. M., and Hanumantha Raju R., Internet of Things, Cengage Learning, 2018.
3 Samuel Greengard, *The Internet of Things*, MIT Press, **2015**.

Mode of Evaluation: Assignments, Mid Term Tests and End Semester Examination.

MCA II Year I Semester

25MDMCATC02 GENERATIVE AI

L	T	P	C
3	0	0	3

Pre-requisite: 25MDMCAEC06

Course Description:

This course provides a comprehensive introduction to Generative Artificial Intelligence (Generative AI) and Large Language Models (LLMs). It covers foundational and the practical use of leading AI platforms such as OpenAI, Google Gemini, Meta LLaMA, and Hugging Face. Students will gain hands-on experience in using APIs and AI tools to design and develop real-world applications, preparing them for emerging roles in AI-driven industries.

Course Objectives:

This course enables students to :

1. Understand the core concepts, techniques, and applications of Generative AI.
2. Develop skills for working with Large Language Models (LLMs).
3. Implement AI-based solutions using OpenAI APIs.
4. Analyze and compare the features and capabilities of Google Gemini and Meta LLaMA models
5. Utilize the Hugging Face platform to access, implement, and deploy pre-trained AI models.

UNIT I BASICS OF GENERATIVE AI 9 Hours

Introduction to Artificial Intelligence and Generative AI - Language Modeling concepts – Transformer architecture and attention mechanism – Evolution of Transformer models to GPT series -Overview of Generative AI technologies – Introduction to Generative Adversarial Networks (GAN)–Encoder-based models – Retrieval-Augmented Generation (RAG) models – Ethical issues and technical challenges in Generative AI.

UNIT II LARGE LANGUAGE MODELS 9 Hours

Overview of Large Language Models (LLMs) – Architecture overview of GPT models – Conceptual understanding of GPT-3 and GPT-4 – Capabilities of LLMs in generating text and code – Introduction to multimodal generation (text-to-image concepts) - Limitations of LLMs - Case Study: Implementation of an LLM-Based Academic Content Generator.

UNIT III OPENAI PLATFORM AND API INTEGRATION 9 Hours

Introduction to the OpenAI platform and ecosystem –Accessing and utilizing OpenAI APIs – API key generation, authentication, and configuration – Making API requests for text generation and chat-based applications – GPT models for text and code generation. Case study: AI-Based Healthcare Assistant Using OpenAI APIs.

UNIT IV GOOGLE GEMINI AND LLAMA 9 Hours

Introduction to Google Gemini and its AI ecosystem- Overview of Gemini model variants and capabilities – Generating and managing Gemini API keys - Accessing and utilizing Gemini APIs – Building AI-powered chatbot applications using Gemini- Text generation using LLaMA models – Comparative analysis of Gemini and LLaMA models for real-world applications.

UNIT V HUGGING FACE ECOSYSTEMS 9 Hours

Introduction to the Hugging Face ecosystem – Overview of the Transformers library – Understanding models and tokenizers – Using pre-trained models for NLP tasks – Sentiment Analysis using transformer pipelines – Text Generation using pre-trained models – Named Entity Recognition (NER) -Text Summarization techniques. Risk assessment, monitoring, and incident response for AI-enabled applications – ethical, legal, and governance issues in AI security.

Course Outcomes:

Upon successful completion of the course, students will be able to
CO1: Understand the fundamentals of Generative AI and transformer models.
CO2: Apply Large Language Models (LLMs) for generating text, code, and images.
CO3: Demonstrate knowledge of OpenAI's ecosystem and API integration techniques.
CO4: Explain the architecture and text-generation capabilities of LLaMA and related models.
CO5: Evaluate and utilize Hugging Face models for various NLP tasks.

Text Books

1. James Phoenix & Mike Taylor, Prompt Engineering for Generative AI, O'Reilly Media, 2024.
2. Elvis Saravia, Prompt Engineering Guide, 2023 (Latest online reference edition).
3. Lewis Tunstall, Leandro von Werra, Thomas Wolf, Natural Language Processing with Transformers, O'Reilly Media, 2022.

Reference Books

1. Denis Rothman, Transformers for Natural Language Processing, 3rd Edition, Packt Publishing, 2024.

2 Jay Alammar & Maarten Grootendorst, Hands-On Large Language Models, O'Reilly Media, 2024.

3 Andriy Burkov, The Hundred-Page Machine Learning Book, Updated Edition, 2023.

Mode of Evaluation: Assignments, Mid Term Tests and End Semester Examination.

MCA II Year I Semester

25MDCOMTC01 RESEARCH METHODOLOGY AND IPR

L	T	P	C
2	0	0	2

Pre-requisite: None

Course Description:

This course aims has been meticulously designed to provide a comprehensive understanding of the essential aspects of research methodology and the critical realm of Intellectual Property Rights (IPR).

Course Objectives:

This course enables students to :

1. Identify and discuss the concepts and finding the criteria of good research.
2. Find sources to data collection.
3. Understand the methodologies of data analysis and reporting.
4. Understand the basics of Intellectual Property Rights.
5. Know the process of patents and filling.

UNIT I FOUNDATIONS OF RESEARCH 9 Hours

Meaning, Objectives, Motivation of Research, Concept of Theory, deductive and inductive theory ,various types of Research, Overview of research process and design, Criteria of Good Research. Research Tools.

UNIT II DATA COLLECTION AND SOURCES 9 Hours

Measurements, Levels of Measurements, Questionnaires and Instruments, Sampling and methods, Qualitative and Quantitative Data - Preparing, Exploring, examining and displaying of data.

UNIT III TESTING OF HYPOTHESES 9 Hours

Hypothesis, Procedure for Hypothesis Testing, Hypothesis Testing of Means, Proportions, Correlation Coefficients. Analysis of Variance (ANOVA).

UNIT IV INTELLECTUAL PROPERTY RIGHTS 9 Hours

Intellectual Property – The concept of IPR, Evolution and development of IPR, Trade secrets, utility Models, Role of WIPO and WTO in IPR establishments, Common rules of IPR practices, Types and Features of IPR Agreement, Trademark, Functions of UNESCO in IPR maintenance..

UNIT V PATENTS 12 Hours

Patents – objectives and benefits of patent, Concept, features of patent, Inventive step, Specification, Types of patent application, process E-filling, Examination of patent, Grant of patent, Revocation, Equitable Assignments, Licenses, Licensing of related patents, patent agents, Registration of patent agents.

Course Outcomes:

Upon successful completion of the course, students will be able to
CO1: Formulate and Design research problem.
CO2: Understand and Comprehend the Data Collection Methods.
CO3: Perform Data analysis and acquire Insights.
CO4: Understand IPR and follow research ethics.
CO5: Evaluate the Patent Drafting and Filing in research areas.

Text Books

1. Kothari C R, Gaurav Garg, "Research Methodology", New Age International Publishers, 5th Edition, 2023.
2. Research Design: Qualitative, Quantitative, and Mixed Methods Approaches, John W. Creswell 2022.
3. The Craft of Research, New Edition, Wayne C. Booth, Gregory G. Colomb, Joseph M. Williams 2021.

Reference Books

1. The Institute of Company Secretaries of India, Statutory body under an Act of parliament, "Professional Programme Intellectual Property Rights, Law and practice", September 2013.
2 Cooper Donald R, Schindler Pamela S and Sharma JK, "Business Research Methods", Tata McGraw Hill Education, 11e (2012).
3 Catherine J. Holland, "Intellectual property: Patents, Trademarks, Copyrights, Trade Secrets", Entrepreneur Press, 2007.

Mode of Evaluation: Assignments, Mid Term Tests and End Semester Examination.

MCA II Year I Semester

25MDMCASC03 AI-BASED SOFTWARE DEVELOPMENT

L	T	P	C
1	0	2	2

Pre-requisite: 25MDMCAEC06

Course Description:

This course introduces the concepts and practices of AI-Based Software Development, focusing on the use of Artificial Intelligence throughout the Software Development Life Cycle (SDLC). It covers AI fundamentals, Generative AI, Large Language Models (LLMs), AI-assisted software design, coding, testing, deployment, and maintenance. The course also emphasizes responsible AI, security, ethics, and hands-on experience with modern AI tools and frameworks for developing intelligent software applications.

Course Objectives:

This course enables students to:

1. Understand the fundamentals of Artificial Intelligence, Machine Learning, Generative AI, and their role in modern software development.
2. Apply AI tools and techniques across various phases of the Software Development Life Cycle (SDLC), including requirements engineering, design, coding, testing, and deployment.
3. Develop AI-assisted software solutions using Large Language Models (LLMs), AI coding assistants, and intelligent software engineering practices.
4. Evaluate ethical, security, explainability, and governance aspects while developing AI-enabled software systems.
5. Design, implement, and deploy AI-based applications using appropriate frameworks, cloud platforms, and DevOps practices.

UNIT I INTRODUCTION TO AI-BASED SOFTWARE DEVELOPMENT 6 Hours

Introduction to Artificial Intelligence (AI), Machine Learning (ML), Generative AI, Large Language Models (LLMs), AI agents, AI-assisted Software Development Life Cycle (AI-SDLC), AI coding assistants, prompt engineering, ethical and responsible AI practices.

1. Explore AI coding assistants (ChatGPT, GitHub Copilot, Gemini, or Claude) to generate simple programs and compare the generated code with manually written code.
2. Design effective prompts for different software development tasks (code generation, debugging, and documentation) and evaluate the quality of AI-generated responses.

UNIT II AI-ASSISTED SOFTWARE DESIGN 6 Hours

AI for requirements engineering, user story generation, software specification, AI-assisted UML and architecture design, design patterns, database schema generation, prototype development, documentation automation.

3. Generate Software Requirement Specification (SRS) and User Stories for a Library Management System using an AI assistant.
4. Create UML diagrams (Use Case, Class, and Sequence) and database schema for a Student Management System using AI-assisted design tools.

UNIT III AI-POWERED CODE GENERATION 6 Hours

AI-assisted programming, code generation, code completion, debugging, code refactoring, API and SQL generation, secure coding, Git and GitHub integration, AI-assisted documentation.

5. Develop a CRUD application using AI-generated code and integrate version control using Git and GitHub.
6. Use AI tools to debug, refactor, optimize, and document an existing software application, comparing code quality before and after optimization.

UNIT IV AI IN SOFTWARE TESTING, DEVOPS, AND DEPLOYMENT 6 Hours

AI-based software testing, automated test case generation, code review, static analysis, defect prediction, CI/CD, AI in DevOps, performance monitoring, software maintenance.

