

## AI ASSISTED CODING

### LAB-14: Web Design Application – AI-Assisted HTML/CSS/JSGeneration

**Roll no:** 2503A51L44

**Name:** Meer Burhan Ali Hashmi

**Batch:** 25BTCAICSB20

**Task -1 Description:** 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

#### **Code Given to the AI:**

##### **HTML Code:**

```
task1-14.html # style.css
Assignment-14 > task-1 > task1-14.html > html > body > header#home > nav
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>Task-1</title>
7      <link rel="stylesheet" href="style.css">
8  </head>
9  <body>
10     <header id="home">
11         <h1>Welcome to AI Assisted Coding</h1>
12         <nav>
13             <a href="#home">Home</a> | 
14             <a href="#about">About</a> | 
15             <a href="#contact">Contact</a>
16         </nav>
17     </header>
18
19     <main id="about">
20         <p>This is a simple HTML page with a header and a paragraph.</p>
21         <video width="320" height="240" controls>
22             <source src="video.mp4" type="video/mp4">
23             Your browser does not support the video tag.
24         </video>
25         
26     </main>
27
28     <footer id="contact">
29         <p style="text-align: center;">&copy; 2025 AI Assisted Coding</p>
30     </footer>
31 </body>
32 </html>
```

##### **CSS Code:**

```
task1-14.html # style.css
Assignment-14 > task-1 > style.css > a
1  header{
2      background-color: #lightblue;
3      padding: 20px;
4      text-align: center;
5      font-size: 15px;
6  }
7  h1{
8      text-align: center;
9      display: flexbox;
10     justify-content: center;
11     align-items: center;
12     background-color: #papayawhip;
13 }
14 a{
15     font-size: 20px;
16     text-decoration: none;
17     color: #blue;
18 }
```

**Prompt:** Assist in generating responsive CSS for different screen sizes. Ensure the layout is clean and visually organized.

#### **Code Generated:**

```

task1-14.html  # style.css
Assignment-14 > task1-14 > # style.css > style.css
1  /* Base styles for the document */
2  body {
3      margin: 0;
4      font-family: 'Segoe UI', Arial, sans-serif;
5      background: #f7f9fa;
6      color: #222;
7  }
8
9  /* Header styles */
10 header {
11     background: #2d3e50;
12     color: #fff;
13     padding: 1.5rem 0 1rem 0;
14     text-align: center;
15 }
16
17 header h1 {
18     margin: 0 0 0.5rem 0;
19     font-size: 2.2rem;
20     letter-spacing: 1px;
21 }
22
23 /* Navigation styles */
24 nav {
25     margin: 0.5rem 0 0 0;
26 }
27
28 nav a {
29     color: #fff;
30     text-decoration: none;
31     margin: 0 1rem;
32     font-weight: 500;
33     transition: color 0.2s;
34 }
35
36 nav a:hover {
37     color: #ffd700;
38 }
39
40 /* Main content styles */
41 main {
42     max-width: 700px;
43     margin: 2rem auto;
44     background: #fff;
45     border-radius: 8px;
46     box-shadow: 0 2px 8px rgba(44, 62, 88, 0.08);
47     padding: 2rem;
48     text-align: center;
49 }

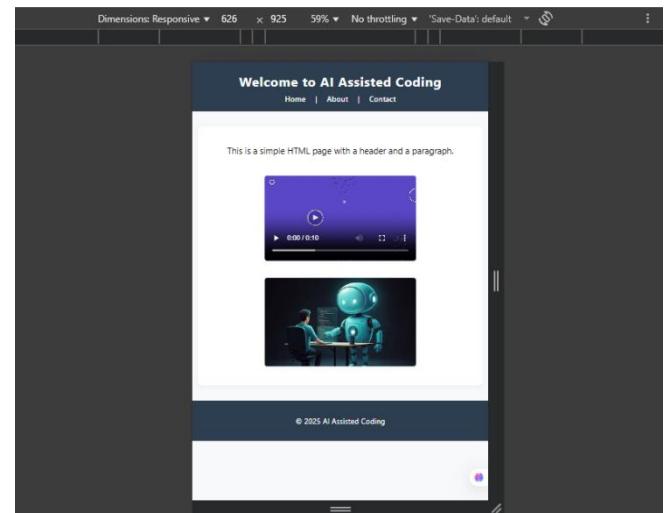
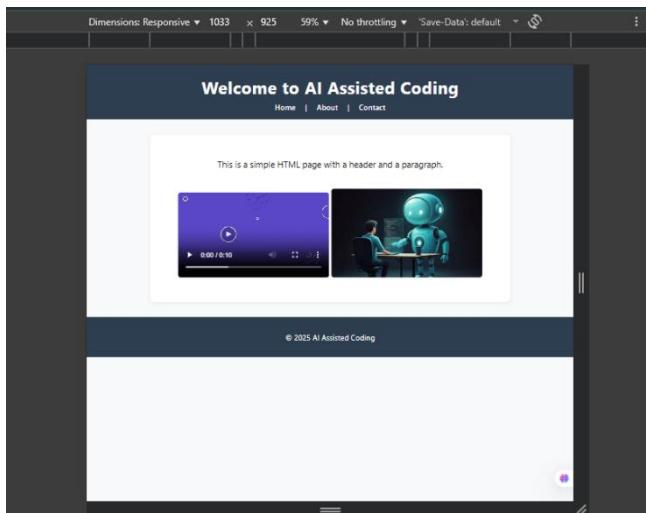
```

```

51 main p {
52     font-size: 1.15rem;
53     margin-bottom: 1.5rem;
54 }
55
56 /* Media elements styles */
57 video, img {
58     max-width: 100%;
59     height: auto;
60     margin: 1rem 0;
61     border-radius: 6px;
62     box-shadow: 0 1px 4px rgba(44, 62, 88, 0.1);
63 }
64
65 /* Footer styles */
66 footer {
67     background: #2d3e50;
68     color: #fff;
69     padding: 1rem 0;
70     margin-top: 2rem;
71     font-size: 1rem;
72 }
73
74 /* Responsive styles */
75 @media (max-width: 900px) {
76     main {
77         max-width: 90vw;
78         padding: 1.2rem;
79     }
80     header h1 {
81         font-size: 1.7rem;
82     }
83 }
84
85 @media (max-width: 600px) {
86     header, footer {
87         padding: 0.7rem 0;
88     }
89     main {
90         padding: 0.7rem;
91     }
92     nav a {
93         margin: 0 0.5rem;
94         font-size: 1rem;
95     }
96     header h1 {
97         font-size: 1.2rem;
98     }
99 }

