

# ASSIGNMENT - 14.4

## 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.

Expected Output:

- A responsive web page with:

Header with navigation links

Main section with placeholder text/images

Footer with copyright or contact info

Layout adapts correctly to desktop, tablet, and mobile screen sizes

HTML CODE :

```
1  <!DOCTYPE html>
Just HTML that goes in the <body> goes here. Learn more
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6  <title>Responsive Web Page</title>
7  <link rel="stylesheet" href="style.css">
8 </head>
9 <body>
10
11  <!-- Header Section -->
12  <header>
13      <div class="logo">MyWebsite</div>
14      <nav class="navbar">
15          <ul id="nav-links">
16              <li><a href="#">Home</a></li>
17              <li><a href="#">About</a></li>
18              <li><a href="#">Services</a></li>
19              <li><a href="#">Contact</a></li>
20          </ul>
21          <div class="menu-toggle" id="menu-toggle">≡</div>
22      </nav>
23  </header>
24
25  <!-- Main Section -->
26  <main>
27      <section class="hero">
28          <h1>Welcome to My Responsive Page</h1>
```

```
<h1>Welcome to My Responsive Page</h1>
<p>This layout adjusts beautifully across desktop, tablet, and mobile screens.</p>
<button>Learn More</button>
</section>

<section class="content">
  <div class="card">
    
    <h3>Card Title 1</h3>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque commodo.</p>
  </div>

  <div class="card">
    
    <h3>Card Title 2</h3>
    <p>Nulla facilisi. Aliquam erat volutpat. Curabitur vel ligula ut magna gravida.</p>
  </div>

  <div class="card">
    
    <h3>Card Title 3</h3>
    <p>Phasellus ac tellus ut elit hendrerit aliquet nec nec lorem.</p>
  </div>
</section>
</main>
```

```
4  <!-- Footer Section -->
5  <footer>
6    <p>© 2025 MyWebsite | Designed by Vikas Kathroju</p>
7  </footer>
8
9  <script src="script.js"></script>
0  </body>
1  </html>
2
```

## CSS CODE :

```
1 /* Reset */
2 * {
3   margin: 0;
4   padding: 0;
5   box-sizing: border-box;
6   font-family: 'Poppins', sans-serif;
7 }
8
9 body {
0   background-color: #f4f6f7;
1   color: #333;
2 }
3
4 /* Header */
5 header {
6   display: flex;
7   justify-content: space-between;
8   align-items: center;
9   background: #2980b9;
0   color: white;
1   padding: 15px 30px;
2 }
3
4 .logo {
5   font-size: 1.5rem;
6   font-weight: bold;
7 }
```

```
35 .navbar ul li a {
36   text-decoration: none;
37   color: white;
38   font-weight: 500;
39   transition: 0.3s;
40 }
41
42 .navbar ul li a:hover {
43   color: #ffeb3b;
44 }
45
46 /* Menu toggle (for mobile) */
47 .menu-toggle {
48   display: none;
49   font-size: 24px;
50   cursor: pointer;
51 }
52
53 /* Hero Section */
54 .hero {
55   text-align: center;
56   padding: 60px 20px;
57   background: linear-gradient(to right, #6dd5fa, #2980b9);
58   color: white;
59 }
```

```
1 .hero h1 {  
2   font-size: 2.5rem;  
3   margin-bottom: 15px;  
4 }  
5  
6 .hero p {  
7   font-size: 1.1rem;  
8   margin-bottom: 25px;  
9 }  
10  
11 .hero button {  
12   background: white;  
13   color: #2980b9;  
14   border: none;  
15   padding: 10px 20px;  
16   font-size: 1rem;  
17   border-radius: 5px;  
18   cursor: pointer;  
19   transition: 0.3s;  
20 }  
21  
22 .hero button:hover {  
23   background: #ffeb3b;  
24 }  
25
```

```
36 /* Content Section */  
37 .content {  
38   display: flex;  
39   justify-content: space-around;  
40   flex-wrap: wrap;  
41   padding: 40px 20px;  
42   gap: 20px;  
43 }  
44  
45 .card {  
46   background: white;  
47   width: 30%;  
48   min-width: 250px;  
49   border-radius: 10px;  
50   box-shadow: 0 4px 10px rgba(0,0,0,0.1);  
51   padding: 20px;  
52   text-align: center;  
53   transition: transform 0.3s;  
54 }  
55  
56 .card:hover {  
57   transform: translateY(-5px);  
58 }  
59  
60 .card img {  
61   width: 100%;  
62   border-radius: 10px;  
63 }
```

```
115/* Footer */
116footer {
117    background: #2980b9;
118    color: white;
119    text-align: center;
120    padding: 15px 0;
121    font-size: 0.9rem;
122}
123
124/* Responsive Design */
125
126/* Tablets */
127@media (max-width: 768px) {
128    .navbar ul {
129        display: none;
130        flex-direction: column;
131        background: #2980b9;
132        width: 100%;
133        text-align: center;
134        position: absolute;
135        top: 60px;
136        left: 0;
137        padding: 10px 0;
138    }
139
140.navbar ul.show {
141    display: flex;
142}
```

```
.menu-toggle {
    display: block;
}

.content {
    flex-direction: column;
    align-items: center;
}

/* Mobile */
@media (max-width: 480px) {
    .hero h1 {
        font-size: 1.8rem;
    }

    .hero p {
        font-size: 1rem;
    }

    .card {
        width: 90%;
    }
}
```