7. Generate automated unit test cases using AI tools and measure code coverage for a sample application.
8. Configure a simple CI/CD pipeline using GitHub Actions or GitLab CI and perform AI-assisted code review and static code analysis.

UNIT V ADVANCED AI IN SOFTWARE ENGINEERING 6 Hours

Retrieval-Augmented Generation (RAG), AI agents, full-stack AI development, low-code/no-code platforms, AI governance, cybersecurity, AI-based project management, industry case studies, future trends in autonomous software engineering.

9. Develop a Retrieval-Augmented Generation (RAG) application using an LLM and a vector database for document-based question answering.
10. Build an AI-powered software engineering assistant (agent) capable of generating code, documentation, and test cases for a selected software project.

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Explain the concepts of AI, Machine Learning, Generative AI, Large Language Models, and AI-assisted Software Development Life Cycle.

CO2: Apply AI techniques and intelligent tools for requirements engineering, software design, coding, documentation, and project planning.

CO3: Utilize AI-powered development environments, coding assistants, testing frameworks, and debugging tools to improve software quality and productivity.

CO4: Analyze ethical, security, explainability, and governance challenges associated with AI-enabled software systems and propose responsible AI solutions.

CO5: Design, develop, evaluate, and deploy AI-based software applications using modern AI frameworks, APIs, cloud services, and MLOps/DevOps practices.

Text Books

1. Generative AI with Python and TensorFlow – Amit Bahree, Packt Publishing, 2024.
2. Hands-On Large Language Models – Jay Alammar and Maarten Grootendorst, O'Reilly Media, 2024.
3. Artificial Intelligence: A Modern Approach – Stuart Russell and Peter Norvig, Pearson, 4th Edition, 2021.

Reference Books

1. Prompt Engineering for Generative AI – James Phoenix and Mike Taylor, O'Reilly Media, 2024.
2 Building LLM Powered Applications – Valentino Zocca, O'Reilly Media, 2024.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination

Discipline Electives
(AI and ML Stream)

Discipline Elective – I (AI and ML Stream) **MCA I Year II Semester**

25MDMCADC01 DATA ANALYTICS AND VISUALIZATION

L	T	P	C
3	0	2	4

Pre-requisite

Course Description:

This syllabus provides a comprehensive introduction to Data Science, covering fundamental concepts, tools, and applications. Students will learn data cleaning, analysis, and visualization techniques. The course teaches machine learning models such as regression, classification, and clustering, along with ways to evaluate their performance. It also includes advanced visualization tools like Tableau and Power BI, as well as real-world applications.

Course Objectives:

This course enables students to

1. Define data science concepts and demonstrate Python libraries for data analysis
2. Apply EDA techniques and perform data preprocessing including feature engineering and dimensionality reduction.
3. Compare machine learning models and evaluate their performance using appropriate metrics.
4. Design effective visualizations using Tableau/Power BI and interpret geospatial data.
5. Predict trends using time series analysis and relate data science to real-world applications.

UNIT I INTRODUCTION TO DATA ANALYTICS 9 Hours

Definition and scope of data analytics- Data Analytics Lifecycle- Types of Data Analytics-Data sources and types - Data Acquisition and Understanding - Introduction to Python libraries for data analysis (NumPy, Pandas, Matplotlib, Seaborn, Plotly, GeoPandas)

UNIT II EXPLORATORY DATA ANALYSIS 9 Hours

Descriptive statistics – Handling missing data, duplicate and Outlier Detection - Data Integration-Joining and merging datasets -Data aggregation and - Data transformation: Scaling-Encoding-Feature extraction and selection - Dimensionality Reduction (PCA and LDA).

UNIT III DATA VISUALIZATION 9 Hours

Gestalt Principles - best practices and design considerations- **Visualizing Univariate Distributions**: box plots, violin plots, histograms, **Visualizing Bivariate Relationships**: Scatter plots, 2D Density plots. **Visualizing Categorical Data:** Bar plots, Heatmaps, **Visualizing Multivariate Data**: Pair plots, Scatter plots– Visualizing geographic data and spatial relationships

UNIT IV VISUALIZATION TOOLS 9 Hours

Tableau: Overview & Architecture-tableau interface & Connection-Charts-reports-Filters-**Calculations** – Dashboard & Stories

POWER Bi- Foundation & Data Ingestion Data Transformation (Power Query)- Data Modeling-DAX- Charts- Slicers- Formatting- Dashboard-publish.

UNIT V TIME SERIES ANALYTICS & APPLICATIONS 9 Hours

Time Series Analysis and Forecasting: Introduction to time series data – Time Series Forecasting methods (**Naïve, AR, MA, ARMA, ARIMA, Exponential Smoothing**).

Application of data science: Healthcare-Predictive-Analytics, Retail-Recommendation-Systems, Social-Media-Sentiment-Analysis.

List of Experiments 30 Hours

1. Write a program to generate synthetic data (e.g., sales trends) using NumPy arrays and compute descriptive statistics on the dataset.
2. Load a dataset (e.g., Iris Dataset), clean column names, and filter rows based on conditions.
3. Write a Python program to perform EDA by detecting and handling missing values, outliers, and inconsistencies in a DataFrame.
4. Merge two datasets using inner/left joins and concatenate a third dataset vertically using Pandas merge/join, pivot_table and groupby operations.
   Data Sets: orders.csv (order_id, customer_id, order_date), customers.csv (customer_id, name, email), orders_new.csv (order_id, customer_id, order_date, product_id, quantity, total_price, payment_method).
5. Preprocess the Titanic dataset by scaling numeric features (Age, Fare), encoding categorical variables (Sex, Embarked), and creating new features (Family-Size, Title from Name).
6. Apply PCA to reduce the Iris dataset to 2 components, visualize the results, and compare their cluster separability.
7. Create a dataset and visualize it with different customization options, such as colors, labels, titles, and annotations, to enhance the visual appeal of the following plots by using matplotlib, seaborn, and plotly library.
   1. bar chart (ii) pie chart (iii) scatter plot and (iv) line plot (v) violin
8. Visualize Indian state populations and major city locations on an interactive map using GeoPandas in Python.
9. Create a Tableau dashboard with a map, bar chart, filters, and trend line using the Superstore dataset.
10. Using the Superstore data Build a Power BI/ Excel report with: A slicer for region, Matrix of sales by sub-category, Tooltip showing profit ratio
11. Implement the visualization of time series forecasting methods by using Air Passengers dataset: evaluate and plot results.

Minor Projects

1. Diabetes risk prediction model (**Data Set**: Pima Indians dataset)
2. Movie recommendation system (**Data Set**: MovieLens dataset)
3. Smart Home Energy Consumption Predictor (**Data set***:* UK Power Networks)

Required Tools: Jupyter Notebook, Google Colab.

AI Libraries: Numpy, Pandas, Seaborn, Matplotlib and GeoPandas.

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Understand Data Science fundamentals and use Python libraries for basic data analysis.
CO2: Execute data cleaning, transformation, and feature engineering to optimize dataset quality.
CO3: Build and assess regression, classification, and clustering models.
CO4: Design interactive dashboards and geospatial visualizations.
CO5: Analyze time series data and evaluate real-world applications.

Text Books

1. VanderPlas, Jake. *Python data science handbook: Essential tools for working with data*. " O'Reilly Media, Inc.", 2016.

2. Murray, Scott. *Interactive data visualization for the web: An introduction to designing with D3*. " O'Reilly Media, Inc.", 2017.

3 Hyndman, Rob J., and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

Reference Books

1. Wagh, Sanjeev J., Manisha S. Bhende, and Anuradha D. Thakare. *Fundamentals of data science*. Chapman and Hall/CRC, 2021.

2. Grus, Joel. *Data science from scratch: first principles with python*. O'Reilly Media, 2019.

3. Larose, Chantal D., and Daniel T. Larose. *Data science using Python and R*. John Wiley & Sons, 2019.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Elective – II (AI and ML Stream) **MCA I Year II Semester**

25MDMCADC02 MACHINE LEARNING

L	T	P	C
3	0	2	4

Pre-requisite

Course Description:

This course provides a concise introduction to the fundamental concepts in machine learning and popular machine learning algorithms. This will cover the standard and most popular supervised learning algorithms including linear regression, logistic regression, decision trees, k-nearest neighbor, an introduction to Bayesian learning and the naïve Bayes algorithm, support vector machines and neural networks.

Course Objectives:

This course enables students to

1. Understand the basic concepts and applications of Machine Learning.
2. Explain the predictive models used in machine learning applications.
3. Analyse the classification model respect to real time problems.
4. Recognize the clustering models and Dimensional Reduction Techniques.
5. Experiment the hybrid models intend to increase the accuracy of ML based applications.

UNIT I INTRODUCTION 9 Hours

Define Machine Learning - Machine Learning vs. Traditional Programming - Types of Machine Learning – Machine Learning and other disciplines - Machine Learning Project Lifecycle - Advantages, Disadvantages & Challenges

UNIT II SUPERVISED LEARNING 9 Hours

Linear regression, Decision trees, k-Nereast Neighbours, Support Vector Machine, Logistic regression, Random Forest. Artificial Neural Network- Multi-layer networks – Evaluation Metrics and Performance.

UNIT III UNSUPERVISED LEARNING 9 Hours

Supervised vs Unsupervised Cluster Analysis, K-means clustering, Hierarchical clustering. Density based clustering – Grid based clustering, Dimension reduction: Principal Component Analysis, Linear Discriminant Analysis - Evaluation Metrics and Performance.

UNIT IV REINFORCEMENT LEARNING 9 Hours

Introduction to Reinforcement Learning- agent, environment, state, action, reward- Markov Decision Processes (MDP)-Dynamic Programming- Monte Carlo Methods- Temporal Difference Learning-TD prediction-Q-learning algorithm-SARSA algorithm

UNIT V ENSEMBLE LEARNING & EXPLAINABLE AI 9 Hours

Boosting – AdaBoost Algorithm – Bagging – Random Forest – XGBoost Algorithm – Stacking – Voting –Evaluating Ensembles of Classifiers. **F**eature Importance- LIME (Local Interpretable Model-agnostic Explanations)- SHAP (SHapley Additive exPlanations)-**.**Dependence Plots-Counterfactual Explanations

List of Experiments **30 Hours**

1. Build a Naïve Bayes classifier to classify emails into spam or non-spam using a dataset of your choice.
2. Use the Expectation-Maximization algorithm to fit a Gaussian Mixture Model to synthetic data and plot the resulting clusters.
3. Implement the k-Nearest Neighbors algorithm and test it on a classification dataset.
4. Implement logistic regression for predicting binary outcomes and evaluate model accuracy.
5. Use Random Forest to classify a dataset and report precision, recall, and F1-score.
6. Apply K-means clustering on the Iris dataset and visualize the clusters.
7. Perform hierarchical clustering on a dataset and plot the dendrogram to show cluster hierarchy.
8. Use Principal Component Analysis (PCA) to reduce the dimensionality of a dataset and visualize the results in 2D.
9. Implement the AdaBoost algorithm on a binary classification problem and evaluate its performance.
10. Train an XGBoost model on a classification dataset and report
11. Create a voting classifier that combines Logistic Regression, SVM, and Random Forest, and compare its performance with individual classifiers.t its accuracy.