```

Output: (Responsive website- Changes its size automatically based on device)



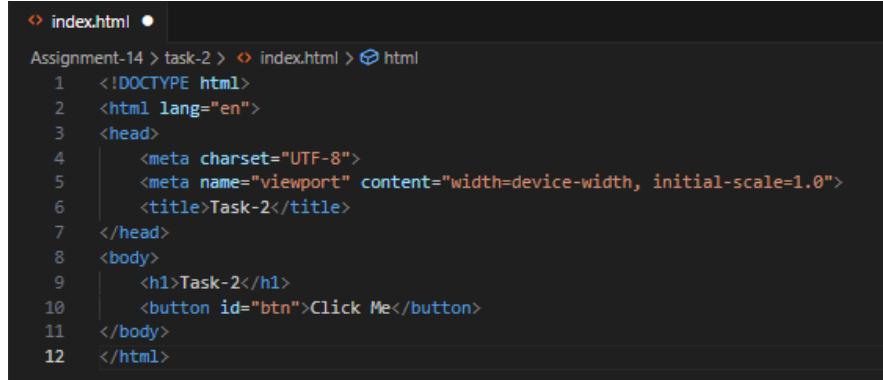
**Observation:** The responsive web page layout designed using HTML and AI-assisted CSS works efficiently across different screen sizes such as mobile, tablet, and desktop. The structure includes a well-defined header, main content area, and footer, maintaining a neat and balanced appearance. The AI-generated CSS code ensures flexibility and visual consistency when resizing. Overall, the layout demonstrates effective use of responsive design principles and clean styling.

