SEMESTER-V

COURSE 11: SOFTWARE ENGINEERING

Theory **Credits: 3** **3 hrs/week**

Course Objectives:

1. Understand the fundamental principles of software engineering, including software development life cycle models and their practical applications.
2. Analyze and document software requirements through feasibility studies and specification techniques.
3. Apply software design principles and development standards for building reliable and maintainable systems.
4. Evaluate software testing methodologies, including design and execution of test cases for various testing levels.
5. Examine software maintenance strategies, types, and metrics to sustain software performance over time.

Course Outcomes

At the end of the course, students will be able to:

1. Compare and contrast software development life cycle models such as Waterfall, Spiral, and Agile, and explain their appropriate use cases.
2. Conduct requirement analysis and distinguish between functional and non-functional requirements to develop a Software Requirements Specification (SRS) document.
3. Utilize design principles like modularity, cohesion, and coupling; implement Data Flow Diagrams (DFDs), structure charts, and follow coding standards and version control practices.
4. Design and perform different types of software tests—white-box, black-box, unit, integration, system—and differentiate between manual and automated testing approaches.
5. Categorize software maintenance types and propose strategies based on software maintenance metrics for effective long-term software sustainability.

Unit 1: Software Engineering Foundations and Requirements Engineering:
Definition of Software engineering, Software life cycle models: Waterfall, prototyping, Evolutionary, Spiral and Agile models. Comparison among development life cycles.

Unit 2: Requirement Analysis and Specification
Feasibility study, Requirements gathering, Functional and Non-functional requirements, Requirements analysis and specification, design of software requirement specification (SRS).

Unit 3: Software Design Principles and Development Practices
Introduction to software design, modularity, cohesion, coupling and layering, functional design, and solution design, use of DFD and Structure chart in software design, UML in software design, user interface design. Software Development Basics, Coding standards, Version Control and Code review techniques.

Unit 4: Software Testing
Fundamentals of testing (verification and validation), White-box, and black-box testing, unit testing, integration testing, system testing, acceptance testing (alpha testing and beta testing), test scenarios and test case design, automation testing and manual testing.

Unit 5: Software Maintenance
Introduction to software maintenance, corrective maintenance, perfective maintenance, adaptive maintenance, preventive maintenance, challenges in software maintenance, metrics related to software maintenance.

Textbooks:

1. Software Engineering: A Practitioner's Approach, Roger Pressman, McGraw Hill, 6th Edition
2. Software Engineering, Sommerville, Adison Wesley, 7th edition.

Reference Books:

1. Fundamentals of Software Engineering, Mall Rajib,PHI, Fifth Edition,
2. Fundamentals of Software Engineering, Hitesh, BPB Publications

Activities:

Outcome: Compare and contrast software development life cycle models such as Waterfall, Spiral, and Agile, and explain their appropriate use cases.

Activity: Create a comparison chart in groups showing key features, pros/cons, and use cases of Waterfall, Spiral, and Agile models. Include a real-world example for each.

Evaluation Method: Use a rubric to assess on a 10-point scale to check:

- Accuracy of model descriptions
- Clarity of comparison
- Relevance of examples
- Presentation skills (if shared in class)

Outcome: Conduct requirement analysis and distinguish between functional and non-functional requirements to develop a Software Requirements Specification (SRS) document.

Activity: Analyze a simple system (e.g., online library or food ordering app). Identify 5 functional and 3 non-functional requirements. Draft a mini-SRS document using a template.

Evaluation Method: Checklist-based review on a 10-point scale:

- Correct classification of requirements
- Completeness of SRS sections
- Clarity and formatting
- Use of standard terminology

Outcome: Utilize design principles like modularity, cohesion, and coupling; implement Data Flow Diagrams (DFDs), structure charts, and follow coding standards and version control practices.

Activity: Design a basic login system using DFD (Level 0 and Level 1) and a structure chart. Highlight modules and discuss cohesion and coupling in pairs.

Evaluation Method: Peer review and instructor feedback on a 10-point scale:

- Correct use of DFD symbols
- Logical flow and modularity
- Explanation of cohesion/coupling
- Neatness and labelling

Outcome: Design and perform different types of software tests—white-box, black-box, unit, integration, system—and differentiate between manual and automated testing approaches.