Software requirements: Juptyer Notebook, Google Colab

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Understand the real-world applications that needs machine learning based solutions.

CO2: Examine the predictive machine learning Models.

CO3: Experiment the Classification algorithms in appropriate real-time applications

CO4: Implement the Clustering algorithms for real world problems.

CO5: Apply Ensemble Learning Modes for Real Time Predictions

Text Book(s)

1. Ethem Alpaydın, Introduction to Machine Learning, The MIT Press, Fourth Edition (2020)

Reference Books

1. Andreas C. Müller and Sarah Guido, Introduction to Machine Learning with Python, O'Reilly Media (2020)

2. Oliver Theobald, Machine Learning for Absolute Beginners, CreateSpace Independent Publishing Platform (2018)

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Elective – III (AI and ML Stream) **MCA II Year I Semester**

25MDMCADC03 NATURAL LANGUAGE PROCESSING

L	T	P	C
3	0	2	4

Pre-requisite: 25MDMCAEC06, 25MDMCADC02

Course Description:

This course provides comprehensive knowledge of Natural Language Processing (NLP) covering text pre-processing, feature engineering, machine learning models, deep learning architectures, and transformer-based approaches. The course integrates theory with practical laboratory sessions to build real-world NLP applications such as sentiment analysis, topic modelling, and question answering, summarization, and chatbot systems.

Course Objectives:

This course enables students

1. To understand linguistic and statistical foundations of NLP.
2. To apply pre-processing and feature extraction techniques to textual data.
3. To develop and evaluate machine learning and transformer-based NLP models.
4. To design intelligent NLP systems for real-world applications.
5. To gain practical experience in implementing end-to-end NLP solutions.

UNIT I ESSENTIALS OF NLP AND TEXT PROCESSING METHODS 9 Hours

Introduction to Natural Language Processing, characteristics and challenges of natural language, NLP pipeline, Text Preprocessing: techniques including tokenization, stop-word removal, stemming and lemmatization, text normalization, feature extraction methods such as Bag-of-Words, TF-IDF and N-grams.

UNIT II MACHINE LEARNING AND SEQUENCE MODELS FOR NLP 9 Hours

Part-of-Speech (POS) tagging, sequence modelling using Hidden Markov Models (HMM) and Conditional Random Fields (CRF), text classification using Naïve Bayes and Logistic Regression, document clustering using the K-Means algorithm, text similarity measures, word embedding techniques using Word2Vec.

UNIT III DATA STORAGE CONCEPTS IN NLP 9 Hours

Relational and NoSQL databases for NLP applications – indexing and retrieval of textual data – storage formats for NLP datasets such as CSV, JSON, XML, and text files – handling large-scale language data in distributed storage systems – data preprocessing, cleaning, and normalization for storage efficiency – privacy, security, and ethical considerations in storing linguistic data..

UNIT IV DEEP LEARNING AND TRANSFORMERS 9 Hours

Introduction to deep learning for NLP, transformer architecture overview, BERT-based text classification, transformer-based Named Entity Recognition, semantic similarity using sentence transformers, and attention mechanism concepts. Case study on transformer-based NLP applications.

UNIT V ADVANCED APPLICATIONS AND DEPLOYMENT 9 Hours

Text summarization using transformer models (BERT/T5)-multilingual NLP models, conversational AI systems, deployment of NLP models using Streamlit or Flask, and ethical issues including bias and fairness in NLP systems. Case studies on real-world NLP applications and deployment scenarios.

List of Experiments 30 Hours

1. Apply text preprocessing and normalization techniques using Python with NLTK and spaCy.
2. Apply Bag-of-Words, TF-IDF and N-gram feature extraction methods using Python with Scikit-learn.
3. Build and evaluate text classification models using Naïve Bayes and Logistic Regression with Python using Scikit-learn.
4. Perform document clustering using the K-Means algorithm with Python using Scikit-learn.
5. Develop a Word2Vec word embedding model and analyze semantic similarity using Python with Gensim.
6. Conduct sentiment analysis using machine learning techniques with Python using Scikit-learn and Pandas.
7. Perform Named Entity Recognition (NER) using Python with spaCy and Hugging Face Transformers.
8. Apply topic modeling using Latent Dirichlet Allocation (LDA) with Python using Gensim.
9. Develop a text classification model using a pre-trained BERT model with Python using Hugging Face Transformers and PyTorch/TensorFlow.
10. Develop a question answering system using a pre-trained BERT model with Python using Hugging Face Transformers.

Software Requirements:
OS: Windows / Linux / macOS
Programming Language: Python 3.8 or higher
Development Environment: Jupyter Notebook / JupyterLab
Code Editor / IDE: Visual Studio Code / PyCharm
Libraries for NLP: NLTK, spaCy

Machine Learning Library: scikit-learn
Topic Modeling & Embeddings: Gensim
Data Processing Libraries: Pandas, NumPy
Transformer Frameworks: Hugging Face Transformers

Deep Learning Framework: PyTorch

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Understand core NLP concepts and pre-processing techniques.

CO2: Apply machine learning and deep learning models for NLP tasks.

CO3: Implement transformer-based models for advanced NLP applications.

CO4: Develop and evaluate real-world NLP systems.

CO5: Design end-to-end NLP solutions for practical applications.

Text Books

1. Jurafsky, D., & Martin, J. H. (2025). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition (3rd ed., draft). Pearson
2. Eisenstein, J. (2019). Introduction to natural language processing. MIT Press. (More recent foundational alternative to Manning & Schütze)
3. Goldberg, Y. (2017). Neural network methods for natural language processing. Morgan & Claypool Publishers.

Reference Books

1. Indurkhya, N., & Damerau, F. J. (Eds.). (2023). Handbook of natural language processing (3rd ed.). CRC Press.
2. Eisenstein, J. (2019). Introduction to natural language processing. MIT Press. (More recent foundational alternative to Manning & Schütze)
3. Goldberg, Y. (2017). Neural network methods for natural language processing. Morgan & Claypool Publishers.
.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Elective – IV (AI and ML Stream) **MCA II Year I Semester**

25MDMCADC04 DEEP LEARNING FOR COMPUTER VISION

L	T	P	C
3	0	2	4

Pre-requisite: 25MDMCADC02, 25MDMCAEC06

Course Description:

This course introduces the fundamentals of deep learning and computer vision, covering neural networks, CNNs, RNNs, and LSTM models for visual and sequential data analysis. It includes classical digital image processing techniques and modern deep learning–based vision approaches for image classification, object detection, and segmentation. Students gain hands-on exposure to popular tools such as OpenCV, TensorFlow, and PyTorch, along with an understanding of real-world applications, challenges, and ethical considerations in computer vision.

Course Objectives:

This course enables students to

1. Understand the fundamentals of deep learning and neural network models.
2. Apply Convolutional Neural Network (CNN) architectures to solve real-world problems.
3. Analyse sequence modelling using RNN, LSTM, and related networks.
4. Implement digital image processing techniques for image analysis.
5. Develop computer vision applications using deep learning approaches.

UNIT I DEEP LEARNING CONCEPTS 9 Hours

Deep Learning - Deep Learning vs Machine Learning -Artificial Neuron and Perceptron Model - Multilayer Perceptrons (MLPs) -Feedforward Neural Networks - Backpropagation Algorithm- Activation Functions: Sigmoid, Tanh, ReLU, Softmax -Overfitting and Regularization

UNIT II CONVOLUTIONAL NEURAL NETWORKS 9 Hours

CNN Architecture: Convolution Layer, Pooling Layer, Fully Connected Layer, Kernel, Stride, and Padding - Pre trained CNN Models: AlexNet, VGGNet, GoogLeNet, ResNet, U-Net.

UNIT III RNN & LSTM 9 Hours

Recurrent Neural Networks (RNNs) - Sequence Modelling and Time Dependencies -Vanishing and Exploding Gradient Problems - Long Short-Term Memory (LSTM) Networks - LSTM Architecture -Bidirectional LSTM -Comparison: RNN vs LSTM.

UNIT IV DIGITAL IMAGE PROCESSING 9 Hours

Digital Image Fundamentals - Image Representation and Color Models -Sampling and Quantization - Relationship Between Pixels - Image Enhancement - Point, Line, and Edge Detection -Edge Detection Operators - Thresholding Techniques.

UNIT V COMPUTER VISION 9 Hours

Computer Vision - Computer Vision vs Human Vision -Challenges in Computer Vision - Image Classification, Object Detection, and Segmentation - Computer Vision Libraries: OpenCV, Tensor Flow, PyTorch - Emerging Trends in Computer Vision.

List of Experiments **30 Hours**

1. Implement a neural network model to recognize handwritten digits using the MNIST dataset.
2. Implement a neural network with different activation functions and compare their classification performance.
3. Develop a Convolutional Neural Network (CNN) to classify images from the CIFAR-10 dataset.
4. Implement a U-Net model to segment medical images using the dataset.
5. Design an image caption generation system using a CNN–LSTM architecture trained on the Flickr8k dataset.
6. Apply an LSTM-based model for weather time-series forecasting using the Daily Climate Time Series dataset.
7. Implement image enhancement and noise removal on medical images from the NIH Chest X-ray dataset.
8. Implement edge-based object boundary detection using the BSDS500 dataset.
9. Develop a face recognition system using CNN and OpenCV with the LFW dataset.
10. Implement and evaluate an object detection model for autonomous driving applications using the KITTI dataset.