## **Task-2 Description:**Interactive Button with JavaScript

### **Instructions:**

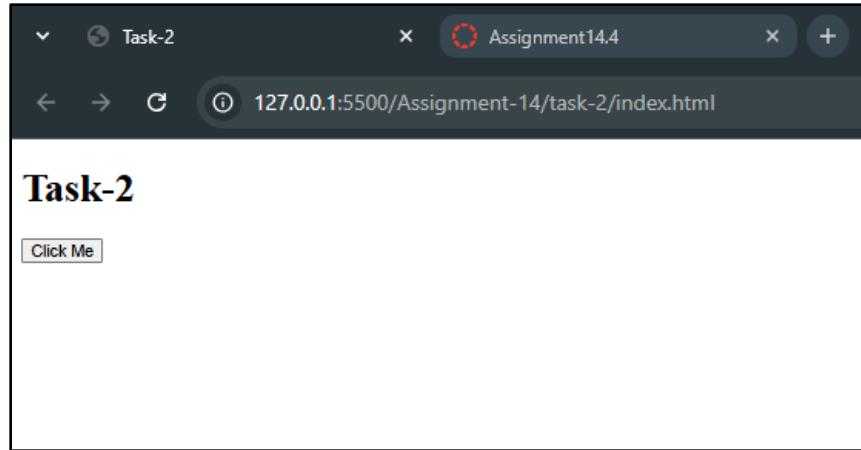
- Create a button on a web page.
- Use AI to generate JavaScript code that displays an alert message when the button is clicked.
- Ensure the code is clean and well-commented.

### **Code Given to AI:**



```
index.html •
Assignment-14 > task-2 > index.html > html
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>Task-2</title>
7  </head>
8  <body>
9      <h1>Task-2</h1>
10     <button id="btn">Click Me</button>
11  </body>
12 </html>
```

### **Output:**

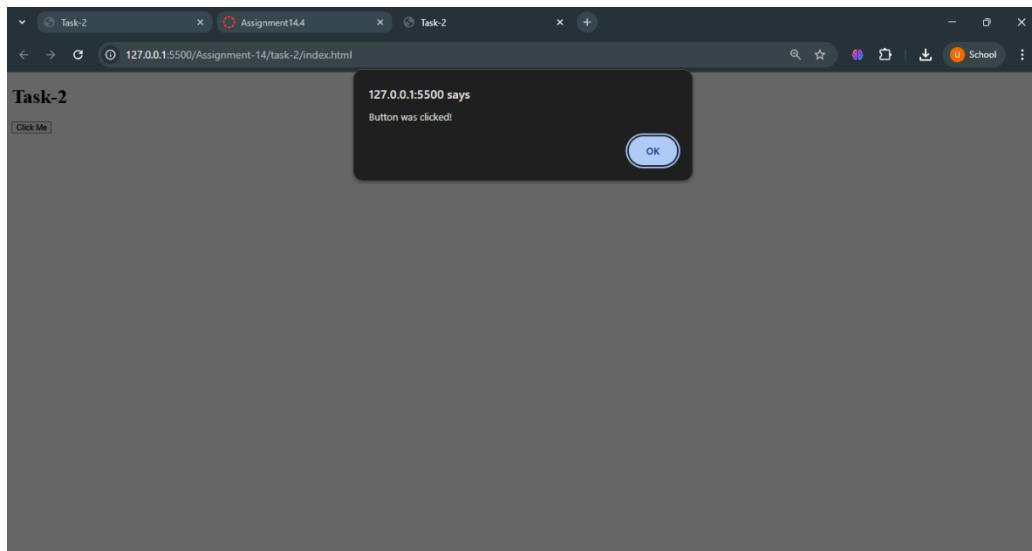


**Prompt:** Generate JavaScript code that displays an alert message when the button is clicked, also ensure the code is clean and well-commented.

## Code Generated by AI:

```
index.html X
Assignment-14 > task-2 > index.html > html
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>Task-2</title>
7  <script>
8      // Wait for the DOM to load before attaching event listeners
9      document.addEventListener('DOMContentLoaded', function() {
10          // Get the button by its ID
11          var btn = document.getElementById('btn');
12          // Add a click event listener to the button
13          btn.addEventListener('click', function() {
14              // Show an alert message when the button is clicked
15              alert('Button was clicked!');
16          });
17      });
18  </script>
19 </head>
20 <body>
21     <h1>Task-2</h1>
22     <button id="btn">Click Me</button>
23 </body>
24 </html>
```

## Output: (When button is clicked)



## Observation:

The interactive button functions properly by displaying an alert message when clicked, as per the task requirement. The JavaScript code generated by AI is clean, easy to understand, and includes meaningful comments that enhance clarity. The implementation successfully demonstrates user interaction on the webpage. This task highlights how AI assistance can simplify JavaScript coding for beginners while ensuring functional accuracy.