Activity: Write simple test cases for a calculator app (e.g., addition, division by zero). Perform manual unit testing and simulate black-box and white-box testing.

Evaluation Method: Observation and worksheet to assess on a 10-point scale:

- Correct identification of test types
- Clear test case steps and expected output
- Execution and result recording
- Understanding of manual vs automated testing

Outcome: Categorize software maintenance types and propose strategies based on software maintenance metrics for effective long-term software sustainability.

Activity: Categorize sample maintenance tasks (e.g., fixing a bug, adding a feature) into corrective, adaptive, perfective, or preventive. Discuss sustainability strategies in small groups.

Evaluation Method: Group activity score (10-point scale):

- Correct classification of maintenance types
- Participation in discussion
- Suggestions for sustainability (e.g., documentation, modular design)
- Use of basic metrics (e.g., number of bugs fixed)

SEMESTER-V

COURSE 11: SOFTWARE ENGINEERING

Practical **Credits: 1** **2 hrs/week**

Select domain of interest (e.g. College Management System) and identify multi-tier software applications to work on (e.g. Online Fee Collection). Analyze, design and develop this application:

1. Develop an IEEE standard SRS document. Also develop risk management and project plan (Gantt chart).
2. Understanding of System modeling: Data model i.e. ER – Diagram and draw the ER Diagram with generalization, specialization and aggregation of specified problem statements.
3. Understanding of System modeling: Functional modeling: DFD Context and draw it
4. Understanding of System modeling: UML and draw it.
5. Develop a sample calculator program and perform Black-box Testing:
    a. Identify test scenarios without viewing the internal code.
    b. Write test cases for valid and invalid inputs.
    c. Execute the test cases and record outcomes.
6. Develop a sample login authentication page and perform White--box Testing**:**
    a. Analyze the code for all possible paths, conditions, and loops.
    b. Apply statement coverage and branch coverage techniques.
    c. Test internal functions with known inputs.
7. For the above login authentication page, perform the following:
    a. Simulate the following maintenance activities:
        - **Corrective Maintenance**: Fix an existing bug (e.g., wrong output, crash, miscalculation).
        - **Perfective Maintenance**: Add a new user-requested feature (e.g., sorting, filter, or improved UI).
        - **Adaptive Maintenance**: Modify the code to adapt to a new platform or library version.
        - **Preventive Maintenance**: Refactor the code to improve readability, performance, or security.
    b. Document each change with:
        - Problem description
        - Type of maintenance
        - Before and after screenshots
        - Change summary

SEMESTER-V

COURSE 12 A: WEB INTERFACE DESIGNING TECHNOLOGIES

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. Understand the principles of web design and distinguish between web and desktop application architectures. Develop static web pages using HTML elements, attributes, and multimedia integration techniques.
2. Style web pages effectively using CSS, including layout control, responsive design, and UI enhancements.
3. Implement dynamic behaviors and form validations using JavaScript and the Document Object Model (DOM).
4. Explore client-side scripting techniques to build responsive, interactive interfaces.
5. Gain foundational knowledge of Content Management Systems (CMS) and apply practical skills in platforms such as WordPress.

Course Outcomes

At the end of the course, students will be able to:

1. Design and structure HTML-based webpages incorporating text, images, tables, forms, and multimedia content.
2. Apply CSS styling rules to manage layout aesthetics, interactivity, and responsiveness across devices.
3. Use JavaScript for string manipulation, event handling, arrays, object operations, and basic validation.
4. Employ client-side scripting to enhance form functionality, create dialog interactions, and add animation via mouse and keyboard events.
5. Analyze different types of CMS and operate WordPress features like posts, pages, themes, plug-ins, and SEO tools to create and deploy basic websites.

Unit 1.HTML:
Introduction to web designing, difference between web applications and desktop applications, introduction to HTML, HTML structure, elements, attributes, headings, paragraphs, images, tables, lists, blocks, symbols, embedding multi-media components in HTML, HTML forms

Unit 2.CSS:
CSS home, introduction, syntax, CSS combinators, colors, background, borders, margins, padding, height/width, text, fonts, tables, lists, position, overflow, float, pseudo class, pseudo elements, opacity, tool tips, image gallery, CSS forms, CSS counters.