Software Requirements:
OS: Windows / Linux / macOS
Programming Language: Python 3.8 or higher
Development Environment: Jupyter Notebook / JupyterLab /Apache Zeppelin / Deepnote
Code Editor / IDE: Visual Studio Code / PyCharm
Libraries for NLP: NLTK, spaCy
Machine Learning Library: scikit-learn
Topic Modeling & Embeddings: Gensim
Data Processing Libraries: Pandas, NumPy
Transformer Frameworks: Hugging Face Transformers
Deep Learning Framework: PyTorch

Course Outcomes:

Upon successful completion of the course, students will be able to
CO1: Understand deep learning concepts to build and train basic neural network models.
CO2: Implement convolutional neural networks for image-based applications.
CO3: Apply RNN and LSTM models for sequence and temporal data analysis.
CO4: Design and use digital image processing techniques.
CO5: Develop computer vision applications using traditional and deep learning methods.

Text Books

1. Simon J. D. Prince, Understanding Deep Learning, MIT Press, 2023.

2. Zhang Aston et al., Dive into Deep Learning, Cambridge University Press, 2023..

3. Mohamed Elgendy, Deep Learning for Vision Systems, Manning, 2020.

Reference Books

1. Joe Minichino, *Learning OpenCV 5 Computer Vision with Python*, O'Reilly Media, **2024**.

2. Simon J. D. Prince, Understanding Deep Learning, MIT Press, 2023.

3. Nikhil Singh, Fundamentals of Deep Learning and Computer Vision, BPB Publications, 2022.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Electives
(Cloud Computing Stream)

Discipline Elective – I (Cloud Computing Stream) **MCA I Year II Semester**

25MDMCADC05 CLOUD COMPUTING

L	T	P	C
3	0	2	4

Pre-requisite Nil

Course Description:

This course describes the introduction of Cloud and the various service delivery models of a Cloud computing architecture, followed by the Cloud management, Performance issues, Hard ware concepts, Storage, Security and Privacy issues.

Course Objectives:

This course enables students to:

1. Understand the fundamentals, architecture, and characteristics of cloud computing.
2. Explore virtualization techniques and various cloud service models.
3. Examine cloud deployment models and applications for personal and community use.
4. Analyze cloud programming models and supporting platforms.
5. Identify key security issues, risks, and mitigation strategies in cloud computing

UNIT I INTRODUCTION TO CLOUD COMPUTING 9 Hours

Computing Paradigms - Distributed Computing, Grid Computing, Cluster Computing, Utility Computing: Cloud Computing – History of Cloud Computing – Cloud Characteristics - Cloud Architecture – Cloud Storage – Advantages & Disadvantages of Cloud Computing.

UNIT II VIRTUALIZATION AND CLOUD SERVICE MODELS 9 Hours

Introduction to Virtualization - Types - Cloud Services: Software as a Service – Platform as a Service – Infrastructure as a Service (IaaS) – On-Demand Computing – Discovering Cloud Services and Tools – Amazon Ec2 – Google App Engine – IBM Clouds.

UNIT III CLOUD DEPLOYMENT MODELS AND CLOUD COMPUTING FOR EVERYONE 9 Hours

Introduction to Cloud Deployment Models - Types - Centralizing Email Communications - Collaborating on Schedules – Collaborating on To-Do Lists – Cloud Computing for the Community – Collaborating on Group Projects and Events.

UNIT IV PROGRAMMING MODEL 9 Hours

Parallel and Distributed Programming Paradigms – Map Reduce, Twister and Iterative Map Reduce – Hadoop Library from Apache – Mapping Applications - Programming Support, Amazon AWS - Cloud Software Environments -Eucalyptus, Open Nebula, Open Stack, Aneka, Cloud Sim.

UNIT V SECURITY IN THE CLOUD 9 Hours

Security Overview - Cloud Security Challenges and Risks - Software-as-a-Service Security- Security Governance - Risk Management - Security Monitoring - Security Architecture Design - Data Security - Application Security - Virtual Machine Security- Serverless Security- Cloud Security Automation and DevSecOps

List of Experiments **30 Hours**

1. Install VirtualBox/VMware Workstation with different flavors of Linux or windows OS on top of windows OS.
2. Install a C compiler in the virtual machine created using virtual box and execute Simple Programs
3. Install Google App Engine. Create hello world app and other simple web applications using python/java.
4. Install the Hadoop framework and create an application using Map Reduce Programming Model
5. Experiment cloud scheduling algorithms using any Cloud tools.
6. Experiment cloud load balancing algorithms using Cloud Sim or any tools.
7. Launch EC2 AWS – Instance Creation, Migration.
8. Experiment VPC in EC2 Instances.
9. Create the Load balance in EC2.
10. Design and implementation the Web application and launch in AWS Server.

Software Requirements: VMware Workstation, GCC, Google Cloud SDK, Java Development Kit, CloudSim (for scheduling and load balancing experiments), AWS Management Console (web-based)

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Describe cloud computing concepts, architecture, storage, and benefits.
CO2: Explain virtualization and compare cloud service models like SaaS, PaaS, and IaaS.
CO3: Classify different deployment models and illustrate cloud-based collaboration tools.
CO4: Apply programming models like MapReduce and utilize cloud platforms such as Hadoop and AWS.
CO5: Evaluate cloud security challenges and propose suitable solutions for secure cloud environments

Text Books

1. Michael Miller, "Cloud Computing: Web-Based Applications That Change the Way You Work and Collaborate Online", Que Publishing, August 2022.

2. Cloud Native Architectures, Tom Laszewski, Kamal Arora Erik Farr, Pivum Zonooz, Packt publishing, August 2018.

Reference Books

1. Kai Hwang, Geoffrey C Fox, Jack G Dongarra, "Distributed and Cloud Computing, From Parallel Processing to the Internet of Things", Morgan Kaufmann Publishers, 2022.

2. John W.Rittinghouse and James F.Ransome, "Cloud Computing: Implementation, Management, and Security", CRC Press, 2019.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Elective – II (Cloud Computing Stream) **MCA I Year II Semester**

25MDMCADC06 EDGE AND FOG COMPUTING

L	T	P	C
3	0	2	4

Pre-requisite Cloud Computing

Course Description:

This course extends cloud computing capabilities to the edge of the network. Students will explore the basic knowledge, architecture, principles, and applications of fog computing, and understand how it supports with real-time applications in domains such as IoT, and healthcare.

Course Objectives:

This course enables students to:

1. Understand the fundamentals, components, and architecture of edge computing systems.
2. Explore edge computing platforms, frameworks, and orchestration tools.
3. Analyze real-world applications and emerging trends in edge computing.
4. Explain the concepts, characteristics, and applications of fog computing.
5. Analyse the architecture, components, and platforms of fog computing systems.

UNIT I INTRODUCTION TO EDGE COMPUTING 9 Hours

Definition and significance of edge computing, Evolution: From centralized to decentralized systems- - Components: Edge devices, Gateways, MEC, System architecture and edge nodes, Data lifecycle at the edge, Hardware platforms and micro data centers.

UNIT II EDGE PLATFORMS AND FRAMEWORKS 9 Hours

Platforms: AWS Greengrass, Azure IoT Edge, Google Coral, Edge containers and lightweight virtualization, Edge orchestration tools: K3s, Open Horizon, Edge data analytics and AI.

UNIT III USE CASES AND FUTURE DIRECTIONS OF EDGE COMPUTING 9 Hours

Case studies: Autonomous vehicles, Edge surveillance, Smart factories, Smart cities, Healthcare, Edge-cloud collaboration, Standardization and industry trends, Future research directions in edge computing.

UNIT IV INTRODUCTION TO FOG COMPUTING 9 Hours

Definition – Evolution – Characteristics – Applications - need for Fog Computing - similarities and differences of Fog vs Edge vs Cloud Computing - Issues and challenges - Key characteristics: proximity - low latency - geographic distribution

UNIT V FOG COMPUTING ARCHITECTURE AND COMPONENTS 9 Hours

Fog Computing Architecture: Layers and Components - Fog Nodes: Hardware and Software Components. Fog Computing Platforms: Cisco Fog Computing: Overview – Architecture - Components. OpenFog Consortium: Overview – Architecture – Components. Google Cloud IoT Edge: Overview – Architecture – Components.

List of Experiments **30 Hours**

1. Set up the Arduino IDE for ESP8266-12 module and program it to blink a LED light.
2. Write a program to simulate data collection and pre-processing on edge devices before sending to the cloud.
3. Deploy micro services and writing your own microservices
4. Setup the Communication Parameters
5. Implement any two Communications protocols
6. Deploy a Lambda function on AWS Green grass or a module on Azure IoT Edge that processes data locally.
7. Create an IoT hub.
8. Register an IoT Edge device to your IoT hub.
9. Install and start the IoT Edge for Linux on Windows runtime on your device
10. Simulate sensor data generation for a smart factory and implement local anomaly detection on edge nodes.
11. Publishing Data using HTTP.
12. Deploy a module Manage your Azure IoT Edge device from the cloud to deploy a module that sends telemetry data to IoT Hub

Software Requirements: Arduino IDE, Python 3, DockerAzure CLI (or AWS CLI depending on cloud platform) Postman

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Understand the principles, architecture and scope of edge computing.
CO2: Design edge-based systems for latency-sensitive applications
CO3: Analyze real-world edge applications and trends
CO4: Acquire Fog Computing fundamentals, compare it with Cloud/Edge Computing
CO5: Analyse Fog architectures, protocols, and resource management techniques

Text Books

1. Perry Lea, *Edge Computing: A Primer*, O'Reilly Media, 1st Edition, 2022.

2. Assad Abbas, Samee U. Khan, "Fog Computing: Theory and Practice " Wiley Publication, 2020.

Reference Books

1. Satyanarayanan, M., *The Emergence of Edge Computing*, IEEE, 2017

2. Amir Vahid Dastjerdi and Rajkumar Buyya, "Fog Computing: Helping the Internet of Things Realize its Potential", University of Melbourne, 2016.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Elective – III (Cloud Computing Stream) **MCA II Year I Semester**

25MDMCADC07 CLOUD WEB SERVICES

L	T	P	C
3	0	2	4

Pre-requisite: 25MDMCADC05

Course Description:

This course introduces the basics of cloud computing and the services of Amazon web Services. It covers cloud service models, deployment models, and basic cloud security. The course explains compute services such as EC2 and application deployment. It also includes AWS networking, monitoring, and messaging services. In addition, storage and database services such as S3, EBS, RDS and DynamoDB are covered.