### **Task-3 Description:**Form with Validation

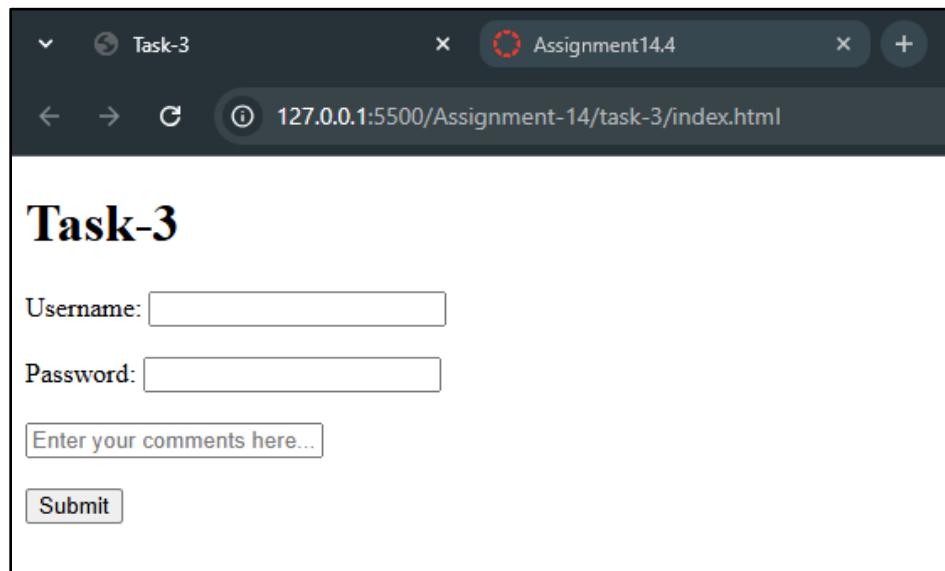
#### **Instructions:**

- 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.

#### **Code given to AI:**

```
index.html ●
Assignment-14 > task-3 > index.html > html
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>Document</title>
7  </head>
8  <body>
9      <h1>Task-3</h1>
10     <form>
11         <label for="username">Username:</label>
12         <input type="text" id="username" name="username" required>
13         <br><br>
14         <label for="password">Password:</label>
15         <input type="password" id="password" name="password" required>
16         <br><br>
17         <input type="textarea" id="comments" placeholder="Enter your comments here...">
18         <br><br>
19         <input type="submit" value="Submit">
20     </body>
21 </html>
```

#### **Output:**



**Prompt:**Generate JavaScript code for form validation (e.g.,non-empty fields, valid email format).Add inline error messages if input is invalid.

## Code

```
indexhtml ✘
Assignment-14 > task-3 > index.html > html > head > script > document.addEventListener('DOMContentLoaded') callback > form.addEventListener('submit', function(e) { ... })
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>Task-3</title>
7      <style>
8          .error { color: red; font-size: 0.95em; margin-left: 8px; }
9          label { display: block; margin-top: 1rem; }
10         input, textarea { margin-top: 0.3rem; }
11     </style>
12     <script>
13         // Wait for DOM to load
14         document.addEventListener('DOMContentLoaded', function() {
15             var form = document.getElementById('myForm');
16             form.addEventListener('submit', function(e) {
17                 // Clear previous errors
18                 document.querySelectorAll('.error').forEach(function(el) { el.textContent = '' });
19                 var valid = true;
20
21                 // Username validation
22                 var username = document.getElementById('username');
23                 if (!username.value.trim()) {
24                     document.getElementById('usernameError').textContent = 'Username is required.';
25                     valid = false;
26                 }
27                 // Password validation
28                 var password = document.getElementById('password');
29                 if (!password.value.trim()) {
30                     document.getElementById('passwordError').textContent = 'Password is required.';
31                     valid = false;
32                 }
33                 // Email validation
34                 var email = document.getElementById('email');
35                 var emailPattern = /^[^@\s]+@[^\s@]+\.[^\s@]+\$/;
36                 if (!email.value.trim()) {
37                     document.getElementById('emailError').textContent = 'Email is required.';
38                     valid = false;
39                 } else if (!emailPattern.test(email.value)) {
40                     document.getElementById('emailError').textContent = 'Enter a valid email address.';
41                     valid = false;
42                 }
43                 // Comments validation
44                 var comments = document.getElementById('comments');
45                 if (!comments.value.trim()) {
46                     document.getElementById('commentsError').textContent = 'Comments are required.';
47                     valid = false;
48                 }
49                 if (!valid) {
50                     e.preventDefault();
51                 }
52             });
53         });
54     </script>
55 </head>
56 <body>
57     <form id="myForm">
58         <div>
59             <label>Username:</label>
60             <input type="text" id="username" name="username" value="JohnDoe" />
61             <span class="error" id="usernameError"></span>
62         </div>
63         <div>
64             <label>Password:</label>
65             <input type="password" id="password" name="password" value="123456" />
66             <span class="error" id="passwordError"></span>
67         </div>
68         <div>
69             <label>Email:</label>
70             <input type="email" id="email" name="email" value="john.doe@example.com" />
71             <span class="error" id="emailError"></span>
72         </div>
73         <div>
74             <label>Comments:</label>
75             <input type="text" id="comments" name="comments" value="Hello World!" />
76             <span class="error" id="commentsError"></span>
77         </div>
78         <div>
79             <input type="submit" value="Submit" />
80         </div>
81     </form>
82 </body>
83 </html>
```

