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 :

```html
<!DOCTYPE html>
```

Just HTML that goes in the <body> goes here. Learn more

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

  <!-- Header Section -->
  <header>
    <div class="logo">MyWebsite</div>
    <nav class="navbar">
      <ul id="nav-links">
        <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>
      <div class="menu-toggle" id="menu-toggle">≡</div>
    </nav>
  </header>

  <!-- Main Section -->
  <main>
    <section class="hero">
      <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">
      <img src="https://via.placeholder.com/250" alt="Placeholder Image">
      <h3>Card Title 1</h3>
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque commodo.</p>
    </div>

    <div class="card">
      <img src="https://via.placeholder.com/250" alt="Placeholder Image">
      <h3>Card Title 2</h3>
      <p>Nulla facilisi. Aliquam erat volutpat. Curabitur vel ligula ut magna gravida.</p>
    </div>

    <div class="card">
      <img src="https://via.placeholder.com/250" alt="Placeholder Image">
      <h3>Card Title 3</h3>
      <p>Phasellus ac tellus ut elit hendrerit aliquet nec nec lorem.</p>
    </div>
  </section>
</main>
```

```
  <!-- Footer Section -->
  <footer>
    <p>© 2025 MyWebsite | Designed by Vikas Kathroju</p>
  </footer>

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

CSS CODE :

```css
/* Reset */
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: 'Poppins', sans-serif;
}

body {
  background-color: #f4f6f7;
  color: #333;
}

/* Header */
header {
  display: flex;
  justify-content: space-between;
  align-items: center;
  background: #2980b9;
  color: white;
  padding: 15px 30px;
}

.logo {
  font-size: 1.5rem;
  font-weight: bold;
}
```

```css
.navbar ul li a {
  text-decoration: none;
  color: white;
  font-weight: 500;
  transition: 0.3s;
}

.navbar ul li a:hover {
  color: #ffeb3b;
}

/* Menu toggle (for mobile) */
.menu-toggle {
  display: none;
  font-size: 24px;
  cursor: pointer;
}

/* Hero Section */
.hero {
  text-align: center;
  padding: 60px 20px;
  background: linear-gradient(to right, #6dd5fa, #2980b9);
  color: white;
}
```

```css
.hero h1 {
  font-size: 2.5rem;
  margin-bottom: 15px;
}

.hero p {
  font-size: 1.1rem;
  margin-bottom: 25px;
}

.hero button {
  background: white;
  color: #2980b9;
  border: none;
  padding: 10px 20px;
  font-size: 1rem;
  border-radius: 5px;
  cursor: pointer;
  transition: 0.3s;
}

.hero button:hover {
  background: #ffeb3b;
}
```

```css
/* Content Section */
.content {
  display: flex;
  justify-content: space-around;
  flex-wrap: wrap;
  padding: 40px 20px;
  gap: 20px;
}

.card {
  background: white;
  width: 30%;
  min-width: 250px;
  border-radius: 10px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  padding: 20px;
  text-align: center;
  transition: transform 0.3s;
}

.card:hover {
  transform: translateY(-5px);
}

.card img {
  width: 100%;
  border-radius: 10px;
}
```