Course Objectives:

This course enables students to:

1. Explain the fundamentals of cloud computing, including cloud service models, deployment models and fundamentals of AWS.
2. Demonstrate the use of computer services of Amazon Web Services.
3. Apply web application deployment and monitoring using services such as Elastic Beanstalk, Cloud Formation, and Cloud Watch.
4. Implement AWS networking components including VPC, subnets, security groups, and network access control lists.
5. Apply AWS storage and database services such as S3, EBS, RDS, and DynamoDB..

UNIT I CLOUD COMPUTING AND AWS FUNDAMENTALS 9 Hours

Introduction to Cloud Computing-Cloud Service Models – Cloud Deployment Models-Cloud Computing Security Basics-Overview of AWS services: Compute Services-Storage Services-Database services-Monitoring and management services.

UNIT II AMAZON EC2 AND COMPUTE SERVICES 9 Hours

Introduction to EC2(Elastic Compute Cloud)- EC2 instance types and their uses-Amazon Machine Images (AMI)- Launching and managing EC2 instances-Auto Scaling concepts-Elastic IP addresses-Elastic Load Balancing.

UNIT III WEB APPLICATION DEPLOYMENT & MONITORIING 9 Hours

Introduction to Elastic Beanstalk-Deploying scalable web applications on AWS-Creating and managing application environments-Introduction to Cloud Formation-Infrastructure as Code Concept-Introduction to Cloud Watch-Monitoring AWS resources-AWS messaging services: SNS, SQS, SES.

UNIT IV MICROSOFT AZURE FUNDAMENTALS 9 Hours

Azure networking concepts including Virtual Networks, Load Balancer, VPN Gateway, DNS, and Content Delivery Network – Azure databases and data services – identity and access management using Microsoft Entra ID (Azure Active Directory) – security, monitoring, backup, disaster recovery, and governance in Azure – deployment, management, and cost optimization of cloud resources using Azure Portal, CLI, and automation tools.

UNIT V AWS STORAGE AND DATABASE SERVICES 9 Hours

AWS Storage services-Amazon S3 storage concepts-Buckets and objects-Hosting a static website using S3-Elastic Block Store (EBS)- Glacier archival storage-AWS database services: Amazon RDS-Amazon DynamoDB.

List of Experiments 30 Hours

1. To create an AWS account and explore the AWS Management Console and its major services (EC2, S3, RDS).
2. Create and access a virtual machine using EC2 in Amazon Web Services.
3. To create a custom Amazon Machine Image (AMI) from a running EC2 instance in Amazon Web Services.
4. To allocate and associate an Elastic IP address to an EC2 instance in Amazon Web Services.
5. To deploy a simple web application using Elastic Beanstalk in Amazon Web Services.
6. Create and use Simple Notification Service (SNS) for sending notifications in Amazon Web Services.
7. To create and use Simple Notification Service (SNS) for sending notifications in Amazon Web Services.
8. To configure a Security Group and Network Access Control List (ACL) to control network access in Amazon Web Services.
9. To create and manage an S3 bucket for storing and accessing files in Amazon Web Services.
10. To launch and connect to an RDS database instance (MySQL) using Amazon Web Services

Course Outcomes:

Upon successful completion of the course, students will be able to
CO1: Explain the fundamental concepts of Cloud Computing
CO2: Demonstrate the use of core services of Amazon EC2 Services
CO3: Deploy scalable applications using AWS compute, storage, and database
CO4: Implement cloud networking components such as VPC, security groups
CO5: Apply AWS storage and database services.

Text Books

1. Mastering Cloud Computing: Foundations and Applications Programming, 2nd Edition — Authors: Rajkumar Buyya et al. — 2nd Edition, 2024
2. Amazon Web Services in Action, Andreas Wittig & Michael Wittig, 3rd Edition 2023.
3. Cloud Computing: Principles and Paradigms, Rajkumar Buyya, James Broberg, Andrzej M. Goscinski, John Wiley & Sons-2011.

Reference Books

1. Cloud Computing with AWS: Everything You Need to Know to be an AWS Cloud Practitioner, Pravin Mishra, Apress, 2023
2. Amazon Web Services in Action, Andreas Wittig & Michael Wittig, Manning, 3rd Edition, 2023.
3. Cloud Computing: Principles, Systems and Applications, Nick Antonopoulos & Lee Gillam, Springer, 2nd Edition, 2017

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Elective – IV (Cloud Computing Stream) **MCA II Year I Semester**

25MDMCADC08 CLOUD SECURITY

L	T	P	C
3	0	2	4

Pre-requisite: 25MDMCADC05, 25MDMCADC06

Course Description:

This course provides an in-depth exploration of security principles, challenges, and best practices within the cloud computing environment. It covers the entire spectrum of cloud security, from fundamental concepts and architectural models to risk management, compliance, and advanced security technologies.

Course Objectives:

This course enables students to:

1. Understand the fundamental security concepts in cloud computing.
2. Explore virtualization technologies and their associated security implications.
3. Learn about cloud security architecture and compliance frameworks.
4. Identify and mitigate various cloud-specific risks, threats, and vulnerabilities.
5. Gain familiarity with security tools and practices.

UNIT I FUNDAMENTALS OF CLOUD SECURITY 9 Hours

Introduction to Cloud Security: Cloud Information Security Objectives - Security Challenges: Risks and approaches for migrating to the cloud, ethical issues in cloud computing. Governance and Compliance: Security governance principles, information risk management.

UNIT II VIRTUALIZATION SECURITY 9 Hours

Guest hopping, attacks on the VM-VM migration attack, hyper jacking. System attacks: Management Server attack, Administrative VM attack, Guest VM attacks, and Hypervisor attacks. Data Security: Cloud data life cycle, data-at-rest and data-in-transit encryption.

UNIT III SECURE CLOUD SOFTWARE & IDENTITY MANAGEMENT 9 Hours

Secure cloud software requirements, secure development practices, and cloud security policy implementation. Identity and Access Management (IAM) - Authentication, access control, identity management.

UNIT IV CLOUD RISK, COMPLIANCE, AND LEGAL ISSUES 9 Hours

CIA- triad, privacy and compliance risks, threats to infrastructure, data, and access control. Cloud Service Provider Risks: Understanding SLAs, trust management, and vendor lock-in. Legal and Compliance Issues: Legal frameworks, data sovereignty.

UNIT V ADVANCED CLOUD SECURITY AND INCIDENT RESPONSE 9 Hours

Trusted cloud computing, secure execution environments. Advanced Security Tools: Intrusion detection systems (IDS), Security information and Event management (SIEM), Roles. Incident Response and Forensics: Disaster management, incident response planning, and forensic investigations in the cloud.

List of Experiments **30 Hours**

1. Demonstrate Environmental setup of IAM using free-tier accounts on AWS / Azure or Google Cloud. Configure users, groups, and roles with IAM policies.
2. Deploy a virtual machine (EC2, Compute Engine), configure security groups and network ACLs to restrict access, and implement OS-level security measures.
3. Create and configure cloud storage buckets (S3, Blob Storage). Implement policies to control public access, enable encryption at rest.
4. Demonstrate Virtual Private Cloud (VPC) with public and private subnets. Configure Web Application Firewalls (WAF) and analyze traffic logs.
5. Use the cloud provider's Key Management Service (KMS) to create and manage encryption keys. Encrypt and decrypt data programmatically using SDKs .
6. Perform a vulnerability scan on a deliberately vulnerable VM using tools like OpenVAS or cloud-native security scanners. Analyze and document findings.
7. Simulate a security incident (e.g., unauthorized API call) and use cloud logging and monitoring tools to detect, investigate, and create a response playbook.
8. Demonstrate cloud compliance tools (like AWS Config or Azure Policy) to assess the compliance of cloud resources against a standard framework (e.g., CIS benchmarks).

Software Requirements:

Cloud Account (AWS / Azure / GCP), Web Browser, Python, AWS CLI, PuTTY

Course Outcomes:

Upon successful completion of the course, students will be able to
CO1: Articulate the core security challenges in cloud computing environments.
CO2: Apply security principles in cloud infrastructure components.
CO3: Design and implement identity, access, and data security controls for cloud-based systems.
CO4: Evaluate and select appropriate security tools for a given cloud use case.
CO5: Demonstrate the ability to configure and use security in cloud computi**ng.**

Text Books

1. Jamuna S. Murthy, Siddesh G. M., Srinivasa K. G. Cloud Security Concepts Applications and Practices, 1st Edition, 2025.
2. Eyal Estrin, Cloud Security Handbook, 1st Edition, 2022.

Reference Books

1. S. Balamurugan and S. Charanyaa, "Concepts, Principles and Practices of Security in Cloud", Scholar's Press, 2015.Amazon Web Services in Action, Andreas Wittig & Michael Wittig, Manning, 3rd Edition, 2023.
2. Mather, T., Kumaraswamy, S., & Latif, S. Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance. O'Reilly Media, 2009

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Electives
(Cyber Security Stream)

Discipline Elective – I (Cyber Security Stream) **MCA I Year II Semester**

25MDMCADC09 CYBER SECURITY

L	T	P	C
3	0	2	4

Pre-requisite

Course Description:

This course introduces the principles of cyber security, including cryptographic techniques, network and system security, threats and vulnerabilities, secure software development, and emerging trends. Students will gain both theoretical knowledge and hands-on skills through lab experiments.

Course Objectives:

This course enables students able to

1. Understand the fundamentals of cyber security, threats, and governance policies.
2. Explore cryptographic techniques for securing data and communications.
3. Examine network security mechanisms and secure communication protocols.
4. Identify secure practices in system, application, and web environments.
5. Recognize emerging security trends and related cyber laws and ethics.

UNIT I INTRODUCTION TO CYBER SECURITY 9 Hours

Overview of Cyber Security – Need, Goals, and Challenges, Cyber Threats: Malware, Phishing, Social Engineering, Denial of Service, Advanced Persistent Threats (APTs), Security Services – Confidentiality, Integrity, Availability, Security Policies and Governance.

UNIT II CRYPTOGRAPHY AND DATA SECURITY 9 Hours

Symmetric Encryption: Block and Stream Ciphers (AES, DES), Asymmetric Encryption: RSA, ECC, Hashing and Message Digest, Digital Signatures and Certificates, Key Management and Public Key Infrastructure (PKI).

UNIT III NETWORK SECURITY 9 Hours

Firewalls, IDS, IPS, Virtual Private Networks (VPNs), Secure Socket Layer (SSL) / Transport Layer Security (TLS), Email Security (PGP, S/MIME), IP Security (IPSec).