Generated:

The screenshot shows a web browser window titled "Assignment14.4" with the URL "127.0.0.1:5500/Assignment-14/task-3/index.html". The page title is "Task-3". The form contains the following fields:

- Username: Uzma
- Password: (redacted)
- Email: 123@gmail.com
- Comments: (empty text area with placeholder "Enter your comments here...")

An error message "Please fill out this field." is displayed in a red box next to the empty comments field.

## Output:

```
51 |     });
52 |   });
53 | 
```

```
54 | </head>
55 | <body>
56 | 
57 |   <h1>Task-3</h1>
58 |   <form id="myForm">
59 |     <label for="username">Username:</label>
60 |     <input type="text" id="username" name="username" required>
61 |     <span class="error" id="usernameError"></span>
62 | 
63 |     <label for="password">Password:</label>
64 |     <input type="password" id="password" name="password" required>
65 |     <span class="error" id="passwordError"></span>
66 | 
67 |     <label for="email">Email:</label>
68 |     <input type="email" id="email" name="email" required>
69 |     <span class="error" id="emailError"></span>
70 | 
71 |     <label for="comments">Comments:</label>
72 |     <textarea id="comments" name="comments" placeholder="Enter your comments here..." required></textarea>
73 |     <span class="error" id="commentsError"></span>
74 | 
75 |     <br><br>
76 |     <input type="submit" value="Submit">
77 |   </form>
78 | </body>
79 | </html>
```

## Observation:

The contact form includes fields for Name, Email, and Message, and effectively validates user input before submission. The AI-generated JavaScript detects empty fields and invalid email formats, displaying appropriate inline error messages. This enhances user experience and ensures

data accuracy. The validation logic is simple yet efficient, making the form interactive, reliable, and user-friendly.

## Task 4: AI-Assisted E-Commerce Product Page

### Scenario:

A startup wants a basic e-commerce product page to display products with prices and an “Add to Cart” button.

```
○ index.html > @head > @style > ↴ .container
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <title>Startup E-Commerce Product Page</title>
6    <meta name="viewport" content="width=device-width, initial-scale=1.0">
7    <style>
8      body {
9        font-family: 'Segoe UI', Arial, sans-serif;
10       background: #f7f7f7;
11       margin: 0;
12       padding: 0;
13     }
14     header {
15       background: #0222;
16       color: #fff;
17       padding: 1rem 0;
18       position: relative;
19     }
20     .container {
21       max-width: 1100px;
22       margin: 2rem auto;
23       padding: 0 1rem;
24     }
25     h1 {
26       margin: 0;
27       text-align: center;
28       font-size: 2rem;
29     }
30     .cart-counter {
31       position: absolute;
32       top: 1.2rem;
33       right: 2rem;
34       background: #fff;
35       color: #0222;
36       border-radius: 50%;
37       width: 2.2rem;
38       height: 2.2rem;
39       display: flex;
40       align-items: center;
41       justify-content: center;
42       font-weight: bold;
43       font-size: 1.1rem;
44       box-shadow: 0 2px 8px rgba(0,0,0,.08);
45     }
46     .product-grid {
47       display: grid;
48       grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
49       gap: 2rem;
50     }
51     .product-card {
52       background: #fff;
53       border-radius: 10px;
54       box-shadow: 0 2px 8px rgba(0,0,0,.08);
55       padding: 1.5rem;
56       text-align: center;
57       transition: transform 0.2s, box-shadow 0.2s;
58     }
59     .product-card:hover {
60       transform: translateX(-5px) scale(1.03);
61       box-shadow: 0 6px 20px rgba(255,152,0,.15);
62       border: 1.5px solid #ff9900;
63     }
64     .product-card img {
65       width: 120px;
66       height: 120px;
67       object-fit: cover;
68       border-radius: 8px;
69       margin-bottom: 1rem;
70     }
71     .product-card h3 {
72       margin-bottom: 0.5rem;
73       color: #0222;
74     }
75     .product-card .price {
76       color: #ff9900;
77       font-weight: bold;
78       font-size: 1.1rem;
79       margin-bottom: 1rem;
80       display: block;
81     }
82     .add-btn {
83       background: #ff9900;
84       color: #fff;
85       border: none;
86       padding: 0.7rem 1.5rem;
87       border-radius: 6px;
88       font-size: 1rem;
89       cursor: pointer;
90       font-weight: 500;
91       transition: background 0.2s;
92     }
93     .add-btn:hover {
94       background: #e68000;
95     }
96     @media (max-width: 600px) {
97       .container { padding: 0 0.5rem; }
98       .cart-counter { right: 1rem; }
99     }
100    </style>
```