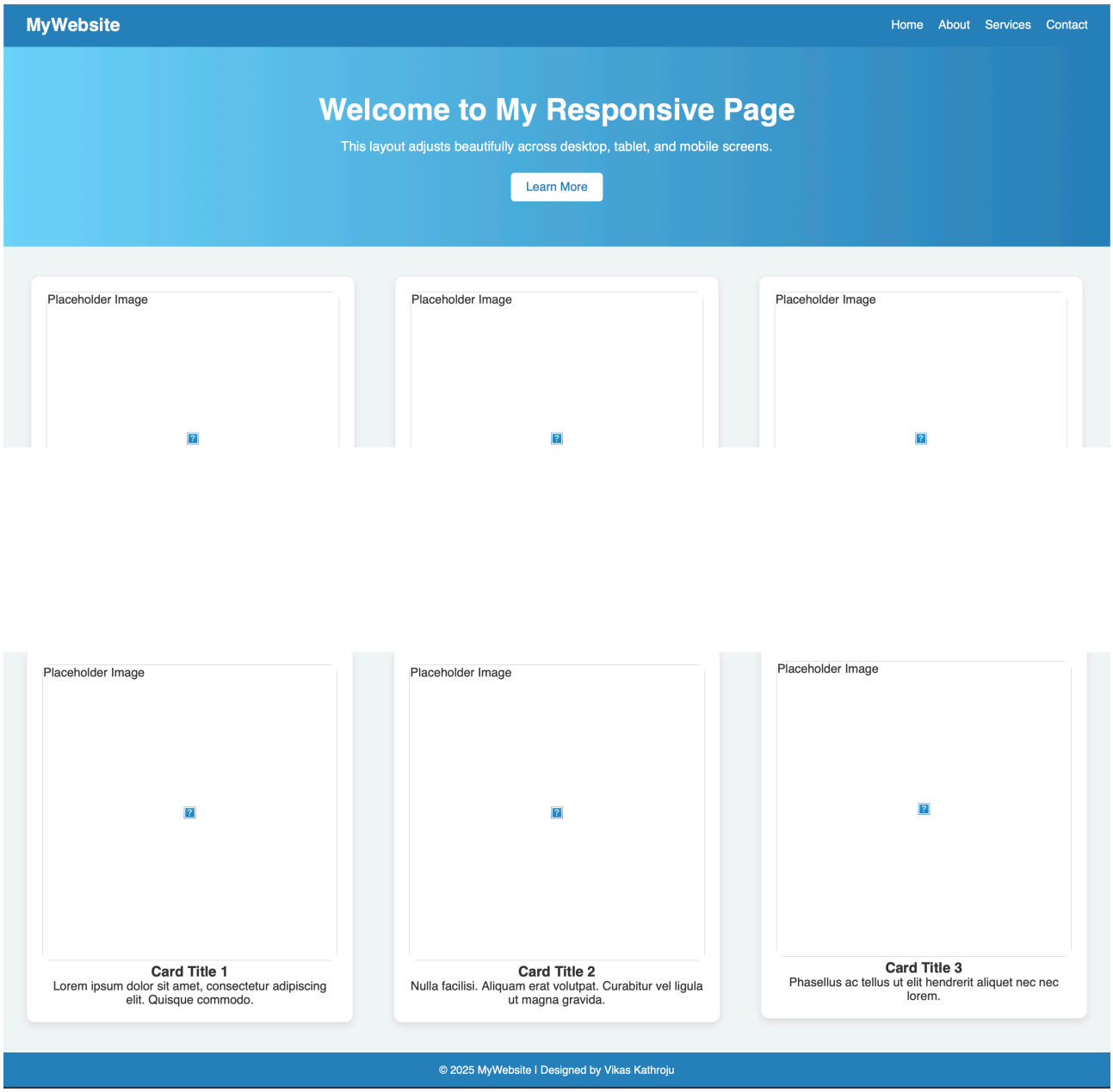
```css
/* Footer */
footer {
  background: #2980b9;
  color: white;
  text-align: center;
  padding: 15px 0;
  font-size: 0.9rem;
}

/* Responsive Design */

/* Tablets */
@media (max-width: 768px) {
  .navbar ul {
    display: none;
    flex-direction: column;
    background: #2980b9;
    width: 100%;
    text-align: center;
    position: absolute;
    top: 60px;
    left: 0;
    padding: 10px 0;
  }

  .navbar ul.show {
    display: flex;
  }
```

```css
  .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 :

```javascript
// Toggle menu for mobile view
const menuToggle = document.getElementById('menu-toggle');
const navLinks = document.getElementById('nav-links');

menuToggle.addEventListener('click', () => {
  navLinks.classList.toggle('show');
});

```

WEBPAGE LAYOUT :



TASK 2: INTERACTIVE BUTTON WITH JAVASCRIPT

Instructions:

•Create a button on a web page.
•Use AI to generate JavaScript code that displays analert 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 :

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

  <div class="container">
    <h1>Click the Button!</h1>
    <!-- Button that will trigger
JavaScript alert -->
    <button id="alertButton">Click
Me</button>
  </div>

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

CSS CODE :