## JAVASCRIPT CODE :

```
1 // Toggle menu for mobile view
2 const menuToggle = document.getElementById('menu-toggle');
3 const navLinks = document.getElementById('nav-links');
4
5 menuToggle.addEventListener('click', () => {
6   navLinks.classList.toggle('show');
7 });
8
```

## WEBPAGE LAYOUT :

The wireframe illustrates a responsive website layout. At the top, a header bar contains the site name "MyWebsite" on the left and links for "Home", "About", "Services", and "Contact" on the right. Below the header is a large blue rectangular area containing the text "Welcome to My Responsive Page" and a "Learn More" button. This is followed by a row of three columns, each labeled "Placeholder Image". In the bottom section, there is a grid of three cards. Each card has a placeholder image at the top and a title below it. The first card's title is "Card Title 1" with the subtitle "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque commodo.". The second card's title is "Card Title 2" with the subtitle "Nulla facilisi. Aliquam erat volutpat. Curabitur vel ligula ut magna gravida.". The third card's title is "Card Title 3" with the subtitle "Phasellus ac tellus ut elit hendrerit aliquet nec nec lorem.". The entire layout is designed to be responsive, with elements scaling down for smaller screens.

## 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.

### Expected Output:

- A web page with a button.
- Clicking the button shows a pop-up alert message, e.g., “Button clicked!”

### HTML CODE :

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport"
6          content="width=device-width, initial-
7              scale=1.0">
8  <title>Button Alert Example</title>
9      <link rel="stylesheet"
10         href="style.css">
11  </head>
12  <body>
13
14      <div class="container">
15          <h1>Click the Button!</h1>
16          <!-- Button that will trigger
17              JavaScript alert -->
18          <button id="alertButton">Click
19              Me</button>
20      </div>
21
22      <script src="script.js"></script>
23  </body>
24  </html>
```

## CSS CODE :

```
▼ /* Basic styling for page layout */
▼ body {
    font-family: 'Poppins', sans-serif;
    background-color: #f0f8ff;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
}

▼ /* Center container */
▼ .container {
    text-align: center;
}

▼ /* Button styling */
▼ button {
    background-color: #2980b9;
    color: white;
    border: none;
    padding: 12px 25px;
    border-radius: 8px;
    font-size: 1rem;
    cursor: pointer;
    transition: 0.3s;
}

▼ /* Button hover effect */
▼ button:hover {
    background-color: #1e6fa1;
}
```

## JAVASCRIPT CODE :

