Master of Computer Applications (MCA)

Web Technology Lab (DMCACO208P24)

Self-Learning Material (SEM II)

Jaipur National University
Centre for Distance and Online Education

Established by Government of Rajasthan
Approved by UGC under Sec 2(f) of UGC ACT 1956
&
NAAC A+ Accredited

TABLE OF CONTENTS

COURSE INTRODUCTION

This course is meticulously designed to provide a comprehensive overview of web development, starting from the foundational concepts to advanced programming techniques. Structured into five units, it spans the breadth of essential technologies and methodologies crucial for effective web design and dynamic web applications.

It delves into the functionalities of HTTP protocol focusing on its request-response pattern, which underpins web communication. Students will explore web browsers and servers, the pivotal components of web interactions.

It covers everything from simple formatting, links, lists, and tables to the more complex forms and frames, alongside an introduction to XHTML and HTML5. Moving forward, the course introduces Cascading Style Sheets (CSS) which is essential for customizing the appearance of web content. Students learn to manipulate everything from backgrounds, fonts, and text to mastering layout control with margins and positioning. The evolution from basic CSS to CSS2 and the latest CSS3 features are thoroughly explored to enhance the stylistic and responsive capabilities of web design.

The client-side scripting using JavaScript, an essential skill for adding interactive elements to web pages. Topics covered include variables, functions, loops, conditions, and event handling, along with JavaScript’s interaction with the Document Object Model (DOM). Advanced JavaScript topics delve into object-oriented aspects and DOM manipulations. This unit also covers XML, its applications, and key components including DTDs, schemas, and the transformation of XML data using XSL and XSLT, integrating these with JavaScript to create dynamic HTML content.

A powerful server-side scripting language used for developing dynamic and interactive websites. Starting from the basic syntax, it covers control structures, data types, functions, and form processing. Advanced topics include cookies, sessions, and object-oriented programming in PHP.

It focuses on integrating PHP with MySQL, a popular database management system. It teaches students how to perform fundamental database operations such as creating, querying, and modifying databases and tables through PHP. This includes hands-on experience with PHPMyAdmin, an administration tool for MySQL databases, and handling common database bugs.

Course Outcomes:

At the completion of the course, a student will be able to:

1. Explain the history of the internet and related internet concepts that are vital in understanding web development.
2. Discuss the insights of internet programming and implement complete application over the web.
3. Demonstrate the important HTML tags for designing static pages and separate design from content using Cascading Style sheet.
4. Use web application development software tools i.e. Ajax, PHP and XML etc. and identify the environments currently available on the market to design web sites.

Acknowledgements:

The content we have utilized is solely educational in nature. The copyright proprietors of the materials reproduced in this book have been tracked down as much as possible. The editors apologize for any violation that may have happened, and they will be happy to rectify any such material in later versions of this book.

Question 1: Responsive Portfolio Website

Program Statement:

Create a responsive portfolio website that displays your work and skills. It should look good on all devices, from desktops to mobile phones.

Solution Hints:

- Use HTML5 and CSS3 to structure and style the website.
- Implement media queries in CSS to handle different screen sizes and orientations.
- Incorporate a grid system or flexbox for layout adjustments.
- Optionally, add animations or transitions for a dynamic user experience.

Question 2: Dynamic Recipe Book

Program Statement:

Develop a dynamic recipe book where users can add, remove, and browse recipes. Include features like categorization and a search function.

Solution Hints:

- Use HTML forms for input tasks.
- Implement JavaScript to dynamically add and remove recipe entries without reloading the page.
- Utilize local storage or a server-side database to store recipes.
- Add a search function using JavaScript to filter through the recipe titles or contents.

Question 3: Weather Dashboard Application

Program Statement:

Build a weather dashboard that allows users to enter a city name and retrieve current weather information using an external API.

Solution Hints:

- Use HTML and CSS for the frontend layout.
- Fetch weather data from a public API like OpenWeatherMap using AJAX requests in JavaScript.
- Display the weather data dynamically in the HTML document.
- Handle errors gracefully, such as displaying a message when the city is not found.

Question 4: Real-time Chat Application

Program Statement:

Create a simple real-time chat application using WebSockets where users can send and receive messages instantly.

Solution Hints:

- Use Node.js and Express for the server-side.
- Implement WebSocket communication with a library like socket.io.
- Design a basic front-end interface for sending and receiving messages.
- Ensure messages are displayed in real-time on all clients without needing to refresh.

Question 5: Blogging Platform

Program Statement:

Develop a blogging platform where users can create, edit, and delete their blog posts. Implement user authentication to secure access.

Solution Hints:

- Use a server-side language like PHP or Node.js to handle operations.
- Store blog posts in a SQL or NoSQL database.
- Implement session-based authentication for users to manage their posts.
- Provide forms for creating and editing posts, and implement CRUD operations.

Question 6: E-commerce Website with Shopping Cart

Program Statement:

Build an e-commerce website that features products, a shopping cart, and a checkout process with form validation.

Solution Hints:

- Design a product listing page with HTML/CSS.
- Use JavaScript for adding items to a shopping cart stored in local storage or session storage.
- Implement form validation using HTML5 and JavaScript before submitting the checkout form.
- Ensure responsiveness across devices for a consistent shopping experience.

Question 7: Custom Video Player

Program Statement:

Create a custom HTML5 video player with custom controls such as play/pause, volume, and fullscreen toggles.

Solution Hints:

- Use the HTML5 **<video>** element to embed a video.
- Build custom control elements using HTML and control the video player using JavaScript.
- Style the player consistently and make sure controls are accessible on mobile devices.
- Add keyboard shortcuts for player controls for an enhanced user experience.

Question 8: Interactive Data Visualization

Program Statement:

Develop a web application that visualizes data interactively from a JSON endpoint using a JavaScript library like D3.js or Chart.js.