```css
/* 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 :

```
// Wait until the page loads completely
document.addEventListener("DOMContentLoaded",
() => {

  // Get the button element by its ID
  const alertButton =
document.getElementById("alertButton");

  // Add a click event listener to the button
  alertButton.addEventListener("click", () =>
{
    // When the button is clicked, show an
alert message
    alert("Button clicked!");
  });

});

```

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 :

```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="container">
    <h1>Contact Us</h1>

    <form id="contactForm" novalidate>
      <!-- Name Field -->
      <div class="form-group">
        <label for="name">Name:</label>
        <input type="text" id="name" placeholder="Enter your name">
        <small class="error-message"></small>
      </div>

      <!-- Email Field -->
      <div class="form-group">
        <label for="email">Email:</label>
        <input type="email" id="email" placeholder="Enter your email">
        <small class="error-message"></small>
      </div>
```

```
      <!-- Message Field -->
      <div class="form-group">
        <label for="message">Message:</label>
        <textarea id="message" placeholder="Enter your message"></textarea>
        <small class="error-message"></small>
      </div>

      <button type="submit">Submit</button>

      <!-- Success Message -->
      <p id="successMessage" class="success-message"></p>
    </form>
  </div>

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

CSS CODE :

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

.container {
  background: white;
  padding: 25px 30px;
  border-radius: 12px;
  box-shadow: 0 4px 12px rgba(0,0,0,0.1);
  width: 100%;
  max-width: 400px;
}

h1 {
  text-align: center;
  margin-bottom: 20px;
  color: #2980b9;
}

/* Form Group */
.form-group {
  margin-bottom: 15px;
}
```

```css
label {
  display: block;
  margin-bottom: 6px;
  font-weight: 500;
}

input, textarea {
  width: 100%;
  padding: 10px;
  border: 1px solid #ccc;
  border-radius: 6px;
  font-size: 1rem;
  outline: none;
}

/* Error and success messages */
.error-message {
  color: red;
  font-size: 0.85rem;
  margin-top: 5px;
  display: block;
}

.success-message {
  color: green;
  text-align: center;
  font-weight: 600;
  margin-top: 15px;
}
```

```css
/* 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;
}
```