UNIT IV SYSTEM AND APPLICATION SECURITY 9 Hours

Authentication and Access Control, Secure Software Development Practices, Operating System Security – Windows & Linux, Web Security – SQL Injection, XSS, CSRF, Security Auditing and Vulnerability- Identity and Access Management (IAM)- Patch and Vulnerability Management

UNIT V EMERGING AREAS AND CYBER LAWS 9 Hours

Cloud Security – Threats and Solutions, IoT Security Issues, Mobile Security, Cyber Forensics Basics, Cyber Laws and Ethics (Indian IT Act 2000, GDPR overview).

List of Experiments **30 Hours**

1. Study of basic Linux security commands (user management, permissions, firewalls).
2. Implementation of Caesar Cipher and Monoalphabetic Cipher.
3. Implementation of DES and AES algorithms.
4. Implementation of RSA algorithm.
5. Implementation of SHA-256 hashing algorithm.
6. Simulation of Digital Signatures.
7. Setting up and testing a Firewall (iptables / Windows Firewall).
8. Configuring and testing VPN using open-source tools.
9. Installation and configuration of IDS/IPS (Snort/Suricata).
10. Web Application Security Testing (SQL Injection, XSS) using DVWA.
11. Performing vulnerability scanning using **Nmap/Nessus**.
12. Case Study / Mini Project: Cyber Forensics investigation or Cloud Security simulation.

Software requirements: Linux OS (Ubuntu or Kali),Python3, OpenSSLip tables (or Windows Firewall), Snort (IDS), DVWA, Nmap

Course Outcomes:

Upon successful completion of the course, students will be able to
CO1: Explain security concepts, threats, and defence mechanisms.
CO2: Apply cryptographic algorithms for securing data.
CO3: Analyze and secure networks using appropriate security protocols.
CO4: Demonstrate skills in intrusion detection, firewalls, and secure communication.
CO5: Evaluate security issues in emerging technologies and apply compliance measures.

Text Books

1. William Stallings, *Cryptography and Network Security: Principles and Practice*, **7th Edition**, Pearson, **2017**.
2. Mark Stamp, *Information Security: Principles and Practice*, **2nd Edition**, Wiley, **2011**.

Reference Books

1. Behrouz A. Forouzan and Debdeep Mukhopadhyay, *Cryptography and Network Security*, **3rd Edition**, McGraw Hill, **2015**.
2. Nina Godbole and Sunit Belapure, *Cyber Security: Understanding Cyber Crimes, Computer Forensics and Legal Perspectives*, **1st Edition**, Wiley, **2011**.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Elective – II (Cyber Security Stream) **MCA I Year II Semester**

25MDMCADC10 APPLIED CRYPTOGRAPHY

L	T	P	C
3	0	2	4

Pre-requisite Computational Mathematics

Course Description:

This course is the study of techniques used to protect digital information. This course explains how data can be kept secure using encryption, hashing, digital signatures, and key management. Students will learn both the theory of cryptographic algorithms and their practical use in securing networks and systems. The lab work includes hands-on practice with coding and tools to implement and test cryptographic methods.

Course Objectives:

This course enables students able to

1. Understand the historical background and mathematical foundations of cryptography.
2. Explore symmetric key cryptographic algorithms and their modes of operation.
3. Study the principles and algorithms of asymmetric key cryptography.
4. Analyze cryptographic hash functions, MACs, and digital signature schemes.
5. Examine real-world applications of cryptography and recent trends.

UNIT I FOUNDATIONS OF CRYPTOGRAPHY 9 Hours

History & need for cryptography, Mathematical foundations: Modular arithmetic, Euler's theorem, Fermat's theorem, GCD, prime numbers, Classical ciphers: Caesar, Monoalphabetic, Playfair, Vigenère, Hill cipher, Cryptanalysis basics

UNIT II SYMMETRIC KEY CRYPTOGRAPHY 9 Hours

Block ciphers: Feistel structure, DES, AES (structure, rounds, applications), Stream ciphers: RC4, LFSR-based ciphers, Modes of operation: ECB, CBC, CFB, OFB, CTR, Applications and limitations.

UNIT III ASYMMETRIC KEY CRYPTOGRAPHY 9 Hours

Principles of public key cryptography, RSA algorithm: key generation, encryption, decryption, ElGamal cryptosystem, Diffie–Hellman key exchange, Comparison of symmetric and asymmetric techniques.

UNIT IV CRYPTOGRAPHIC HASH FUNCTIONS & DIGITAL SIGNATURES 9 Hours

Message authentication codes (MAC), Cryptographic hash functions: MD5, SHA family, Digital signatures: DSA, RSA-based signatures, Authentication applications: Kerberos, X.509 certificates, PKI.

UNIT V APPLICATIONS OF CRYPTOGRAPHY 9 Hours

Secure email (PGP, S/MIME), Secure web transactions (SSL/TLS basics), Disk/file encryption, Blockchain fundamentals & cryptocurrencies (Bitcoin, Ethereum basics), Recent trends in cryptography (homomorphic encryption, post-quantum cryptography overview).

List of Experiments **30 Hours**

1. Write code to encrypt and decrypt messages using the Vigenère cipher. Experiment with different keys and explain how the key length affects security.
2. Implement the Euclidean algorithm to find the GCD of two integers. Use this function in a simple modular arithmetic computation.
3. Design a simple Feistel cipher with a few rounds. Encrypt and decrypt a plaintext, demonstrating how the structure works.
4. Using a cryptographic library, encrypt a plaintext using AES-128, AES-192, and AES-256. Decrypt the ciphertexts and verify the outputs.
5. Generate RSA public and private keys. Encrypt a message with the public key and decrypt it using the private key.
6. Simulate two parties performing Diffie-Hellman key exchange and derive the shared secret key.
7. Implement ElGamal encryption. Encrypt a message and decrypt it using the appropriate keys.
8. Compute the hash of various inputs using MD5 and SHA-256. Observe how small changes in input affect the hash.
9. Use PGP or GPG tools to encrypt and digitally sign an email or a file. Verify the signature.
10. Encrypt a file or folder using VeraCrypt or similar software. Decrypt the data and verify integrity.
11. Implement a simple homomorphic encryption example that supports addition or multiplication on encrypted values.

Software requirements: Python 3 (with libraries like pycryptodome, cryptography),GPG (GNU Privacy Guard),VeraCrypt, OpenSSL (optional, for cryptographic operations)

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Explain the fundamental principles and mathematical basis of cryptography.
CO2: Apply symmetric and asymmetric cryptographic techniques for secure communication.
CO3: Analyze hashing and digital signature mechanisms in authentication.
CO4: Evaluate cryptographic protocols and their role in network/system security.
CO5: Implement and test cryptographic algorithms in a simulated environment.

Text Books

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

2. Behrouz A. Forouzan, *Cryptography and Network Security*, 3rd Edition, McGraw Hill, 2015.

Reference Books

1. Bruce Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, 2nd Edition, Wiley, 2015.

2. Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 2018.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Elective – III (Cyber Security Stream) **MCA II Year I Semester**

25MDMCADC11 DIGITAL FORENSICS

L	T	P	C
3	0	2	4

Pre-requisite Nil

Course Description:

This course provides an integrated study of Digital Forensics covering identification, acquisition, preservation, analysis, and presentation of digital evidence. The course combines theoretical foundations with practical laboratory exposure in disk, memory, network, mobile, and cloud forensics along with legal and ethical considerations.

Course Objectives:

This course enables students able

1. To understand the digital forensic investigation process and methodologies.
2. To acquire and preserve digital evidence using forensic tools.
3. To analyse file systems, memory dumps, and network artifacts.
4. To apply mobile and cloud forensic investigations.
5. To understand legal and ethical aspects of digital evidence handling

UNIT I INTRODUCTION TO DIGITAL FORENSICS 9 Hours

Definition and scope of digital forensics, Types of cybercrimes and digital evidence, Phases of digital forensic investigation process, Incident response and forensic readiness, Chain of custody and digital evidence handling, Overview of forensic tools (Autopsy, FTK Imager, EnCase).

UNIT II FILE SYSTEM AND DISK FORENSICS 9 Hours

Storage media fundamentals, Live and static data acquisition techniques, Disk imaging and cloning, File system structures (FAT, NTFS, ext3/ext4), Hashing and integrity verification, Deleted file recovery and metadata analysis.

UNIT III MEMORY AND NETWORK FORENSICS 9 Hours

RAM acquisition techniques, Volatile data collection, Memory dump analysis, Network traffic capture and packet analysis, Log file analysis, Intrusion detection using network artifacts.

UNIT IV MOBILE AND CLOUD FORENSICS 9 Hours

Android and iOS forensic concepts, Mobile data acquisition techniques, SIM and SD card forensics, Cloud forensic investigation challenges, Tools for mobile and cloud forensic analysis

UNIT V LEGAL, ETHICAL AND REPORTING ASPECTS 9 Hours

Legal admissibility of digital evidence, Indian IT Act basics and GDPR overview, Privacy and jurisdiction issues, Ethics in digital investigation, Documentation and forensic report writing, Expert witness and courtroom procedures.

List of Experiments **30 Hours**

1. Creation of a forensic disk image and verification of image integrity using hash values.
2. Analysis of FAT and NTFS file systems using forensic analysis tools.
3. Recovery of deleted files and examination of slack space.
4. Extraction and analysis of file metadata and creation of a timeline of digital artifacts.
5. Acquisition and analysis of system memory using the Volatility framework.
6. Capture and analysis of network packets using Wireshark.
7. Analysis of system log files to identify suspicious or intrusion activities.
8. Extraction and analysis of data from an Android device using mobile forensic tools.
9. Preparation of a digital forensic investigation report based on analyzed evidence.

Course Outcomes:

Upon successful completion of the course, students will be able to
CO1: Explain digital forensic processes, tools, and evidence handling procedures.
CO2: Apply symmetric and asymmetric cryptographic techniques for secure communication.
CO3: Analyze memory dumps and network traffic to detect cyber intrusions.
CO4: Investigate cloud and mobile forensics and security incidents.
CO5: Prepare legally admissible forensic reports considering ethical and legal standards.

Text Books

1. Nelson, B., Phillips, A., and Steuart, C., Guide to Computer Forensics and Investigations, Cengage Learning, 7th Edition, 2024.

