SARDAR PATEL UNIVERSITY, BALAGHAT

School of Computer Application

Syllabus (under NEP with Option 2)

Course: Master of Computer Applications (MCA) **Semester: III**

Branch: Computer Science and Application **w.e.f. Academic Session: 2025-26**

Subject: Foundations of Data Analysis and Big Data (MCA301)

Course Objectives

- To introduce the fundamentals of data analysis and its role in decision-making.
- To familiarize students with data types, data preprocessing, and visualization techniques.
- To provide an overview of big data concepts, tools, and technologies.
- To explore data storage, distributed computing, and Hadoop ecosystem.
- To develop basic skills in handling large datasets using open-source platforms.

Course Outcomes

After completing this course, students will be able to:

- CO1: Understand the principles and applications of data analysis.
- CO2: Perform data cleaning, transformation, and visualization.
- CO3: Explain the architecture and components of big data systems.
- CO4: Use Hadoop and related tools for distributed data processing.
- CO5: Apply basic big data techniques to analyze large datasets.

UNIT-I: Introduction to Data Analysis

What is data analysis? Importance and applications. Types of data: structured, unstructured, semi-structured. Data collection methods and sources. Introduction to data analytics lifecycle. Tools used in data analysis (Excel, Python, R – overview).

UNIT-II: Data Preprocessing and Visualization

Data cleaning: handling missing values, duplicates. Data transformation: normalization, encoding, scaling. Exploratory Data Analysis (EDA). Data visualization techniques: charts, graphs, dashboards. Introduction to tools: Matplotlib, Seaborn, Tableau (overview).

UNIT-III: Fundamentals of Big Data

Definition and characteristics of Big Data (Volume, Velocity, Variety, Veracity, Value). Challenges in big data processing. Traditional vs. big data systems. Introduction to NoSQL databases: MongoDB, Cassandra. Applications of big data in various domains.

UNIT-IV: Big Data Architecture and Hadoop Ecosystem

Big data architecture overview. Introduction to Hadoop: HDFS and MapReduce. Hadoop components: NameNode, DataNode, JobTracker, TaskTracker. Introduction to YARN and HDFS commands. Hadoop ecosystem tools: Hive, Pig, Sqoop, Flume (basic overview).

UNIT-V: Big Data Processing and Case Studies

Introduction to Spark and its components. Data ingestion and ETL processes. Real-world case studies using big data (e.g., healthcare, retail, finance). Hands-on with sample datasets using Hadoop or Spark. Ethical considerations and data privacy in big data.

Text Book

- Seema Acharya and Subhashini Chellappan, *Big Data and Analytics*, Wiley India.

Reference Books

- Rajaraman, *Fundamentals of Data Science*, PHI Learning.
- Tom White, *Hadoop: The Definitive Guide*, O'Reilly Media.
- Anand Rajaraman and Jeffrey D. Ullman, *Mining of Massive Datasets*, Cambridge University Press.
- Joel Grus, *Data Science from Scratch*, O'Reilly Media.

SARDAR PATEL UNIVERSITY, BALAGHAT

School of Computer Application

Syllabus (under NEP with Option 2)

Course: Master of Computer Applications (MCA) **Semester: III**

Branch: Computer Science and Application **w.e.f. Academic Session: 2025-26**

Subject: Software Engineering (MCA302)

Course Objectives

- To understand the fundamental concepts and lifecycle of software engineering.
- To learn various software development models and methodologies.
- To explore requirements analysis, design techniques, and software architecture.
- To study software testing, maintenance, and quality assurance.
- To introduce project management and software configuration control.

Course Outcomes

After completing this course, students will be able to:

- CO1: Explain the software development lifecycle and engineering principles.
- CO2: Apply appropriate software process models to real-world projects.
- CO3: Perform requirements analysis and design software systems.
- CO4: Conduct software testing and ensure quality assurance.
- CO5: Manage software projects using planning, estimation, and configuration tools.

UNIT-I: Introduction to Software Engineering

Definition and scope of software engineering. Software characteristics and applications. Software crisis and myths. Software development lifecycle (SDLC). Software process models: Waterfall, Incremental, Spiral, Agile.