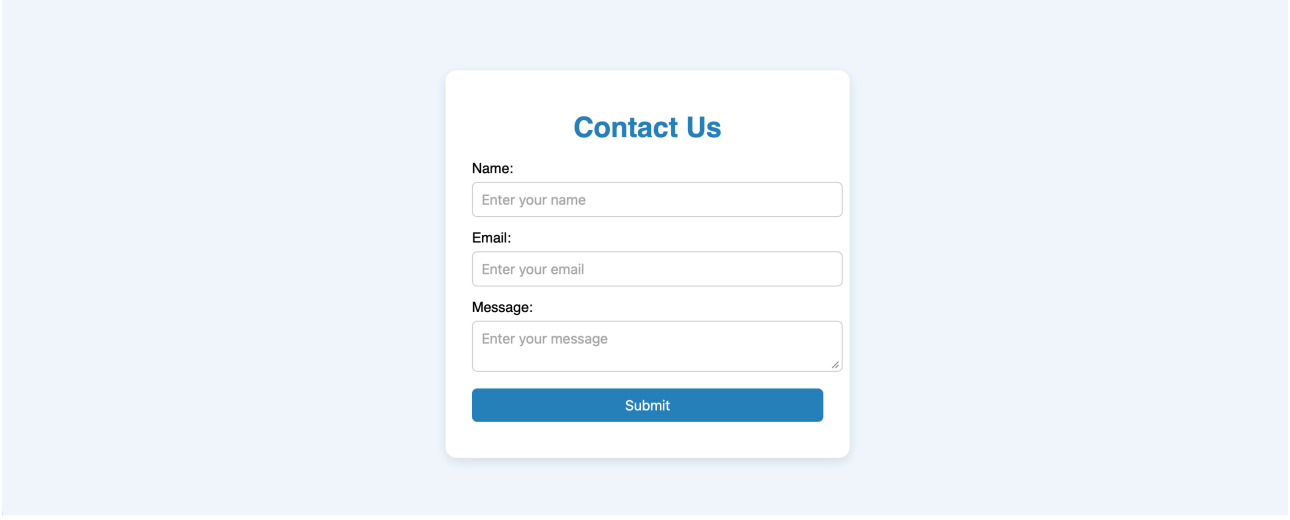
JAVASCRIPT CODE :

```
// Wait until the DOM is fully loaded
document.addEventListener("DOMContentLoaded", () => {

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

  // Form submit event
  form.addEventListener("submit", function(event) {
    event.preventDefault(); // Prevent form refresh

    // Get input elements
    const nameInput = document.getElementById("name");
    const emailInput = document.getElementById("email");
    const messageInput = document.getElementById("message");

    // Reset previous messages
    clearErrors();
    successMessage.textContent = "";

    // Validation flags
    let isValid = true;

    // Validate name
    if (nameInput.value.trim() === "") {
      showError(nameInput, "Name is required.");
      isValid = false;
    }
```

```
    // Validate email
    if (emailInput.value.trim() === "") {
      showError(emailInput, "Email is required.");
      isValid = false;
    } else if (!isValidEmail(emailInput.value.trim())) {
      showError(emailInput, "Please enter a valid email address.");
      isValid = false;
    }

    // Validate message
    if (messageInput.value.trim() === "") {
      showError(messageInput, "Message cannot be empty.");
      isValid = false;
    }

    // If valid, show success message
    if (isValid) {
      successMessage.textContent = "Form submitted successfully!";
      form.reset();
    }
  });

  // Helper function: show error below input
  function showError(inputElement, message) {
    const errorMsg = inputElement.parentElement.querySelector(".error-message");
    errorMsg.textContent = message;
  }
```

```
  // 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 :



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 :

```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 Example</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>

  <div class="container">
    <h1>My Product List</h1>

    <!-- List of items -->
    <ul id="itemList">
      <li>Product 1</li>
      <li>Product 2</li>
      <li>Product 3</li>
    </ul>

    <!-- Buttons -->
    <div class="buttons">
      <button id="addBtn">Add Item</button>
      <button id="removeBtn">Remove Item</button>
    </div>
  </div>

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

CSS CODE :

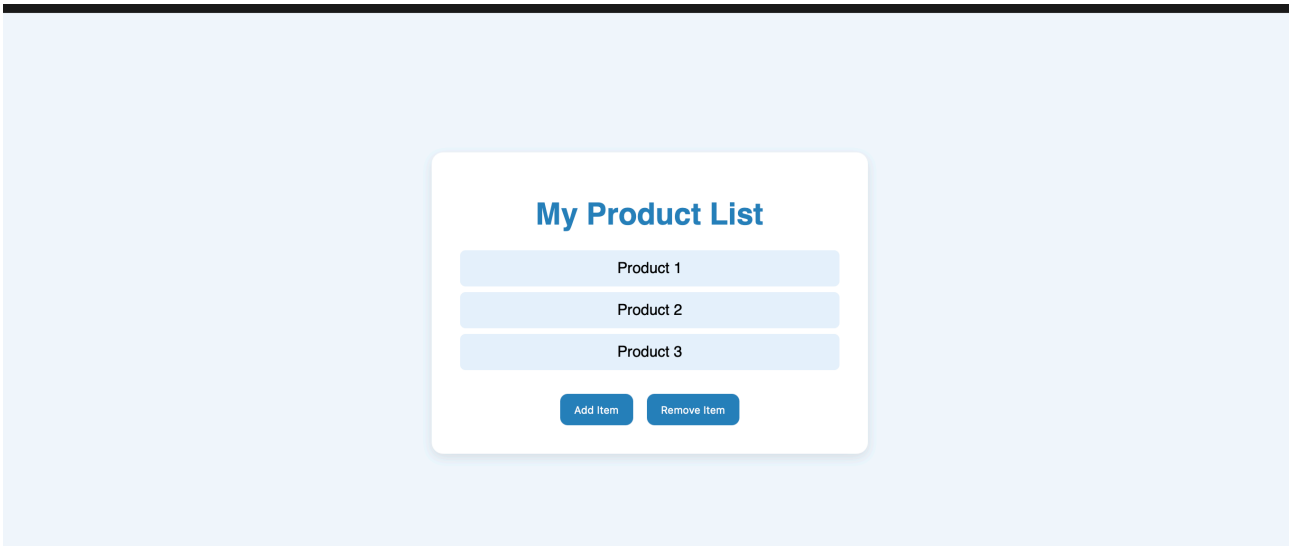
```css
body {
  font-family: 'Poppins', sans-serif;
  background-color: #f0f8ff;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
}

.container {
  background: white;
  padding: 25px 30px;
  border-radius: 12px;
  box-shadow: 0 4px 12px rgba(0,0,0,0.1);
  text-align: center;
  width: 90%;
  max-width: 400px;
}

h1 {
  color: #2980b9;
  margin-bottom: 20px;
}

ul {
  list-style-type: none;
  padding: 0;
  margin-bottom: 20px;
}
```

```css
li {
  background-color: #e3f2fd;
  margin: 6px 0;
  padding: 10px;
  border-radius: 6px;
}

button {
  background-color: #2980b9;
  color: white;
  border: none;
  padding: 10px 15px;
  border-radius: 8px;
  cursor: pointer;
  margin: 5px;
  transition: 0.3s;
}

button:hover {
  background-color: #1e6fa1;
}
```

