**AI ASSISSTED CODING-14**

**TASK-1:**

**Create a Responsive Web Page Layout**

**Instructions:**

* Design a basic web page layout with a **header, main content area, and footer** using HTML and CSS.
* Use AI to assist in generating **responsive CSS** for different screen sizes.

Ensure the layout is clean and visually organized

**HTML:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Responsive Layout</title>

<link rel="stylesheet" href="style.css" />

</head>

<body>

<header>

<div class="container">

<h1>My Website</h1>

<nav>

<ul>

<li><a href="#">Home</a></li>

<li><a href="#">About</a></li>

<li><a href="#">Services</a></li>

<li><a href="#">Contact</a></li>

</ul>

</nav>

</div>

</header>

<main>

<div class="container">

<h2>Welcome to My Site</h2>

<p>This is a basic responsive web page layout with a header, main content, and footer. Resize the browser window to see how the layout adapts to different screen sizes.</p>

<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris lacinia velit at felis commodo, in commodo sapien pretium.</p>

</div>

</main>

<footer>

<div class="container">

<p>&copy; 2025 My Website. All rights reserved.</p>

</div>

</footer>

</body>

</html>

**CSS:**

/\* Base styles \*/

\* {

box-sizing: border-box;

margin: 0;

padding: 0;

}

body {

font-family: 'Segoe UI', sans-serif;

line-height: 1.6;

color: #333;

background-color: #f9f9f9;

}

.container {

width: 90%;

max-width: 1100px;

margin: 0 auto;

}

/\* Header styles \*/

header {

background-color: #0077cc;

color: #fff;

padding: 20px 0;

}

header h1 {

margin-bottom: 10px;

text-align: center;

}

nav ul {

display: flex;

flex-wrap: wrap;

justify-content: center;

list-style: none;

gap: 20px;

}

nav a {

color: #fff;

text-decoration: none;

font-weight: bold;

}

nav a:hover {

text-decoration: underline;

}

/\* Main content styles \*/

main {

background-color: #fff;

padding: 40px 0;

}

main h2 {

margin-bottom: 20px;

}

main p {

margin-bottom: 15px;

}

/\* Footer styles \*/

footer {

background-color: #333;

color: #fff;

text-align: center;

padding: 20px 0;

font-size: 14px;

}

/\* Responsive styles \*/

@media (max-width: 768px) {

nav ul {

flex-direction: column;

gap: 10px;

}

header h1 {

font-size: 24px;

}

main {

padding: 20px 0;

}

main h2 {

font-size: 22px;

}

}

@media (max-width: 480px) {

.container {

width: 95%;

}

header h1 {

font-size: 20px;

}

nav ul {

font-size: 14px;

}

}

**OUTPUT:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**TASK-2:**

**: Interactive Button with JavaScript**

**Instructions:**

* Create a **button** on a web page.
* Use AI to generate JavaScript code that displays an **alert message** when the button is clicked.
* Ensure the code is clean and well-commented.

**HTML:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0"/>

<title>Interactive Button</title>

<style>

body {

font-family: Arial, sans-serif;

text-align: center;

margin-top: 100px;

background-color: #f9f9f9;

}

button {

padding: 12px 24px;

font-size: 16px;

background-color: #0077cc;

color: white;

border: none;

border-radius: 5px;

cursor: pointer;

}

button:hover {

background-color: #005fa3;

}

</style>

</head>

<body>

<!-- Button element -->

<button id="myButton">Click Me!</button>

<!-- JavaScript file -->

<script src="script.js"></script>

</body>

</html>

**JAVASCRIPT:**

// Wait until the DOM is fully loaded

document.addEventListener("DOMContentLoaded", function () {

// Get the button element by its ID

const button = document.getElementById("myButton");

// Add a click event listener to the button

button.addEventListener("click", function () {

// Display an alert message when the button is clicked

alert("Hello! You clicked the button!");

});

});

**OUTPUT:**

A screenshot of a computer

AI-generated content may be incorrect.

**TASK-3:**

**Form with Validation**

**Instructions:**

* Design a **contact form** with fields: Name, Email, Message.
* Use AI to generate JavaScript code for **form validation** (e.g., non-empty fields, valid email format).