UNIT-II: Requirements Engineering and Analysis

Requirements gathering and elicitation techniques. Functional and non-functional requirements. Use case modeling and requirement specification. Software requirement specification (SRS) document. Requirements validation and management.

UNIT-III: Software Design and Architecture

Design principles and concepts. Structured design and object-oriented design. Architectural styles: layered, client-server, microservices. Design patterns: Singleton, Factory, Observer (overview). User interface design and prototyping.

UNIT-IV: Software Testing and Quality Assurance

Testing fundamentals: verification vs validation. Types of testing: unit, integration, system, acceptance. Testing techniques: black-box, white-box. Test case design and test automation (overview). Software quality metrics and standards (ISO, CMMI).

UNIT-V: Software Project Management and Maintenance

Project planning and scheduling. Cost estimation models: COCOMO. Risk management and mitigation. Software configuration management. Software maintenance and evolution.

Text Book

- Ian Sommerville, *Software Engineering*, 10th Edition, Pearson Education.

Reference Books

- Pankaj Jalote, *An Integrated Approach to Software Engineering*, Springer.
- Rajib Mall, *Fundamentals of Software Engineering*, PHI Learning.
- Roger S. Pressman, *Software Engineering: A Practitioner's Approach*, McGraw Hill.
- Ghezzi, Jazayeri, and Mandrioli, *Fundamentals of Software Engineering*, PHI.

SARDAR PATEL UNIVERSITY, BALAGHAT

School of Computer Application

Syllabus (under NEP with Option 2)

Course: Master of Computer Applications (MCA) **Semester: III**

Branch: Computer Science and Application **w.e.f. Academic Session: 2025-26**

Subject: Python Programming (MCA303P)

Course Objectives

- To introduce the syntax, structure, and features of Python programming.
- To develop problem-solving skills using Python's built-in data types and control structures.
- To teach modular programming using functions and object-oriented principles.
- To explore file handling, exception management, and basic data structures.
- To provide hands-on experience with Python for real-world applications.

Course Outcomes

After completing this course, students will be able to:

- CO1: Write Python programs using variables, operators, and control structures.
- CO2: Use built-in data types and functions to solve computational problems.
- CO3: Apply object-oriented programming concepts in Python.
- CO4: Perform file operations and handle exceptions effectively.
- CO5: Implement basic data structures and develop small applications.

UNIT-I: Introduction to Python

History and features of Python. Installing Python and setting up the environment. Writing and executing Python scripts. Variables, data types, and type conversion. Operators: arithmetic, relational, logical, assignment.

UNIT-II: Control Structures and Functions

Conditional statements: if, if-else, nested if. Loops: for, while, nested loops. Loop control statements: break, continue, pass. Functions: definition, arguments, return values. Lambda functions and recursion.

UNIT-III: Data Structures in Python

Lists: creation, indexing, slicing, methods. Tuples and sets: operations and use cases. Dictionaries: key-value pairs, methods. Iterating over data structures. Comprehensions: list, set, dictionary.

UNIT-IV: Object-Oriented Programming and Modules

Classes and objects. Constructors and destructors. Inheritance and method overriding. Encapsulation and polymorphism. Importing and creating modules. Built-in modules: math, random, datetime.

UNIT-V: File Handling, Exceptions, and Applications

Reading and writing text files. Working with CSV files. Exception handling: try, except, finally. Creating custom exceptions. Introduction to Python libraries: NumPy, Pandas (overview). Mini project: simple application using Python.

Text Book

- Reema Thareja, *Python Programming*, Oxford University Press.

Reference Books

- Mark Lutz, *Learning Python*, O'Reilly Media.
- Charles Dierbach, *Introduction to Computer Science Using Python*, Wiley.
- Paul Barry, *Head First Python*, O'Reilly Media.
- Kenneth A. Lambert, *Fundamentals of Python: First Programs*, Cengage Learning.