Unit 3.Java Script:
What is DHTML, JavaScript, basics, variables, operators, statements, string manipulations, mathematical functions, arrays, functions. objects, regular expressions, exception handling.

Unit 4. Client-Side Scripting:
Accessing HTML form elements using Java Script object model, basic data validations, data format validations, generating responsive messages, opening windows using java script, different kinds of dialog boxes, accessing status bar using java script, embedding basic animative features using different keyboard and mouse events.

Unit 5. Content Management Systems
Introduction to CMS: What is a CMS?, Types: traditional, headless, cloud-based

Popular CMS Platforms: WordPress, Joomla, Drupal, Shopify, When to choose each

Wordpress Basics: Introduction to word press, features, and advantages, wordpress (hosted access and local access), working with posts, managing pages, working with media, working with widgets, working with themes, extending wordpress with plug-ins, SEO and deployment.

Text Book(s)

1. Web Programming Building Internet Applications, Chris Bates, Second Edition, Wiley
2. An Introduction to Web Design plus Programming, Paul S.WangSanda S. Katila, Thomson.

Reference Books

1. Head First HTML and CSS, Elisabeth Robson, Eric Freeman, O'Reilly Media Inc.
2. An Introduction to HTML and JavaScript: for Scientists and Engineers, David R. Brooks. Springer, 2007
3. Schaum's Easy Outline HTML, David Mercer, Mcgraw Hill Professional.
4. Wordpress for Beginners, Dr.Andy Williams

Activities:

Outcome: Design and structure HTML-based webpages incorporating text, images, tables, forms, and multimedia content.

Activity: Create a personal profile webpage using HTML that includes:

- Text (headings, paragraphs)
- Images
- Tables (e.g., contact info or skills)
- Forms (e.g., feedback form)
- Embedded multimedia (e.g., YouTube video or audio clip)

Evaluation Method: Checklist-based review on a 10-point scale:

- Correct use of HTML tags
- Proper structure and nesting
- Functionality of form and media elements
- Visual clarity and completeness

Outcome: Apply CSS styling rules to manage layout aesthetics, interactivity, and responsiveness across devices.

Activity: Style the personal profile page using CSS:

- Apply colors, fonts, spacing
- Use Flexbox or Grid for layout
- Add hover effects and media queries for responsiveness

Evaluation Method: Rubric-based assessment on a 10-point scale:

- Visual appeal and consistency
- Responsive behaviour on different screen sizes
- Use of selectors and layout techniques
- Code cleanliness and organization

Outcome: Use JavaScript for string manipulation, event handling, arrays, object operations, and basic validation.

Activity: Enhance the profile page with JavaScript:

- Validate form inputs (e.g., email format)
- Display a greeting using string manipulation
- Use arrays/objects to store and display hobbies or skills
- Handle click events on button

Evaluation Method: Code demonstration and output testing (10-point scale):

- Correct use of JS syntax and logic
- Functionality of validation and event handling
- Use of arrays/objects
- Console output or alert behaviour

Outcome: Employ client-side scripting to enhance form functionality, create dialog interactions, and add animation via mouse and keyboard events.

Activity: Add interactive features to the form:

- Show/hide sections using mouse events
- Trigger animations on keypress
- Display confirmation dialog on form submission

Evaluation Method: Live demo and observation (10-point scale):

- Smooth interaction and feedback
- Correct use of event listeners
- Animation and dialog behavior
- User experience quality

Outcome: Analyze different types of CMS and operate WordPress features like posts, pages, themes, plug-ins, and SEO tools to create and deploy basic websites.