```
1 // Wait until the page loads completely
2 ▾ document.addEventListener("DOMContentLoaded",
3   () => {
4
5     // Get the button element by its ID
6     const alertButton =
7       document.getElementById("alertButton");
8
9     // Add a click event listener to the button
10    alertButton.addEventListener("click", () =>
11      {
12        // When the button is clicked, show an
13        // alert message
14        alert("Button clicked!");
15      });
16
17    });
18
19  );
20
```

## WEBPAGE LAYOUT:



# 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.

## Expected Output:

- A functional contact form where:  
Submitting with empty fields shows error messages.  
Submitting with valid input displays a “Form submitted successfully” message.

## HTML CODE :

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6  <title>Contact Form with Validation</title>
7  <link rel="stylesheet" href="style.css">
8 </head>
9 <body>
10
11 <div class="container">
12     <h1>Contact Us</h1>
13
14 <form id="contactForm" novalidate>
15     <!-- Name Field -->
16     <div class="form-group">
17         <label for="name">Name:</label>
18         <input type="text" id="name" placeholder="Enter your name">
19         <small class="error-message"></small>
20     </div>
21
22     <!-- Email Field -->
23     <div class="form-group">
24         <label for="email">Email:</label>
25         <input type="email" id="email" placeholder="Enter your email">
26         <small class="error-message"></small>
27     </div>
```

```

29  <!-- Message Field -->
30  <div class="form-group">
31    <label for="message">Message:</label>
32    <textarea id="message" placeholder="Enter your message"></textarea>
33    <small class="error-message"></small>
34  </div>
35
36  <button type="submit">Submit</button>
37
38  <!-- Success Message -->
39  <p id="successMessage" class="success-message"></p>
40  </form>
41 </div>
42
43  <script src="script.js"></script>
44 </body>
45 </html>

```

## CSS CODE :

```

1  /* Basic layout and form styling */
2  body {
3    font-family: 'Poppins', sans-serif;
4    background: #f0f8ff;
5    display: flex;
6    justify-content: center;
7    align-items: center;
8    height: 100vh;
9  }
10
11 .container {
12   background: white;
13   padding: 25px 30px;
14   border-radius: 12px;
15   box-shadow: 0 4px 12px rgba(0,0,0,0.1);
16   width: 100%;
17   max-width: 400px;
18 }
19
20 h1 {
21   text-align: center;
22   margin-bottom: 20px;
23   color: #2980b9;
24 }
25
26 /* Form Group */
27 .form-group {
28   margin-bottom: 15px;
29 }
30

```

```
31 ▼ label {
32   display: block;
33   margin-bottom: 6px;
34   font-weight: 500;
35 }
36
37 ▼ input, textarea {
38   width: 100%;
39   padding: 10px;
40   border: 1px solid #ccc;
41   border-radius: 6px;
42   font-size: 1rem;
43   outline: none;
44 }
45
46 /* Error and success messages */
47 ▼ .error-message {
48   color: red;
49   font-size: 0.85rem;
50   margin-top: 5px;
51   display: block;
52 }
53
54 ▼ .success-message {
55   color: green;
56   text-align: center;
57   font-weight: 600;
58   margin-top: 15px;
59 }
60
```

```
▼ /* Button styling */
▼ button {
  background-color: #2980b9;
  color: white;
  border: none;
  padding: 10px;
  border-radius: 6px;
  font-size: 1rem;
  width: 100%;
  cursor: pointer;
  transition: 0.3s;
}
▼ button:hover {
  background-color: #1e6fa1;
}
```