- Use Copilot to generate a grid-based product catalog in HTML/CSS.
- Implement a JavaScript “Add to Cart” functionality with Copilot’s guidance.
- Modify Copilot’s suggestions to include a cart counter at the top-right corner of the page.

### Prompt:

A startup wants a basic e-commerce product page to display products

with prices and an “Add to Cart” button.

- generate a grid-based product catalog in HTML/CSS.
- Implement a JavaScript “Add to Cart” functionality
- include a cart counter at the top-right corner of the page. give html,css and js in the same file

### Code Generated:

## Output:

The screenshot shows a web page titled "Shop" with a header containing a cart counter. Below the header is a grid of five product cards. Each card displays an image, the product name, its price, and an "Add to Cart" button.

Product	Image	Price	Action
Wireless Headphones		rs.2,319	Add to Cart
Smart Watch		rs.1499	Add to Cart
Laptop		rs.89000	Add to Cart
Handbag		rs.899	Add to Cart
Bluetooth Speaker		rs.8,000	Add to Cart

```
<html>
</head>
<body>
  <header>
    <h1>Shop</h1>
    <div class="cart-counter" id="cartCounter">0</div>
  </header>
  <div class="container">
    <div class="product-card">
      <div class="product-grid">
        
        <h3>Wireless Headphones</h3>
        <span>rs.2,319</span>
        <button class="add-btn" onclick="addToCart()">Add to Cart</button>
      </div>
    </div>
    <div class="product-card">
      <div class="product-grid">
        
        <h3>Smart Watch</h3>
        <span>rs.1499</span>
        <button class="add-btn" onclick="addToCart()">Add to Cart</button>
      </div>
    </div>
    <div class="product-card">
      <div class="product-grid">
        
        <h3>Bluetooth Speaker</h3>
        <span>rs.8,000</span>
        <button class="add-btn" onclick="addToCart()">Add to Cart</button>
      </div>
    </div>
    <div class="product-card">
      <div class="product-grid">
        
        <h3>Fitness Tracker</h3>
        <span>rs.3,499</span>
        <button class="add-btn" onclick="addToCart()">Add to Cart</button>
      </div>
    </div>
    <div class="product-card">
      <div class="product-grid">
        
        <h3>Laptop</h3>
        <span>rs.89000</span>
        <button class="add-btn" onclick="addToCart()">Add to Cart</button>
      </div>
    </div>
    <div class="product-card">
      <div class="product-grid">
        
        <h3>Handbag</h3>
        <span>rs.899</span>
        <button class="add-btn" onclick="addToCart()">Add to Cart</button>
      </div>
    </div>
    <div class="product-card">
      <div class="product-grid">
        
        <h3>Show</h3>
        <span>rs.1000</span>
        <button class="add-btn" onclick="addToCart()">Add to Cart</button>
      </div>
    </div>
  </div>
  <script>
    // Simple cart counter logic
    let cartCount = 0;
    function addToCart() {
      cartCount++;
      document.getElementById("cartCounter").textContent = cartCount;
    }
  </script>
</body>
</html>
```

## Observation:

The code creates a basic e-commerce product page using HTML, CSS, and JavaScript in a single file. It displays products in a responsive grid layout with images, names, prices, and an “Add to Cart” button. A cart counter is placed at the top-right corner inside the header and dynamically updates whenever the user clicks “Add to Cart.” The implementation is simple, lightweight, and serves as a starting point for building a full shopping cart system.