2. Oettinger, Learn Computer Forensics, Packt Publishing, 2nd Edition, 2022.

3. Easttom, Digital Forensics, Investigation, and Response, Jones & Bartlett Publishing, 4th edition, 2022

Reference Books

1. Oettinger, W. — Learn Computer Forensics, 2nd Edition, Packt Publishing, 2022

2. Tamma, Skulkin, Mahalik & Bommisetty — Practical Mobile Forensics, 4th Edition, Packt Publishing, 2020

3. Jaswal, N. — Hands-On Network Forensics, Packt Publishing, 2019.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Elective – IV (Cyber Security Stream) **MCA II Year I Semester**

25MDMCADC12 ETHICAL HACKING

L	T	P	C
3	0	2	4

Pre-requisite 25MDMCADC09, 25MDMCADC10

Course Description:

This course introduces the fundamentals ethical hacking, focusing on protecting computer systems, networks, and web applications from security threats. It covers topics such as information security threats, attack vectors, vulnerability assessment, penetration testing, and security controls. Students learn reconnaissance, scanning, and enumeration techniques using tools like Nmap and Wireshark, along with password cracking and network attacks.

Course Objectives:

This course enables students able

1. To understand the fundamental concepts of ethical hacking, and information security threats.
2. To learn reconnaissance, scanning, and enumeration techniques used to identify system vulnerabilities.
3. To study penetration testing tools and techniques.
4. To analyze common web and network attacks.
5. To develop the ability to apply cybersecurity frameworks.

UNIT I OVERVIEW OF INFORMATION SECURITY 9 Hours

Threats and attack vectors, vulnerability assessment and penetration testing concepts, information security controls, security laws and standards. OWASP top 10 vulnerabilities Foot printing and Reconnaissance: Introduction to network reconnaissance tools.

UNIT II SCANNING AND ENUMERATION 9 Hours

Network penetration testing, Password cracking techniques and countermeasures, NetBIOS tools Gaining and Maintaining Access: Network level attacks and countermeasures, Metasploit framework, Burp Suite.

UNIT III EXPLOITATION AND COVERING TRACKS 9 Hours

Privilege escalation, social Engineering, identity theft, countermeasures, Covering tracks using attrib command and creating Alternate Data Stream (ADS) in Windows, Erasing evidence from Windows logs, Strategies for maintaining access.

UNIT IV ADVANCED STAGES OF ATTACKS 9 Hours

Denial of service, Session hijacking, hacking web servers, hacking web applications, SQL injection: NIST cybersecurity framework, Cyber Kill chain.

UNIT V CYBER DEFENSE AND REPORTING 9 Hours

Preparing vulnerability assessment reports, presenting post testing findings, preparing recommendations- AI & Machine Learning in Cyber Security - Behavioral Analytics and Anomaly Detection -Zero Trust Architecture - Future Trends in Cyber Defense.

List of Experiments **30 Hours**

1. Implement honey pot and monitor the honey pot on network.
2. Write a script to demonstrate SQL injection attacks.
3. Implement the social networking website login page using phishing techniques
4. Write a script to demonstrate DoS attacks.
5. Install rootkits and study variety of options.
6. Implement the Techniques uses for Web Based Password Capturing.
7. Install jcrypt tool (or any other equivalent) and demonstrate Asymmetric, Symmetric Crypto algorithm,
8. Hash and Digital/PKI signatures studied in theory Network Security and Management
9. Implement passive scanning, active scanning, session hizaking, cookies extraction using Burp suit.

Course Outcomes:

Upon successful completion of the course, students will be able to
CO1: Explain the fundamental concepts of Ethical hacking.
CO2: Apply reconnaissance, scanning, and enumeration techniques
CO3: Analyze penetration testing methods and ethical hacking tools.
CO4: Examine common cyber-attacks along with their countermeasures.
CO5: Prepare vulnerability assessment reports.

Text Books

1. William Stallings and Lawrie Brown, Computer Security: Principles and Practice, 5th Edition, Pearson, 2024.
2. Michael T. Simpson, Kent Backman, James E. Corley, Hands-On Ethical Hacking and Network Defense, Cengage Learning, 2016.
3. Georgia Weidman, Penetration Testing: A Hands-On Introduction to Hacking, 1st Edition, No Starch Press, 2014.

Reference Books

1. Kevin Mitnick and William L. Simon, The Art of Deception: Controlling the Human Element of Security, Wiley, 2011 (Reprint Edition).
2. Stuart McClure, Joel Scambray, George Kurtz, Hacking Exposed: Network Security Secrets and Solutions, 7th Edition, McGraw-Hill, 2012.
3. Jon Erickson, Hacking: The Art of Exploitation, 2nd Edition, No Starch Press, 2008.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Electives
(Software Engineering Stream)

Discipline Elective – II (Software Engineering Stream) **MCA I Year II Semester**

25MDMCADC13 AGILE METHODOLOGIES

L	T	P	C
3	0	2	4

Pre-requisite: 25MDMCAEC05

Course Description:

This course introduces the principles, values, and practices of Agile software development. It covers Agile frameworks such as Scrum, Extreme Programming (XP), Kanban, and Lean, focusing on adaptive planning, iterative development, and continuous delivery. Students will learn Agile roles, ceremonies, artifacts, estimation techniques, and tools for collaborative project management. Case studies will illustrate successful Agile adoption in real-world scenarios.

Course Objectives:

This course enables students to:

1. Understand the Agile manifesto, values, and principles.
2. Compare traditional SDLC models with Agile methodologies.
3. Learn the frameworks of Scrum, XP, Kanban, and Lean.
4. Apply Agile planning, estimation, and user story techniques.
5. Analyze industry case studies to evaluate Agile adoption and best practices.

UNIT I INTRODUCTION TO AGILE 9 Hours

Agile manifesto: values and principles, Traditional vs. Agile methodologies, Benefits and challenges of Agile adoption, Agile frameworks overview.

UNIT II SCRUM FRAMEWORK 9 Hours

Scrum roles: Product Owner, Scrum Master, Development Team. Scrum events: Sprint, Planning, Daily Scrum, Review, Retrospective. Scrum artifacts: Product Backlog, Sprint Backlog, Increment. Sprint planning, velocity, and burndown charts.

UNIT III EXTREME PROGRAMMING (XP) AND KANBAN 9 Hours

XP principles: Pair programming, TDD, Continuous Integration, Refactoring. Kanban principles: Visual workflow, Work-in-Progress (WIP) limits, Pull system. Differences between Scrum and Kanban.

UNIT IV AGILE PLANNING & ESTIMATION 9 Hours

User stories and acceptance criteria. Story points, Planning Poker, Velocity. Release planning, iteration planning. Agile metrics: Lead time, Cycle time, Burnup/Burndown charts.

UNIT V AGILE ADOPTION & CASE STUDIES 9 Hours

Scaling Agile: SAFe, LeSS, Disciplined Agile.Challenges in Agile adoption (culture, scaling, distributed teams).Case studies of Agile in software industry. Emerging trends: DevOps and Agile, Agile in AI/Cloud projects.

List of Experiments **30 Hours**

1. Write a short reflection on the Agile Manifesto values. Provide two examples where Agile principles can improve project outcomes compared to Waterfall.

2. Draw flowcharts for Waterfall and Agile SDLC processes. Highlight key differences and submit with a one-paragraph explanation.

3. Assign Scrum roles (Product Owner, Scrum Master, Dev Team).Simulate a Sprint Planning meeting using a provided product backlog.

4. Write at least 5 user stories with acceptance criteria for a sample product (e.g., To-Do app). Prioritize the backlog based on business value.

5. In pairs, implement a simple algorithm (e.g., Fibonacci sequence). Switch roles every 10 minutes (Driver/Navigator).

6. Write tests first for a function (e.g., palindrome checker).Implement the function to pass all tests. Refactor code to improve readability.

7. Create a Kanban board (physical or digital like Trello).Define columns and WIP limits.Simulate task flow for 5 sample tasks.

8. Using sample data, calculate lead time and cycle time. Plot burnup and burndown charts using Excel.

9. Simulate two Scrum teams working together using SAFe or LeSS concepts.Conduct a Program Increment (PI) Planning meeting.

10. Read an Agile case study. Write a report summarizing successes, challenges, and recommendations.

Software requirements: Microsoft Excel (for charts and calculations),Trello (or any digital Kanban board tool), Diagramming Tool (e.g., Lucidchart, Draw.io, or even MS PowerPoint), Code Editor (VS Code, Sublime Text) Version Control (Git/GitHub) (optional but recommended)

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Explain Agile principles, manifesto, and frameworks.

CO2: Apply Scrum roles, events, and artifacts in project planning.

CO3: Demonstrate Agile practices such as XP, Kanban, and TDD.

CO4: Estimate and plan projects using user stories, velocity, and Agile metrics.

CO5: Analyze Agile adoption challenges and case studies.

Text Books

1. Schwaber, Ken, and Jeff Sutherland. *The Scrum Guide*. Scrum.org, 2020

2. Cohn, Mike. *Agile Estimating and Planning*. Pearson Education, 2006.

Reference Books

1. Cockburn, Alistair. *Agile Software Development: The Cooperative Game*. 2nd ed., Addison-Wesley, 2007.
2. Martin, Robert C. *Agile Software Development, Principles, Patterns, and Practices*. Pearson Education, 2003.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Elective – II (Software Engineering Stream) **MCA I Year II Semester**

25MDMCADC14 DEVOPS AND MICROSERVICES

L	T	P	C
3	0	2	4

Pre-requisite 25MDMCAEC05

Course Description:

This course provides an in-depth exploration of DevOps practices and micro services architecture, aimed at equipping students with the skills necessary to design, deploy, and manage scalable applications. Upon completion, Students will be able to effectively implement DevOps practices and design micro services-based applications, improving their organization's software development lifecycle and operational efficiency

Course Objectives:

This course enables students to:

1. Understand DevOps principles, lifecycle, and tools like Git, Selenium, Jenkins.
2. Explore the integration of Microservices within a DevOps environment
3. Analyze microservices architecture, deployment, and security in cloud settings.
4. Compare monolithic and microservices architectures with real-world case studies.
5. Analyse continuous delivery pipelines and testing strategies in DevOps.