## JAVASCRIPT CODE :

```
1 // Wait until the DOM is fully loaded
2 document.addEventListener("DOMContentLoaded", () => {
3
4   const form = document.getElementById("contactForm");
5   const successMessage = document.getElementById("successMessage");
6
7   // Form submit event
8   form.addEventListener("submit", function(event) {
9     event.preventDefault(); // Prevent form refresh
10
11   // Get input elements
12   const nameInput = document.getElementById("name");
13   const emailInput = document.getElementById("email");
14   const messageInput = document.getElementById("message");
15
16   // Reset previous messages
17   clearErrors();
18   successMessage.textContent = "";
19
20   // Validation flags
21   let isValid = true;
22
23   // Validate name
24   if (nameInput.value.trim() === "") {
25     showError(nameInput, "Name is required.");
26     isValid = false;
27   }
28
29
30   // Validate email
31   if (emailInput.value.trim() === "") {
32     showError(emailInput, "Email is required.");
33     isValid = false;
34   } else if (!isValidEmail(emailInput.value.trim())) {
35     showError(emailInput, "Please enter a valid email address.");
36     isValid = false;
37   }
38
39   // Validate message
40   if (messageInput.value.trim() === "") {
41     showError(messageInput, "Message cannot be empty.");
42     isValid = false;
43   }
44
45   // If valid, show success message
46   if (isValid) {
47     successMessage.textContent = "Form submitted successfully!";
48     form.reset();
49   }
50 });
51
52 // Helper function: show error below input
53 function showError(inputElement, message) {
54   const errorMsg = inputElement.parentElement.querySelector(".error-message");
55   errorMsg.textContent = message;
56 }
```

```
1 // Validate email
2 if (emailInput.value.trim() === "") {
3   showError(emailInput, "Email is required.");
4   isValid = false;
5 } else if (!isValidEmail(emailInput.value.trim())) {
6   showError(emailInput, "Please enter a valid email address.");
7   isValid = false;
8 }
9
10 // Validate message
11 if (messageInput.value.trim() === "") {
12   showError(messageInput, "Message cannot be empty.");
13   isValid = false;
14 }
15
16 // If valid, show success message
17 if (isValid) {
18   successMessage.textContent = "Form submitted successfully!";
19   form.reset();
20 }
21
22 // Helper function: show error below input
23 function showError(inputElement, message) {
24   const errorMsg = inputElement.parentElement.querySelector(".error-message");
25   errorMsg.textContent = message;
26 }
```

```
// Helper function: clear all errors
function clearErrors() {
  document.querySelectorAll(".error-message").forEach(msg => msg.textContent = "");
}

// Helper function: validate email format
function isValidEmail(email) {
  const emailRegex = /^[^@\s]+@[^\s]+\.\[^@\s]+$/;
  return emailRegex.test(email);
}
});
```

## WEBPAGE LAYOUT :

The image shows a contact form titled "Contact Us" centered on a light gray background. The form has a white rounded rectangular background. It contains three input fields: "Name:" with a placeholder "Enter your name", "Email:" with a placeholder "Enter your email", and "Message:" with a placeholder "Enter your message". Below these fields is a blue "Submit" button.

Contact Us

Name:

Enter your name

Email:

Enter your email

Message:

Enter your message

Submit

## TASK4: 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.

Expected Output:

A web page with a list and buttons:

“Add Item” adds a new item to the list dynamically.

“Remove Item” deletes the last item.

Updates occur without reloading the page.

**HTML CODE :**

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width,
6      initial-scale=1.0">
6  <title>Dynamic List Example</title>
7  <link rel="stylesheet" href="style.css">
8 </head>
9 <body>
0
1  <div class="container">
2  <h1>My Product List</h1>
3
4  <!-- List of items -->
5  <ul id="itemList">
6      <li>Product 1</li>
7      <li>Product 2</li>
8      <li>Product 3</li>
9  </ul>
0
1  <!-- Buttons -->
2  <div class="buttons">
3      <button id="addBtn">Add Item</button>
4      <button id="removeBtn">Remove Item</button>
5  </div>
6 </div>
7
8  <script src="script.js"></script>
9 </body>
0 </html>
```

## CSS CODE :

```
1 ▾ body {
2   font-family: 'Poppins', sans-serif;
3   background-color: #f0f8ff;
4   display: flex;
5   justify-content: center;
6   align-items: center;
7   height: 100vh;
8 }
9
10 ▾ .container {
11   background: white;
12   padding: 25px 30px;
13   border-radius: 12px;
14   box-shadow: 0 4px 12px rgba(0,0,0,0.1);
15   text-align: center;
16   width: 90%;
17   max-width: 400px;
18 }
19
20 ▾ h1 {
21   color: #2980b9;
22   margin-bottom: 20px;
23 }
24
25 ▾ ul {
26   list-style-type: none;
27   padding: 0;
28   margin-bottom: 20px;
29 }
30
31 ▾ li {
32   background-color: #e3f2fd;
33   margin: 6px 0;
34   padding: 10px;
35   border-radius: 6px;
36 }
37
38 ▾ button {
39   background-color: #2980b9;
40   color: white;
41   border: none;
42   padding: 10px 15px;
43   border-radius: 8px;
44   cursor: pointer;
45   margin: 5px;
46   transition: 0.3s;
47 }
48
49 ▾ button:hover {
50   background-color: #1e6fa1;
51 }
52
```

## JAVASCRIPT CODE :