Activity: Create a basic website using WordPress:

- Add posts and pages
- Apply a theme and customize layout
- Install plug-ins (e.g., contact form, SEO tool)
- Configure basic SEO settings

Evaluation Method: Site walkthrough and checklist (10-point scale):

- Functionality of posts/pages
- Theme customization
- Deployment readiness

SEMESTER-V

COURSE 12 A: WEB INTERFACE DESIGNING TECHNOLOGIES

Practical **Credits: 1** **2 hrs/week**

List of Experiments:

1. Create an HTML document with the following formatting options:

   (a) Bold, (b) Italics, (c) Underline, (d) Headings (Using H1 to H6 heading styles), (e) Font (Type, Size and Color), (f) Background (Colored background/Image in background), (g) Paragraph, (h) Line Break, (i) Horizontal Rule, (j) Pre tag

2. Create an HTML document which consists of:

   (a) Ordered List (b) Unordered List (c) Nested List (d) Image

3. Create a Table with four rows and five columns. Place an image in one column.

4. 4.Collect any ten images of your choice. Using table tag, align the images as follows:

5. Create a menu form using HTML.

6. Style the menu buttons using CSS.

7. Create a form using HTML which has the following types of controls:

   (a) Text Box (b) Option/radio buttons (c) Check boxes (d) Reset and Submit buttons

8. Embed a calendar object in your web page.

9. Create a form that accepts the information from the subscriber of a mailing system.

Word press:

10. Installation and configuration of word press
11. Access admin panel and manage posts
12. Access admin panel and manage pages
13. Add widgets and menus
14. Create users and assign roles
15. Create a site and add a theme to it

SEMESTER-V

COURSE 12 B: DATA SCIENCE WITH R

Theory **Credits: 3** **3 hrs/week**

Course Objectives

1. Introduce key mathematical and statistical tools essential for data analysis.
2. Explain the Data Science process, lifecycle, and its applications in diverse domains.
3. Develop proficiency in R programming for data manipulation and basic analytics.
4. Teach effective data handling, transformation, and visualization techniques using R.
5. Explore practical use cases in Data Science with modeling, clustering, and ethical awareness.

Course Outcomes

At the end of the course, students will be able to:

1. Apply statistical methods and probability distributions to analyze and interpret data.
2. Describe the Data Science workflow and perform exploratory data analysis (EDA).
3. Use R programming constructs and libraries for data input, control flows, and functions.
4. Handle and clean datasets using R tools like dplyr, tidyr, and manage missing/time-based data.
5. Implement basic machine learning models and evaluate performance using appropriate metrics and visual tools.

Unit 1. Mathematical & Statistical Foundations:
Sets, Functions, Probability, Random Variables, Descriptive Statistics: Mean, Median, Mode, Variance, Standard Deviation, Probability Distributions: Binomial, Normal, Poisson, Hypothesis Testing: t-test, Chi-square test, Correlation & Regression

Unit 2. Introduction to Data Science Process:
Introduction- Definition - Data Science in various fields - Examples - Impact of Data Science - Data Analytics Life Cycle - Data Science Toolkit - Data Scientist - Data Science Team, Exploratory Data Analysis (EDA), Feature Engineering & Data Transformation

Unit 3. Basics of R Programming:
Introduction to R and RStudio, Data Types, Variables, Operators, Control Structures (if, loops), Functions and Packages, Data Input/Output (CSV, Excel, XML, JSON).

Unit 4. Data Handling & Visualization in R:
Data Frames, Lists, Matrices, dplyr and tidyr for Data Wrangling, Handling Missing Data, Working with Date/Time in R.

Unit 5. Applications & Case Studies in Data Science:
Simple Linear Regression, Model Evaluation: Accuracy, Confusion Matrix, ROC.

K-Means Clustering, Text Mining & Word Clouds, Recommender Systems Basics, Ethical Issues in Data Science

Textbooks

1. An Introduction to Statistical Learning with Applications in R, Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, Springer, 2nd Edition, 2021
2. R for Data Science, Hadley Wickham and Garrett Grolemund, O'Reilly Media, 2017.

Reference Books

1. The Art of R Programming, Norman Matloff,, No Starch Press, 2011.
2. Modern Applied Statistics with S, W.N. Venables & B.D. Ripley, Springer, 2002.
3. Introduction to Data Science: Data Analysis and Prediction Algorithms with R, Rafael A. Irizarry, CRC Press, 2020.
4. Data Science from Scratch: First Principles with Python (for conceptual clarity only), Joel Grus,

Activities:

Outcome: Apply statistical methods and probability distributions to analyze and interpret data.