Add **inline error messages** if input is invalid

**HTML:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0"/>

<title>Contact Form with Validation</title>

<link rel="stylesheet" href="style.css" />

</head>

<body>

<div class="form-container">

<h2>Contact Us</h2>

<form id="contactForm" novalidate>

<div class="form-group">

<label for="name">Name:</label>

<input type="text" id="name" name="name" />

<small class="error" id="nameError"></small>

</div>

<div class="form-group">

<label for="email">Email:</label>

<input type="email" id="email" name="email" />

<small class="error" id="emailError"></small>

</div>

<div class="form-group">

<label for="message">Message:</label>

<textarea id="message" name="message" rows="5"></textarea>

<small class="error" id="messageError"></small>

</div>

<button type="submit">Send Message</button>

</form>

</div>

<script src="script.js"></script>

</body>

</html>

**CSS:**

body {

font-family: Arial, sans-serif;

background-color: #f4f4f4;

margin: 0;

padding: 0;

display: flex;

justify-content: center;

align-items: flex-start;

height: 100vh;

padding-top: 50px;

}

.form-container {

background-color: white;

padding: 30px;

width: 100%;

max-width: 500px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0,0,0,0.1);

}

h2 {

text-align: center;

margin-bottom: 20px;

}

.form-group {

margin-bottom: 20px;

}

label {

display: block;

font-weight: bold;

margin-bottom: 6px;

}

input, textarea {

width: 100%;

padding: 10px;

border: 1px solid #ccc;

border-radius: 4px;

font-size: 16px;

}

button {

background-color: #0077cc;

color: white;

border: none;

padding: 12px 20px;

border-radius: 5px;

font-size: 16px;

cursor: pointer;

width: 100%;

}

button:hover {

background-color: #005fa3;

}

.error {

color: red;

font-size: 14px;

display: block;

margin-top: 5px;

}

**JS:**

// Wait until the DOM is fully loaded

document.addEventListener("DOMContentLoaded", function () {

const form = document.getElementById("contactForm");

form.addEventListener("submit", function (e) {

e.preventDefault(); // Prevent actual form submission

// Clear previous errors

clearErrors();

// Get input values

const name = document.getElementById("name").value.trim();

const email = document.getElementById("email").value.trim();

const message = document.getElementById("message").value.trim();

let isValid = true;

// Validate Name

if (name === "") {

showError("nameError", "Name is required.");

isValid = false;

}

// Validate Email

if (email === "") {

showError("emailError", "Email is required.");

isValid = false;

} else if (!validateEmail(email)) {

showError("emailError", "Please enter a valid email address.");

isValid = false;

}

// Validate Message

if (message === "") {

showError("messageError", "Message cannot be empty.");

isValid = false;

}

// If all fields are valid, you could submit or show a success message

if (isValid) {

alert("Form submitted successfully!");

form.reset();

}

});

// Show inline error message

function showError(elementId, message) {

const errorElement = document.getElementById(elementId);

errorElement.textContent = message;

}

// Clear all error messages

function clearErrors() {

const errorElements = document.querySelectorAll(".error");

errorElements.forEach(el => el.textContent = "");

}

// Basic email format check using regex

function validateEmail(email) {

const emailPattern = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

return emailPattern.test(email);

}

});

**OUTPUT:**

A screenshot of a contact us

AI-generated content may be incorrect.

**TASK-4:**

**Dynamic Content Generation**

**Instructions:**

* Create a list of items (e.g., product names) using **HTML**.

Use AI-generated JavaScript to **dynamically add or remove items** from the list when a button is clicked

**HTML:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0"/>

<title>Dynamic List</title>

<link rel="stylesheet" href="style.css" />

</head>

<body>

<div class="container">

<h2>Product List</h2>

<!-- List container -->

<ul id="productList">

<li>Apple</li>

<li>Banana</li>

<li>Orange</li>

</ul>

<!-- Input + buttons -->

<div class="form-group">

<input type="text" id="productInput" placeholder="Enter product name..." />

<button onclick="addItem()">Add Item</button>

<button onclick="removeItem()">Remove Last Item</button>

</div>

</div>

<script src="script.js"></script>

</body>

</html>

**CSS:**

body {

font-family: Arial, sans-serif;

background-color: #f4f4f4;

margin: 0;

padding: 40px;

display: flex;

justify-content: center;

}

.container {

background-color: #fff;

padding: 30px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0,0,0,0.1);

max-width: 500px;

width: 100%;