Solution Hints:

- Fetch data using AJAX from a JSON-formatted API or local JSON file.
- Use a data visualization library to parse data and display it in various chart formats.
- Implement interactive elements, such as tooltips and clickable legends, to make the data visualization more user-friendly.
- Style the charts to match the overall design theme of the application.

Question 9: To-Do List Application

Program Statement:

Create a to-do list application where users can add, delete, and mark tasks as complete. Include the ability to filter completed tasks.

Solution Hints:

- Use HTML and CSS to design the user interface.
- Implement the task management logic in JavaScript, allowing users to dynamically add, delete, and toggle the completion status of tasks.
- Use local storage to save the state of the to-do list so that it persists between sessions.
- Add a filtering mechanism to view all, completed, or active tasks.

Question 10: Pagination and Content Filtering

Program Statement:

Develop a web application that displays a list of items with pagination and options to filter items based on criteria like date and popularity.

Solution Hints:

- Display items using HTML/CSS and fetch data either from a server or local JSON file.
- Implement pagination controls using JavaScript to navigate through the data.
- Add filtering logic in JavaScript to sort or filter items based on the selected criteria.
- Ensure that the user interface is intuitive and responsive.

Question 11: Interactive Map with Location Markers

Program Statement:

Create an interactive map using a service like Google Maps API or Leaflet to display location markers that show additional information when clicked.

Solution Hints:

- Initialize the map and set its bounds to a relevant area.
- Use the mapping API to add markers at specified locations.
- Implement pop-ups or modals that display more information about each location when a marker is clicked.
- Style the map and markers to fit the theme of the application.

Question 12: Photo Gallery with Lightbox Feature

Program Statement:

Build a responsive photo gallery where images are displayed in a grid layout. Clicking on an image should open it in a lightbox view.

Solution Hints:

- Use CSS Grid or Flexbox to create a dynamic grid layout that adjusts to screen size.
- Implement a lightbox feature using JavaScript, which allows users to view a larger version of the image and navigate between images in the gallery.
- Ensure images are lazy-loaded as the user scrolls to improve performance and reduce initial load time.

Question 13: Dynamic Form Builder

Program Statement:

Develop a dynamic form builder that allows users to drag and drop form elements to create custom forms. Include the ability to save and load forms.

Solution Hints:

- Use HTML5 drag-and-drop API to implement the form builder interface.
- Allow users to select different types of form inputs (e.g., text, checkbox, dropdown) and configure their properties.
- Use local storage or a server-side database to save the form configurations.
- Provide options to load and edit existing forms.

Question 14: API Testing Tool

Program Statement:

Create a simple web-based API testing tool that allows users to make HTTP requests to a specified URL and display the response.

Solution Hints:

- Use HTML forms to capture the user input for the URL, HTTP method, and any headers or body data.
- Implement the request handling using JavaScript's Fetch API or XMLHttpRequest.
- Display the server response in a formatted manner, highlighting different parts of the response like headers and body.
- Provide options to save commonly used requests for quick reuse.

Question 15: Document Editor with Real-time Collaboration

Program Statement:

Build a simple document editor that supports real-time collaboration, allowing multiple users to edit a document simultaneously.

Solution Hints:

- Use WebSocket for real-time data transmission between clients and the server.
- Implement operational transformation or conflict-free replicated data types (CRDTs) to handle concurrent modifications.
- Provide basic text editing features and possibly some formatting tools.
- Ensure changes are reflected in real-time across all connected clients.

Question 16: Customizable Dashboard

Program Statement:

Develop a customizable dashboard where users can add, remove, and rearrange widgets such as clocks, weather, and news feeds.

Solution Hints:

- Implement a grid layout system that supports drag-and-drop rearrangement of widgets.
- Allow users to add or remove widgets from a predefined set.
- Fetch data for widgets from various APIs and update them periodically.
- Store user preferences in local storage or on a server to persist the state of the dashboard.

Question 17: File Uploader with Progress Indicator

Program Statement:

Create a file uploader tool that allows users to upload files to the server and shows a progress bar indicating the upload status.

Solution Hints:

- Use HTML5 File API to handle file inputs.
- Implement AJAX with progress events to upload files and update the user interface with the progress.
- Handle different file types and sizes, implementing appropriate restrictions and user feedback.
- Provide error handling for failed uploads and the option to retry.

Question 18: Interactive Floor Plan

Program Statement:

Develop an interactive floor plan for a building that users can navigate to view details about different rooms and areas.

Solution Hints:

- Use SVG or a canvas element to render the floor plan.
- Implement interactive elements such as clickable areas that display information or images about the room or area.
- Consider accessibility features, ensuring that all users can navigate the floor plan easily.
- Allow users to zoom in and out and navigate through different floors if applicable.

Question 19: Online Code Compiler

Program Statement:

Build an online code compiler that allows users to write code in various programming languages, compile it, and see the output or errors.

Solution Hints:

- Provide a text editor interface, possibly using libraries like Ace or CodeMirror.
- Implement server-side logic to safely compile and execute user-submitted code in a sandbox environment.
- Display the output or any compilation errors to the user.
- Support multiple programming languages by integrating various compilers or using an API like Judge0.

Question 20: Feedback Form with Sentiment Analysis

Program Statement:

Create a feedback form that analyzes the sentiment of the text submitted by the user and displays the sentiment analysis results.

Solution Hints:

- Use HTML and CSS to design the feedback form.
- Integrate a natural language processing API that provides sentiment analysis, such as Google Cloud Natural Language or similar.
- Display the sentiment analysis results in a user-friendly manner, such as a visual gauge or color-coded text.
- Store and possibly aggregate feedback for further analysis or administrative review.