Activity: Analyze a dataset (e.g., student scores or sales data) to:

- Calculate mean, median, mode, variance, and standard deviation
- Fit and visualize a normal distribution
- Apply probability rules to answer questions (e.g., likelihood of scoring above 80)

Evaluation Method: Worksheet-based assessment (10-point scale):

- Accuracy of statistical calculations
- Correct use of probability formulas
- Interpretation of results and distribution plots

Outcome: Describe the Data Science workflow and perform exploratory data analysis (EDA).

Activity: Use a real-world dataset (e.g., Titanic or COVID data) to:

- Outline the steps of the Data Science workflow
- Perform EDA using summary statistics and visualizations (histograms, boxplots, scatterplots)

Evaluation Method: Presentation and checklist (10-point scale):

- Clear explanation of workflow stages
- Quality of EDA insights
- Use of appropriate plots and summaries

Outcome: Use R programming constructs and libraries for data input, control flows, and functions.

Activity: Write an R script that:

- Reads a CSV file
- Uses if, for, and while loops
- Defines and calls custom functions with arguments and return values

Evaluation Method: Code review and execution test to verify (10-point scale):

- Correctness of the syntax and logic
- Functionality of control structures
- Output accuracy and modularity

Outcome: Handle and clean datasets using R tools like dplyr, tidyr, and manage missing/time-based data.

Activity: Clean a messy dataset using:

- dplyr for filtering, selecting, and mutating
- tidyr for reshaping and handling missing values
- Time-based operations (e.g., filling gaps, formatting dates)

Evaluation Method: Before-and-after comparison (10 point score):

- Completeness of cleaning steps
- Use of appropriate functions
- Handling of missing/time data

Outcome: Implement basic machine learning models and evaluate performance using appropriate metrics and visual tools.

Activity: Build a simple classification model (e.g., logistic regression or decision tree) using R:

- Train/test split
- Predict outcomes
- Evaluate using confusion matrix, accuracy, precision, recall

Evaluation Method: Model report and demo (10 point scale):

- Correct implementation of model
- Use of evaluation metrics

SEMESTER-V

COURSE 12 B: DATA SCIENCE WITH R

Practical **Credits: 1** **2 hrs/week**

List of Practicals:

1. Compute Mean, Median, Mode, Variance, and Standard Deviation
2. Visualize Binomial, Normal, and Poisson Distributions
3. Perform t-test and Chi-Square Test in R
4. Calculate Correlation and Build a Simple Linear Regression Model
5. Conduct Exploratory Data Analysis (EDA) on a Real-World Dataset
6. Apply Feature Engineering: Scaling, Normalization, and Encoding
7. Practice R Programming: Variables, Control Structures, and Functions
8. Read and Write Data from CSV, Excel, JSON, and XML Files
9. Use dplyr and tidyr for Data Wrangling Tasks
10. Handle Missing Data and Detect Outliers
11. Work with Dates and Times in R
12. Visualize Data Using ggplot2 (Bar, Scatter, Histogram, Boxplot)
13. Perform K-Means Clustering and Visualize Clusters
14. Evaluate Models Using Confusion Matrix, Accuracy, and ROC Curve
15. Perform Text Mining and Create a Word Cloud

SEMESTER-V

COURSE 13 A: WEB APPLICATION DEVELOPMENT USING PHP & MySQL

Theory **Credits: 3** **3 hrs/week**

Course Objectives:

1. Understand the foundational elements of PHP, including variables, data types, operators, and flow control functions.
2. Develop proficiency in managing arrays, objects, strings, dates, and time functionalities using PHP.
3. Design and process HTML forms integrated with PHP, including advanced operations like file uploads, redirection, and exception handling.
4. Implement session management and cookie handling to preserve user state and provide secure, personalized experiences.
5. Connect PHP with MySQL databases, enabling learners to perform CRUD operations and build dynamic web applications.

Course Outcomes

At the end of the course, students will be able to:

1. Demonstrate effective use of PHP building blocks such as variables, expressions, constants, control structures, and functions with appropriate scope and argument handling.
2. Manipulate complex data structures including arrays and objects, and utilize PHP string and date/time functions for dynamic content generation.
3. Create functional web forms using PHP, retrieve form inputs, manage file uploads, and execute form-based operations like redirection and email dispatch.
4. Apply secure techniques for maintaining user sessions and cookies, including session lifecycle control, session variable manipulation, and user authentication workflows.
5. Integrate PHP with MySQL to build database-driven web components, perform record management, and architect structured menu-based data operations.