UNIT I DEV OPS TOOLS 9 Hours

History of DevOps- DevOps and Software Development Life Cycle – Waterfall Model _Agile Model – DevOps Life Cycle – DevOps Tools: Git (version control) GitHub / GitLab (remote repositories & collaboration, Docker (containerization) Docker Compose (multi-container orchestration)Kubernetes (container orchestration)- AWS CLI / Azure CLI (cloud interaction) Postman (API testing) Chaos Monkey (chaos engineering)-User Acceptance Testing – Jenkins

UNIT II MICROSERVICES IN DEV OPS ENVIRONMENT 9 Hours

Evolution of Micro services and DevOps – Benefits of combining DevOps and Micro services- working of DevOps and Micro services in Cloud environment - DevOps Pipeline representation for a Node JS based Micro services

UNIT III INTRODUCTION TO MICROSERVICES 9 Hours

Definition of Microservices – Characteristics - Microservices and Containers – Interacting with Other Services – Monitoring and Securing the Services – Containerized Services – Deploying on Cloud-Case Study: Deploying Micro services in the Cloud for Shop Easy – An E-commerce Platform

UNIT IV MICROSERVICES ARCHITECTURE 9 Hours

Monolithic architecture- Microservices architectural style- Benefits - Drawbacks of Microservices architectural style -decomposing monolithic applications into Micro services- Monolithic Architecture at Edu Learn – An Online Learning Platform.

UNIT V **MLOps Practices & Tools** **9 Hours**

Version Control-Experiment Tracking-Data Management:-Continuous Integration/Continuous Deployment (CI/CD)-Containerization:-Orchestration & Serving:-Monitoring & Logging-Automated Testing Case Study: High Velocity and Continuous Delivery at Health Sync – A Digital Healthcare Platform.

List of Experiments

1. Git Basics: Initialize repo, commit, branch, and merge
2. Remote Repositories: Push/pull to GitHub or GitLab
3. Team Collaboration with Git Workflow
4. Docker: Create Docker file, build image, run container
5. Docker Compose: Multi-container setup
6. Kubernetes Basics: Deploy Pods, Services, Deployments
7. Kubernetes Probes: Liveness and Readiness
8. Build a Simple Microservice (REST API)
9. Inter-Microservice Communication (REST/gRPC)
10. Cloud Deployment (AWS/GCP/Azure)
11. Serveries Microservices (e.g., AWS Lambda)
12. Chaos Engineering Experiment (using Chaos Monkey)

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Describe the DevOps lifecycle and tools used for automation and testing.
CO2: Explain the synergy between Microservices and DevOps in cloud environments.
CO3: Deploy containerized microservices and monitor their performance and security.
CO4: Compare monolithic and microservices styles and design decomposition strategies.
CO5: Implement high-velocity delivery pipelines with integrated testing stages.
CO6 : Maintain the Professional ethics

Text Books

1. Tanasseri, N., & Rai, R. Microservices with Azure: Build highly maintainable and scalable enterprise-grade apps. Packt Publishing.2017.

2. Wolff, E. Microservices: Flexible Software Architecture. Pearson Education, 2016.

Reference Books

1. James A Scott, A Practical Guide to Microservices and Containers, MapR Data Technologies https://mapr.com/ebook/microservices-and-containers/assets/microservices-andcontainers.pdf

2. Gene Kim, Kevin Behr, George Spafford, The Phoenix Project, A Novel about IT, DevOps, 5th Edition, IT Revolution Press, 2018

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Elective – III (Software Engineering Stream) **MCA II Year I Semester**

25MDMCADC15 SOFTWARE TESTING

L	T	P	C
3	0	2	4

Pre-requisite: 25MDMCAEC05

Course Description:
This course builds upon the concepts of Software Engineering and focuses on advanced software testing practices and quality engineering techniques. It emphasizes systematic test design, defect management, automation practices, API testing, and performance testing using industry-standard tools. The course enables students to apply testing strategies in modern Agile-based development environments.

Course Objectives:
This course enables students to:

1. Understand structured test processes and defect management practices.
2. Apply advanced test design techniques for effective validation.
3. Develop automated test scripts using industry tools.
4. Perform API and performance testing in real-world scenarios.
5. Implement testing practices in Agile and modern development environments.

UNIT I TEST PROCESS AND TEST MANAGEMENT 9 Hours

Test Process in SDLC – Test Planning and Strategy – Test Case Design and Documentation – Defect Life Cycle – Defect Metrics – Introduction to Test Management Tools.

UNIT II ADVANCED TEST DESIGN TECHNIQUES 9 Hours

Equivalence Partitioning and Boundary Value Analysis (Review) – Decision Table Testing – State Transition Testing – Cyclomatic Complexity in Application – Code Coverage Techniques in Statement and Branch.

UNIT III TEST AUTOMATION BASICS 9 Hours

Introduction to Test Automation – Selenium WebDriver – Locators and Web Elements – Handling Forms and Alerts – TestNG/JUnit Basics – Page Object Model – Test Execution and Reporting.

UNIT IV API AND PERFORMANCE TESTING 9 Hours

REST API Concepts – API Testing using Postman – HTTP Methods – JSON Validation – Test Case Design for APIs – Introduction to Performance Testing – Load Testing using JMeter – Result Analysis.

UNIT V CONTEMPORARY TESTING PRACTICES 9 Hours

Role of Testing in Agile – Continuous Testing – Test Metrics and Reporting – Risk-Based Testing AI tools - Testim, Applitools, Mabl, Functionize, Testsigma, and Katalon – integration with CI/CD and DevOps for automated quality assurance.

List of Experiments **30 Hours**

1. Prepare a Test Plan document for a sample application.
2. Design test cases using Equivalence Partitioning and Boundary Value Analysis.
3. Calculate Cyclomatic Complexity for a given program.
4. Perform white-box testing and compute statement coverage.
5. Automate a login page using Selenium WebDriver.
6. Handle forms and alerts using Selenium.
7. Implement simple automation using TestNG/JUnit.
8. Perform API testing using Postman (GET and POST methods).
9. Validate JSON responses in Postman.
10. Mini Project: Develop and execute manual + automated test cases for a web application.

Software Requirements: Selenium WebDriver, TestNG/JUnit, Postman, Apache JMeter, Java/Python.

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Explain structured testing processes and defect management practices.
CO2: Apply advanced test design techniques to validate software systems.
CO3: Develop automated test solutions using modern testing tools.
CO4: Analyze software systems through API and performance testing.
CO5: Evaluate testing practices in Agile and modern development environments.

Text Books

1. Mark Winteringham, Software Testing with Generative AI, Manning Publications, 2024.
2. Gayathri Mohan, Full Stack Testing: A Practical Guide, O'Reilly Media, 2021.
3. Aditya P. Mathur, Foundations of Software Testing, 3rd Edition, Pearson, 2022.

Reference Books

1. James Bach & Michael Bolton, The Rapid Software Testing Approach, Addison-Wesley, 2023.

2. Marie Cruz & Lewis Prescott, *Contract Testing in Action*, Manning Publications, **2023**.

3. Vijay Kulkarni, *Practical Software Testing with Selenium and DevOps*, Packt Publishing, **2022**.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.

Discipline Elective – IV (Software Engineering Stream) **MCA II Year I Semester**

25MDMCADC16 SOFTWARE PROJECT MANAGEMENT

L	T	P	C
3	0	2	4

Pre-requisite: 25MDMCAEC05

Course Description:
This course builds upon Software Engineering principles and focuses on managing software projects effectively. It covers project planning, estimation, scheduling, risk management, team coordination, and quality management practices. The course equips students with the skills required to manage software development projects in dynamic and agile environments.

Course Objectives:
This course enables students to:

1. Understand project management concepts in software development.
2. Apply estimation and scheduling techniques for project planning.
3. Manage teams, resources, and communication effectively.
4. Analyze risks and implement mitigation strategies.
5. Apply quality and project monitoring practices for successful delivery.

UNIT I SOFTWARE PROJECT INITIATION AND PLANNING 9 Hours

Introduction to Software Project Management – Project Life Cycle – Project Initiation – Project Scope and Objectives – Feasibility Analysis (Technical, Economic, Operational) – Work Breakdown Structure (WBS) – Project Planning Process.

UNIT II PROJECT ESTIMATION AND SCHEDULING 9 Hours

Size Estimation Techniques (LOC, Function Points – basics) – Effort Estimation Models (COCOMO – overview) – Project Scheduling – Gantt Charts – PERT/CPM – Critical Path Analysis – Resource Allocation.

UNIT III PROJECT TEAM AND COMMUNICATION MANAGEMENT 9 Hours

Team Structure and Organization – Roles and Responsibilities – Leadership in Software Projects – Communication Planning – Stakeholder Management – Conflict Resolution – Motivation Techniques.

UNIT IV RISK MANAGEMENT AND PROJECT MONITORING 9 Hours

Risk Identification – Risk Analysis (Qualitative & Quantitative – overview) – Risk Mitigation Strategies – Project Monitoring and Control – Earned Value Management (EVM) – Status Reporting – Change Management.

UNIT V SOFTWARE QUALITY AND AGILE PROJECT MANAGEMENT 9 Hours

Quality Management Concepts – Quality Assurance vs Quality Control – Software Quality Metrics – Agile Project Management (Scrum roles, artifacts, sprints – management perspective only) – Project Closure and Evaluation.

List of Experiments **30 Hours**

1. Preparation of Software Project Proposal and Project Initiation Document
2. Conduct Feasibility Analysis and Develop Work Breakdown Structure (WBS)
3. Estimate Project Size using LOC and Function Point Methods
4. Perform Effort Estimation using Basic COCOMO Model
5. Develop Project Schedule using Gantt chart and PERT/CPM Techniques
6. Design Project Team Structure and Communication Management Plan
7. Prepare Risk Register and Perform Project Monitoring using Earned Value Management (EVM)
8. Implement Agile Project Management Process using Scrum and Perform Project Closure Evaluation

Course Outcomes:

Upon successful completion of the course, students will be able to

CO1: Explain software project management concepts and project life cycle.

CO2: Apply estimation and scheduling techniques for project planning.

CO3: Manage project teams and communication effectively.

CO4: Analyze project risks and monitor project performance.

CO5: Evaluate quality practices and agile project management approaches.

Text Books

1. Mark Winteringham, Software Testing with Generative AI, Manning Publications, 2024.
2. Gayathri Mohan, Full Stack Testing: A Practical Guide, O'Reilly Media, 2021.
3. Aditya P. Mathur, Foundations of Software Testing, 3rd Edition, Pearson, 2022.

Reference Books

1. James Bach & Michael Bolton, The Rapid Software Testing Approach, Addison-Wesley, 2023.

2. Marie Cruz & Lewis Prescott, *Contract Testing in Action*, Manning Publications, **2023**.

3. Vijay Kulkarni, *Practical Software Testing with Selenium and DevOps*, Packt Publishing, **2022**.

Mode of Evaluation: Assignments, Mid Term Tests, Continuous Internal Evaluation (Record) and End Semester Examination.