}

h2 {

text-align: center;

margin-bottom: 20px;

}

ul {

list-style-type: disc;

padding-left: 20px;

margin-bottom: 20px;

}

li {

margin-bottom: 8px;

font-size: 16px;

}

.form-group {

display: flex;

gap: 10px;

flex-wrap: wrap;

}

input[type="text"] {

flex: 1;

padding: 10px;

font-size: 16px;

}

button {

padding: 10px 16px;

font-size: 16px;

background-color: #0077cc;

color: #fff;

border: none;

border-radius: 5px;

cursor: pointer;

}

button:hover {

background-color: #005fa3;

}

**JS:**

// Function to add a new item to the list

function addItem() {

const input = document.getElementById("productInput");

const value = input.value.trim();

if (value === "") {

alert("Please enter a product name.");

return;

}

const li = document.createElement("li");

li.textContent = value;

document.getElementById("productList").appendChild(li);

input.value = ""; // Clear input field

}

// Function to remove the last item in the list

function removeItem() {

const list = document.getElementById("productList");

const lastItem = list.lastElementChild;

if (lastItem) {

list.removeChild(lastItem);

} else {

alert("No more items to remove.");

}

}

**OUTPUT:**

A screenshot of a product list

AI-generated content may be incorrect.

**TASK-5:**

**B: Styled Modal Popup**

**Instructions:**

* **Use AI to generate a modal popup that opens when a button is clicked.**
* **Style the modal using CSS with a semi-transparent overlay.**
* **Include a close button that hides the modal.**

**HTML:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

<title>Styled Modal Popup</title>

<link rel="stylesheet" href="style.css" />

</head>

<body>

<button id="openModalBtn">Open Modal</button>

<!-- Modal structure -->

<div id="myModal" class="modal">

<div class="modal-content">

<span class="close-btn" id="closeModalBtn">&times;</span>

<h2>Modal Title</h2>

<p>This is a styled modal popup with a semi-transparent overlay.</p>

</div>

</div>

<script src="script.js"></script>

</body>

</html>

**CSS:**

/\* Basic styling for the button \*/

body {

font-family: Arial, sans-serif;

background-color: #f4f4f4;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

}

button#openModalBtn {

padding: 12px 24px;

font-size: 18px;

background-color: #0077cc;

color: white;

border: none;

border-radius: 6px;

cursor: pointer;

transition: background-color 0.3s ease;

}

button#openModalBtn:hover {

background-color: #005fa3;

}

/\* Modal container (hidden by default) \*/

.modal {

display: none; /\* Hidden \*/

position: fixed; /\* Stay in place \*/

z-index: 1000; /\* Sit on top \*/

left: 0;

top: 0;

width: 100%; /\* Full width \*/

height: 100%; /\* Full height \*/

overflow: auto; /\* Enable scroll if needed \*/

background-color: rgba(0, 0, 0, 0.5); /\* Black with opacity \*/

}

/\* Modal content box \*/

.modal-content {

background-color: #fff;

margin: 10% auto; /\* 10% from top and centered \*/

padding: 20px 30px;

border-radius: 8px;

width: 90%;

max-width: 500px;

box-shadow: 0 5px 15px rgba(0,0,0,0.3);

position: relative;

animation: fadeInScale 0.3s ease forwards;

}

/\* Close button styling \*/

.close-btn {

color: #aaa;

float: right;

font-size: 28px;

font-weight: bold;

cursor: pointer;

position: absolute;

right: 15px;

top: 10px;

transition: color 0.3s ease;

}

.close-btn:hover,

.close-btn:focus {

color: #000;

}

/\* Animation for modal appearance \*/

@keyframes fadeInScale {

0% {

opacity: 0;

transform: scale(0.9);

}

100% {

opacity: 1;

transform: scale(1);

}

}

**JS:**

// Get modal elements

const modal = document.getElementById("myModal");

const openBtn = document.getElementById("openModalBtn");

const closeBtn = document.getElementById("closeModalBtn");

// Open modal when button clicked

openBtn.addEventListener("click", () => {

modal.style.display = "block";

});

// Close modal when close button clicked

closeBtn.addEventListener("click", () => {

modal.style.display = "none";

});

// Also close modal if user clicks outside modal content

window.addEventListener("click", (event) => {

if (event.target === modal) {

modal.style.display = "none";

}

});

**OUTPUT:**