Unit 1. The building blocks of PHP:
Variables, Data Types, Operators and Expressions, Constants.

Flow Control Functions in PHP: Switching Flow, Loops, Code Blocks and Browser Output.

Working with Functions: Creating functions, Calling functions, Returning the values from User-Defined Functions, Variable Scope, Saving state between Function calls with the static statement, arguments of functions

Unit 2.Working with Arrays:
Creating Arrays, Some Array-Related Functions.

Working with Objects: Creating Objects, Accessing Object Instances,

Working with Strings, **Dates and Time:** Formatting strings with PHP, Manipulating Strings with PHP, Using Date and Time Functions in PHP.

Unit 3.Working with Forms:
Creating Forms, Accessing Form Input with User defined Arrays, Combining HTML and PHP code on a single Page, Using Hidden Fields to save state, Page redirection, Sending Mail on Form Submission, **Working with File Uploads**, Managing files on server, **Exception handling.**

Unit 4. Working with Cookies and User Sessions:
Introducing Cookies, setting a Cookie with PHP, Session Function Overview, starting a Session, working with session variables, passing session IDs in the Query String, Destroying Sessions and Unsetting Variables, Using Sessions in an Environment with Registered Users.

Unit 5.Interacting with MySQL using PHP:
MySQL Versus MySQLi Functions, connecting to MySQL with PHP, Working with MySQL Data. Planning and Creating Database Tables, Creating Menu, Creating Record Addition Mechanism, Viewing Records, Creating the Record Deletion Mechanism.

Text Book(s)

1. SAMS Teach yourself PHP MySQL and Apache,, Julie C. Meloni, Pearson Education
2. PHP: The Complete Reference, Steven Holzner , McGraw-Hill

Reference Books

1. Learning PHP, MySQL, JavaScript, CSS & HTML5, Robin Nixon, Third Edition, O'reilly, 2014
2. The web warrior guide to Web Programming, Xue Bai Michael Ekedahl,, Thomson, 2006.

Activities:

Outcome: Demonstrate effective use of PHP building blocks such as variables, expressions, constants, control structures, and functions with appropriate scope and argument handling.

Activity: Write a PHP script that:

- Declares variables and constants
- Uses expressions and control structures (if, switch, loops)
- Defines and calls functions with arguments and return values
- Demonstrates variable scope (global, local)

Evaluation Method: Evaluate on a 10-point scale based on code review checklist to verify the

- Correct syntax and use of each building block
- Logical flow using control structures
- Proper function definition and scope handling
- Output accuracy and readability

Outcome: Manipulate complex data structures including arrays and objects, and utilize PHP string and date/time functions for dynamic content generation.

Activity: Create a PHP script that:

- Stores student data in arrays and objects
- Formats and manipulates strings (e.g., name formatting)
- Displays current date/time and calculates age from DOB

Evaluation Method: Rubric-based assessment on a 10-point scale to check the:

- Correct use of arrays and objects
- Effective string and date/time functions
- Dynamic output generation
- Code clarity and structure

Outcome: Create functional web forms using PHP, retrieve form inputs, manage file uploads, and execute form-based operations like redirection and email dispatch.

Activity: Build a contact form that:

- Accepts name, email, message
- Uploads a file (e.g., resume)
- Sends an email confirmation
- Redirects to a thank-you page

Evaluation Method: Evaluate on a 10-point scale based on functional testing to perform:

- Form input retrieval and validation
- File upload success
- Email dispatch and redirection
- Error handling and user feedback

Outcome: Apply secure techniques for maintaining user sessions and cookies, including session lifecycle control, session variable manipulation, and user authentication workflows.

Activity: Develop a login system that:

- Starts a session on login
- Stores user data in session variables
- Sets a cookie for Remember Me
- Logs out and destroys session securely

Evaluation Method: Security checklist (10-point scale):

- Session lifecycle control
- Cookie setup with secure flags
- Authentication logic
- Session/cookie cleanup on logout

Outcome: Integrate PHP with MySQL to build database-driven web components, perform record management, and architect structured menu-based data operations.

