**ASSIGNMENT 14**

**ROLLNO:2503A51L33**

**NAME: T. SHIVA**

**LAB: AI ASSISTED CODING**

**TASK1:**

Create a Responsive Web Page Layout.

**PROMPT:**

Create a responsive HTML and CSS layout for a basic website with a header, navigation bar, main content area, and footer. The layout should adapt to mobile, tablet, and desktop screen sizes using media queries. Keep the design clean, modern, and easy to customize.

**CODE:**

A screen shot of a computer screen

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

**OUTPUT:**

**A black and white website page

AI-generated content may be incorrect.**

**OBSERVATION:**

Structural Clarity: The layout uses semantic HTML elements (, , ) that clearly separate content zones, improving readability and accessibility.

Responsive Design: Media queries adapt the layout for tablets and mobile devices. Navigation links stack vertically on smaller screens, and font sizes adjust for readability.

Visual Simplicity: The design maintains a clean aesthetic with neutral background colors, consistent padding, and centered text, making it easy to customize without clutter.

**TASK 2:**

Interactive Button with JavaScript.

**PROMPT:**

Create a simple HTML page with a styled button. When the button is clicked, use JavaScript to display an alert message. Include clean, well-commented code for both HTML and JavaScript.

**CODE:**

A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

**OUTPUT:**

A blue rectangle with white text

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**OBSERVATION:**

**Event-Driven Interaction**: The button uses the onclick event to trigger a JavaScript function, demonstrating a core principle of dynamic web behavior.

**Clean Separation**: HTML handles structure, CSS handles styling, and JavaScript handles behavior—making the code modular and easy to maintain.

**User Feedback**: The alert provides immediate visual feedback, reinforcing the action-response loop and improving user engagement.

**Educational Clarity**: The JavaScript function is simple and well-commented, making it ideal for beginners learning DOM interaction and event handling.

**TASK 3:**

Form with Validation.

**PROMPT:**

Create a contact form with fields for Name, Email, and Message. Write JavaScript code to validate the form: ensure all fields are filled, and the email is in a valid format. Show inline error messages for invalid inputs. Use simple, readable code with comments.

**CODE:**

A screen shot of a computer program

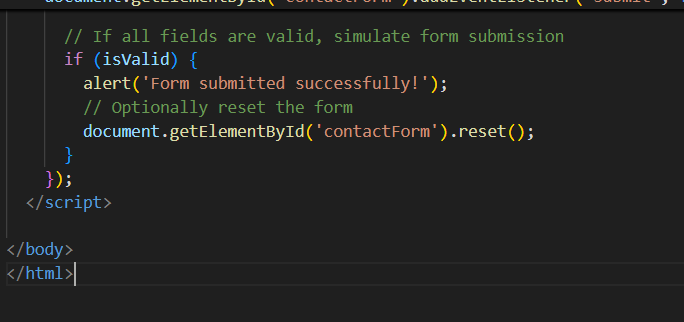
AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.



**OUTPUT:**

A white background with black lines

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**OBSERVATION:**

Field Completeness: The form ensures all fields—Name, Email, and Message—are filled before submission, preventing empty inputs.

Email Format Check: A regular expression validates the email format, catching common mistakes like missing “@” or domain.

Inline Error Feedback: Each field displays a specific error message directly below it, guiding users to correct issues immediately.

**TASK 4:**

Dynamic Content Generation.

**PROMPT:**

Create an HTML list of product names. Write JavaScript code to dynamically add a new item to the list when a button is clicked, and remove the last item when another button is clicked. Use simple, readable code with comments.

**CODE:**

A screen shot of a computer program

AI-generated content may be incorrect.

A computer screen shot of a program code

AI-generated content may be incorrect.

**OUTPUT:**

A screenshot of a product list

AI-generated content may be incorrect.

**OBSERVATION:**

**Add Functionality**: Users can append new items to the list by entering a name and clicking "Add Item", enabling flexible content expansion.

**Remove Functionality**: The "Remove Last Item" button deletes the most recent entry, simplifying list management without needing to select specific items.

**Input Validation**: The code checks for empty input before adding, preventing blank entries and improving data quality.

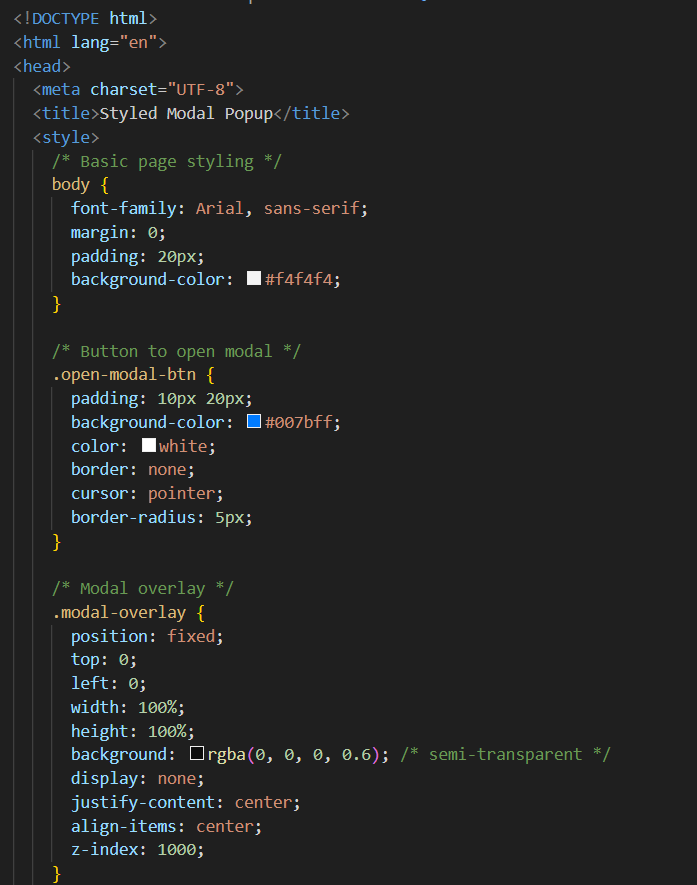
**TASK 5:**

Styled Modal Popup.

**PROMPT:**

Create a responsive modal popup that opens when a button is clicked. Style it with a semi-transparent dark overlay, center the modal box, and include a close button (×) in the top-right corner to hide the modal. Use HTML, CSS, and JavaScript with clear comments and simple structure.

**CODE:**

****

**A screen shot of a computer

AI-generated content may be incorrect.A screen shot of a computer program

AI-generated content may be incorrect.**

**OUTPUT:** **A screenshot of a computer

AI-generated content may be incorrect.**

**OBSERVATION:**

 The modal is triggered by a button click using onclick="openModal()", ensuring intuitive user interaction.

* The .modal-overlay uses position: fixed and rgba(0, 0, 0, 0.5) to create a full-screen, semi-transparent background that dims the rest of the page.
* The modal box (.modal-content) is centered with margin: 10% auto and styled with padding, rounded corners, and a subtle box shadow for visual depth.
* The close button (×) is placed in the top-right using position: absolute, and its onclick="closeModal()" handler hides the modal cleanly.
* JavaScript functions openModal() and closeModal() toggle the modal’s visibility by changing the display property between block and none.