JAVASCRIPT CODE :

```javascript
// AI-generated JavaScript for dynamic list updates

// Get references to HTML elements
const addBtn = document.getElementById('addBtn');
const removeBtn =
document.getElementById('removeBtn');
const itemList = document.getElementById('itemList');

// Counter to track number of products
let itemCount = itemList.children.length;

// Add a new item when "Add Item" is clicked
addBtn.addEventListener('click', () => {
  itemCount++;
  const newItem = document.createElement('li');
  newItem.textContent = `Product ${itemCount}`;
  itemList.appendChild(newItem);
});

// Remove the last item when "Remove Item" is clicked
removeBtn.addEventListener('click', () => {
  if (itemList.lastElementChild) {
    itemList.removeChild(itemList.lastElementChild);
    itemCount--;
  } else {
    alert("No more items to remove!");
  }
});
```

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:

```html
<!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 :

```css
/* Basic Page Styling */
body {
  font-family: 'Poppins', sans-serif;
  background-color: #f0f8ff;
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
  margin: 0;
}

.container {
  text-align: center;
}

button {
  background-color: #2980b9;
  color: white;
  border: none;
  padding: 12px 20px;
  border-radius: 8px;
  cursor: pointer;
  font-size: 1rem;
  transition: 0.3s;
}

button:hover {
  background-color: #1e6fa1;
}
```

```css
/* 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;
}
```

```css
/* Close Button */
.close-btn {
  position: absolute;
  top: 10px;
  right: 15px;
  font-size: 1.5rem;
  color: #555;
  cursor: pointer;
  transition: 0.2s;
}

.close-btn:hover {
  color: #e74c3c;
}

/* Simple animation for modal */
@keyframes slideDown {
  from {
    transform: translateY(-30px);
    opacity: 0;
  }
  to {
    transform: translateY(0);
    opacity: 1;
  }
}
```

JAVASCRIPT CODE :

```javascript
// AI-generated JavaScript for Modal Popup

// Get DOM elements
const openModalBtn = document.getElementById('openModalBtn');
const modal = document.getElementById('modal');
const closeBtn = document.querySelector('.close-btn');

// Open modal when button clicked
openModalBtn.addEventListener('click', () => {
  modal.style.display = 'flex'; // Show modal
});

// Close modal when "X" is clicked
closeBtn.addEventListener('click', () => {
  modal.style.display = 'none'; // Hide modal
});

// Close modal when user clicks outside the modal box
window.addEventListener('click', (event) => {
  if (event.target === modal) {
    modal.style.display = 'none';
  }
});
```

WEBPAGE LAYOUT :