Activity: Create a student management system:

- Connect to MySQL database
- Add, view, update, delete student records
- Display menu-based navigation (e.g., by class or grade)

Evaluation Method: Database interaction test on a 10-point scale:

- Successful CRUD operations
- Structured menu navigation
- SQL query correctness

SEMESTER-V

COURSE 13 A: WEB APPLICATION DEVELOPMENT USING PHP & MySQL

Practical **Credits: 1** **2 hrs/week**

List of Experiments:

1. Write a PHP program to Display Hello
2. Write a PHP Program to display today's date.
3. Write a PHP program to display Fibonacci series.
4. Write a PHP Program to read the employee details.
5. Write a PHP program to prepare the student marks list.
6. Create student registration form using text box, check box, radio button, select, submit button. And display user inserted values in the new PHP page.
7. Create Website Registration Form using text box, check box, radio button, select, submit button. And display user inserted values in the new PHP page.
8. Write a PHP script to demonstrate passing variables with cookies.
9. Write a PHP script to connect to the MySQL server from your website.
10. Write a program to keep track of how many times a visitor has loaded the page.
11. Write a PHP application to perform CRUD (Create, Read, Update and Delete) operations on a database table.
12. Create a web site using any open-source framework built on PHP and MySQL – It is a team activity wherein students are divided into multiple groups and each group comes up with their own website with basic features.

SEMESTER-V

COURSE 13 B: PYTHON FOR DATA SCIENCE

Theory **Credits: 3** **3 hrs/week**

Course Objectives:

1. Introduce foundational concepts of NumPy arrays and array operations for efficient numerical computing.
2. Teach key data structures and manipulation techniques using Pandas.
3. Enable students to perform data input/output operations and implement basic data cleaning workflows.
4. Explore string processing methods and feature engineering strategies in Pandas.
5. Guide learners in advanced data wrangling tasks including merging, reshaping, and hierarchical indexing.

Course Outcomes:

1. Demonstrate proficiency in creating and manipulating NumPy arrays for mathematical operations and simulations.
2. Apply Pandas Series and DataFrame operations for structured data handling and analysis.
3. Read, write, and clean diverse data formats using Python tools, addressing missing values and outliers.
4. Implement vectorized string operations and create derived features for enhanced model readiness.
5. Perform complex data wrangling tasks such as merging datasets, reshaping data structures, and generating group-level statistics.

Unit 1. NumPy Essentials:
NumPy ndarray: A Multidimensional Array Object, Creating ndarrays, Data Types for ndarrays, Arithmetic with Arrays, Basic Indexing and Slicing, Boolean Indexing, Fancy Indexing, Transposing Arrays, Swapping Axes, Universal Functions: Element-wise Operations, Basic Mathematical and Statistical Functions, Random Number Generation (basic use)

Unit 2. Pandas Basics and Data Structures:
Series, DataFrame, Index objects, Indexing and Selection, Filtering and Boolean Indexing, Arithmetic and Data Alignment, Sorting and Ranking, Dropping Entries, Handling Duplicate Indexes

Unit 3. Data Input, Output, and Cleaning:
Reading and Writing Data in Text Format (CSV, TXT), Working with JSON, Reading Microsoft Excel Files, Handling Missing Data, Dropping and Filling Missing Values, Replacing Values, Renaming Axis Indexes, Removing Duplicates, Filtering Outliers, Transforming Data Using Mapping or Functions

Unit 4. String Operations and Feature Engineering:
String Methods in pandas, Basic Regular Expressions, Vectorized String Functions, Creating Dummy/Indicator Variables, Permutation and Random Sampling.

Unit 5. Data Wrangling and Reshaping:
Merging and Joining Datasets, Concatenating Along an Axis, Combining Data with Overlap, Reshaping with Pivot, Stack, and Unstack, Basic Hierarchical Indexing, Summary Statistics by Group or Level

Textbooks

1. Python for Data Analysis: Data Wrangling with pandas, NumPy, and Jupyter, Wes McKinney, 3rd Edition, O'Reilly Media, 2022.
2. Python for Data Science For Dummies, Yuli Vasiliev, 2nd Edition, Wiley, 2022.