```
1 // AI-generated JavaScript for dynamic list updates
2
3 // Get references to HTML elements
4 const addBtn = document.getElementById('addBtn');
5 const removeBtn =
6 document.getElementById('removeBtn');
7 const itemList = document.getElementById('itemList');
8
9 // Counter to track number of products
10 let itemCount = itemList.children.length;
11
12 // Add a new item when "Add Item" is clicked
13 addBtn.addEventListener('click', () => {
14   itemCount++;
15   const newItem = document.createElement('li');
16   newItem.textContent = `Product ${itemCount}`;
17   itemList.appendChild(newItem);
18 });
19
20 // Remove the last item when "Remove Item" is clicked
21 removeBtn.addEventListener('click', () => {
22   if (itemList.lastElementChild) {
23     itemList.removeChild(itemList.lastElementChild);
24     itemCount--;
25   } else {
26     alert("No more items to remove!");
27   }
28});
```

## WEBPAGE LAYOUT :



# Task 5: 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.

## Expected Output:

- A web page with a button labeled “Open Modal”.
- Clicking the button displays the modal popup with content.
- Modal can be closed by clicking the “X” button or overlay area.

## HTML CODE:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Styled Modal Popup</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>

  <div class="container">
    <h1>AI-Generated Modal Example</h1>
    <!-- Button to open modal -->
    <button id="openModalBtn">Open Modal</button>
  </div>

  <!-- Modal Structure -->
  <div id="modal" class="modal">
    <div class="modal-content">
      <span class="close-btn">&times;</span>
      <h2>Welcome to My Modal!</h2>
      <p>This modal popup was created using HTML, CSS, and JavaScript.</p>
    </div>
  </div>

  <script src="script.js"></script>
</body>
</html>
```

## CSS CODE :

```
1 /* Basic Page Styling */
2 body {
3   font-family: 'Poppins', sans-serif;
4   background-color: #f0f8ff;
5   display: flex;
6   justify-content: center;
7   align-items: center;
8   height: 100vh;
9   margin: 0;
10 }
11
12 .container {
13   text-align: center;
14 }
15
16 button {
17   background-color: #2980b9;
18   color: white;
19   border: none;
20   padding: 12px 20px;
21   border-radius: 8px;
22   cursor: pointer;
23   font-size: 1rem;
24   transition: 0.3s;
25 }
26
27 button:hover {
28   background-color: #1e6fa1;
29 }
```

```
/* Modal Overlay (hidden by default) */
.modal {
  display: none; /* Hidden initially */
  position: fixed;
  z-index: 1000;
  left: 0;
  top: 0;
  width: 100%;
  height: 100%;
  background-color: rgba(0,0,0,0.5); /* Semi-transparent overlay */
  justify-content: center;
  align-items: center;
}

/* Modal Content Box */
.modal-content {
  background-color: white;
  padding: 25px 30px;
  border-radius: 12px;
  text-align: center;
  width: 90%;
  max-width: 400px;
  position: relative;
  animation: slideDown 0.3s ease;
}
```

```

7 /* Close Button */
8 .close-btn {
9   position: absolute;
0   top: 10px;
1   right: 15px;
2   font-size: 1.5rem;
3   color: #555;
4   cursor: pointer;
5   transition: 0.2s;
6 }
7
8 .close-btn:hover {
9   color: #e74c3c;
0 }
1
2 /* Simple animation for modal */
3 @keyframes slideDown {
4 from {
5   transform: translateY(-30px);
6   opacity: 0;
7 }
8 to {
9   transform: translateY(0);
0   opacity: 1;
1 }
2

```

## JAVASCRIPT CODE :

```

1 // AI-generated JavaScript for Modal Popup
2
3 // Get DOM elements
4 const openModalBtn = document.getElementById('openModalBtn');
5 const modal = document.getElementById('modal');
6 const closeBtn = document.querySelector('.close-btn');
7
8 // Open modal when button clicked
9 openModalBtn.addEventListener('click', () => {
0   modal.style.display = 'flex'; // Show modal
1 });
2
3 // Close modal when "X" is clicked
4 closeBtn.addEventListener('click', () => {
5   modal.style.display = 'none'; // Hide modal
6 });
7
8 // Close modal when user clicks outside the modal box
9 window.addEventListener('click', (event) => {
0   if (event.target === modal) {
1     modal.style.display = 'none';
2   }
3 });
4

```

## WEBPAGE LAYOUT :