Practical Program List:

Basics and Control Structures

1. Program to display “Hello, World!”
2. Program to perform arithmetic operations
3. Program to check if a number is even or odd
4. Program to find the largest of three numbers
5. Program to print multiplication table using loops

Functions and Recursion

6. Program to calculate factorial using recursion
7. Program to check if a number is prime
8. Program to generate Fibonacci series
9. Program to demonstrate lambda function
10. Program to find GCD of two numbers using function

Data Structures

11. Program to perform list operations (append, remove, sort)
12. Program to count frequency of elements in a tuple
13. Program to perform set operations (union, intersection)
14. Program to create and manipulate a dictionary
15. Program to use list comprehension for filtering numbers

OOP and Modules

16. Program to create a class and object
17. Program to demonstrate inheritance and method overriding
18. Program to use built-in module (math or datetime)
19. Program to create and import a user-defined module

File Handling and Exceptions

20. Program to read and write data to a text file with exception handling

SARDAR PATEL UNIVERSITY, BALAGHAT

School of Computer Application

Syllabus (under NEP with Option 2)

Course: Master of Computer Applications (MCA) **Semester: III**

Branch: Computer Science and Application **w.e.f. Academic Session: 2025-26**

Subject: Advance Web Technology (MCA304P)

Course Objectives

- To understand server-side scripting using PHP and database integration with MySQL.
- To develop dynamic and interactive web applications using ReactJS.
- To explore backend development using NodeJS and Express.
- To integrate front-end and back-end technologies for full-stack development.
- To provide hands-on experience in building modern web applications.

Course Outcomes

After completing this course, students will be able to:

- CO1: Develop dynamic web pages using PHP and connect them to MySQL databases.
- CO2: Design responsive and component-based user interfaces using ReactJS.
- CO3: Build RESTful APIs and server-side logic using NodeJS and Express.
- CO4: Integrate front-end and back-end technologies for complete web solutions.
- CO5: Deploy and manage full-stack applications with basic security and performance considerations.

UNIT-I: Server-Side Programming with PHP

Introduction to PHP: syntax, variables, data types. Control structures and loops. Functions and arrays. Working with forms and user input. Sessions, cookies, and file handling. Connecting PHP to MySQL using mysqli and PDO.

UNIT-II: Database Management with MySQL

Introduction to relational databases. MySQL data types and constraints. Creating, modifying, and querying tables. CRUD operations using SQL. Joins, subqueries, and indexing. Integrating MySQL with PHP for dynamic content.

UNIT-III: Front-End Development with ReactJS

Introduction to React and JSX. Components: functional and class-based. Props and state management. Event handling and conditional rendering. Lists and forms in React. React Router for navigation.

UNIT-IV: Backend Development with NodeJS and Express

Introduction to NodeJS and npm. Creating a server with NodeJS. Express framework: routing, middleware. Handling requests and responses. Connecting NodeJS to MySQL or MongoDB. Building RESTful APIs.

UNIT-V: Full-Stack Integration and Deployment

Connecting React front-end with NodeJS backend. API consumption in React using fetch/axios. Authentication basics (JWT overview). Error handling and debugging. Deployment using platforms like Heroku or Vercel. Mini project: Full-stack web application.

Text Book

- Luke Welling & Laura Thomson, *PHP and MySQL Web Development*, Pearson Education.

Reference Books

- Robin Nixon, *Learning PHP, MySQL & JavaScript*, O'Reilly Media.
- Ethan Brown, *Learning Node.js Development*, O'Reilly Media.
- Alex Banks & Eve Porcello, *Learning React*, O'Reilly Media.
- Brad Traversy, *Modern Full-Stack Development*, Packt Publishing.

Practical Program List:

PHP & MySQL

1. Create a PHP program to display user input from a form.
2. Build a login system using PHP and MySQL.
3. Create a registration form with validation and database storage.
4. Develop a CRUD application using PHP and MySQL.
5. Display records from a database in an HTML table.

ReactJS

6. Create a React app with multiple components.
7. Build a dynamic to-do list using React.
8. Implement a form in React and handle user input.
9. Use React Router to navigate between pages.
10. Fetch data from an API and display it in React.

NodeJS & Express

11. Create a simple HTTP server using NodeJS.
12. Build a RESTful API using Express.
13. Handle GET and POST requests in Express.
14. Connect NodeJS to MySQL and perform CRUD operations.
15. Implement middleware for logging requests.

Full-Stack Integration

16. Connect React front-end to NodeJS backend using fetch.
17. Build a full-stack login system with React and NodeJS.
18. Create a product catalog with React frontend and Express backend.
19. Implement basic JWT authentication in NodeJS.
20. Deploy a full-stack app to Heroku or Vercel.