Reference Books

1. Python Data Science Handbook: Essential Tools for Working with Data, Jake VanderPlas, 2nd Edition, O'Reilly, 2022.

2. Introduction to Machine Learning with Python, Andreas Müller & Sarah Guido, O'Reilly Media, Reprint Edition, 2023.

3. Foundations for Analytics with Python: From Non-programmer to Hacker, Clinton Brownley, 2nd Edition, Pearson, 2020.

Activities:

Outcome: Demonstrate proficiency in creating and manipulating NumPy arrays for mathematical operations and simulations.

Activity: Create a NumPy-based simulation:

- Generate a 2D array representing temperature data over time
- Apply mathematical operations (mean, std, element-wise addition)
- Simulate random noise and visualize the effect

Evaluation Method: Code-based assessment (10-point scale):

- Correct use of np.array, np.random, and math functions
- Accuracy of simulation logic
- Output clarity and reproducibility

Outcome: Apply Pandas Series and DataFrame operations for structured data handling and analysis.

Activity: Analyze a CSV dataset (e.g., sales or COVID data):

- Load into a DataFrame
- Perform Series operations (filter, map, value_counts)

- Apply DataFrame methods (groupby, sort, describe)

Evaluation Method: 10-point scale checklist and peer review to verify:

- Proper use of Series vs DataFrame
- Logical data manipulation
- Insightful summary statistics

Outcome: Read, write, and clean diverse data formats using Python tools, addressing missing values and outliers.

Activity: Work with multiple formats:

- Read CSV, Excel, and JSON files
- Identify and handle missing values (dropna, fillna)
- Detect and treat outliers using IQR or Z-score

Evaluation Method: Rubric-based evaluation to check (10-point scale):

- File handling accuracy
- Cleaning completeness
- Outlier detection logic

Outcome: Implement vectorized string operations and create derived features for enhanced model readiness.

Activity: Prepare text data for modelling to:

- Use str methods to clean and standardize strings
- Extract features (e.g., domain from email, length of name)
- Encode categorical variables (e.g., get_dummies, LabelEncoder)

Evaluation Method: Feature report to check (10-point scale):

- Efficiency of vectorized operations
- Relevance of derived features
- Readiness for ML input

Outcome: Perform complex data wrangling tasks such as merging datasets, reshaping data structures, and generating group-level statistics.

Activity: Integrate and reshape datasets:

- Merge two datasets on a common key
- Reshape using pivot, melt, stack, unstack
- Generate group-level stats (e.g., mean sales per region)

Evaluation Method: Before-and-after comparison to validate:

- Accuracy of merge and reshape
- Correct use of aggregation
- Final structure suitability for analysis

SEMESTER-V

COURSE 13 B: PYTHON FOR DATA SCIENCE

Practical **Credits: 1** **2 hrs/week**

List of Practicals:

1. Create and Manipulate NumPy ndarrays; Explore Data Types
2. Perform Arithmetic Operations and Element-wise Calculations on Arrays
3. Practice Indexing, Slicing, Boolean, and Fancy Indexing on ndarrays
4. Use Universal Functions and Compute Basic Mathematical/Statistical Functions with NumPy
5. Create and Manipulate Pandas Series and DataFrames
6. Perform Indexing, Selection, Filtering, and Boolean Indexing in Pandas
7. Conduct Arithmetic Operations and Data Alignment in DataFrames
8. Sort, Rank, Drop Entries and Handle Duplicate Indexes in Pandas
9. Read and Write Data in CSV, TXT, JSON, and Excel Formats
10. Handle Missing Data: Detect, Drop, Fill, and Replace Missing Values
11. Rename Axis Indexes, Remove Duplicates, and Filter Outliers
12. Transform Data Using Mapping Functions and Apply String Operations
13. Perform String Operations and Use Regular Expressions on DataFrames
14. Create Dummy Variables and Perform Permutations and Random Sampling
15. Merge, Join, and Concatenate Datasets Using Pandas
16. Reshape Data Using Pivot, Stack, Unstack, and Perform Hierarchical Indexing
17. Compute Summary Statistics Grouped by Levels or Categories