

Name:BANDI SRISHANTH

Batch:06

Hall Tno. :2403A510G3

Task 1 – Portfolio Website Design

You are building a personal portfolio website to showcase your work.

Requirements:

- Create sections for About Me, Projects, and Contact.
- Use AI to:
 - Suggest color palettes and typography.
 - Create a responsive layout with Grid/Flexbox.
 - Add smooth scrolling navigation

Prompt: Build a Personal Portfoio website using html css js.it should contain section :about me,projects,contact and it should be responsive layout with grid/flexbox and smooth scrolling navigation

Codes:

Screenshot of a code editor showing an HTML file named index.html. The code is a portfolio website for Abhishek Verma, a FullStack & AI Engineer. It includes sections for About Me, Education, Skills, Hobbies, and Projects. The code uses semantic HTML and CSS classes for styling.

```

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Abhishek Verma | FullStack & AI Engineer</title>
    <link rel="stylesheet" href="style.css">
    <!-- Fonts from Google Fonts -->
    <link href="https://fonts.googleapis.com/css2?family=Montserrat:wght@400,700&family=Open+Sans:wght@300,400&display=swap" rel="stylesheet">
  </head>
  <body>
    <!-- Header and Navigation -->
    <header class="main-header">
      <div class="logo">Abhishek Verma</div>
      <nav>
        <a href="#about">About Me</a>
        <a href="#projects">Projects</a>
        <a href="#contact">Contact</a>
      </nav>
    </header>
    <main>
      <!-- About Me Section -->
      <section id="about" class="section-container">
        <h1 class="intro-name">Abhishek Verma</h1>
        <p class="subtitle">FullStack Developer and AI engineer | Nawalparasi, Nepal</p>
        <!-- ADDED PHOTO CODE HERE -->
        
        <div class="bio-content">
          <div class="profile-summary">
            <p>I love coding! As a FullStack Developer and AI Engineer, I focus on building robust, scalable solutions using modern web technologies and applying machine learning principles to solve complex problems. My expertise lies in creating efficient and user-friendly interfaces using HTML, CSS, and JavaScript, while utilizing databases like MySQL and MongoDB for backend operations. I am also well-versed in AI/ML, having worked on projects involving natural language processing and computer vision. In my free time, I enjoy coding, reading, and exploring new technologies. I am always looking for opportunities to learn and grow in my field.</p>
          </div>
          <div class="details-section">
            <h2>Education</h2>
            <ul class="education-list">
              <li>**B.Tech CSE** 9.5/10</li>
              <li>12th class: 3.8/4</li>
              <li>11th class: 3.27/4</li>
              <li>10th class: 3.95/4</li>
            </ul>
          </div>
          <div class="details-section">
            <h2>Skills</h2>
            <ul class="skills-list">
              <li>HTML, CSS, JavaScript</li>
              <li>Python, Java, C</li>
              <li>FullStack Development</li>
              <li>AI/ML Engineering</li>
            </ul>
          </div>
        </div>
      </section>
      <!-- Projects Section (Unchanged) -->
      <section id="projects" class="section-container">
        <h2>Featured Works</h2>
        <div class="project-grid">
          <div class="project-card">
            <h3>AI-Powered Web App</h3>
            <p>A project demonstrating integration of Python (AI Backend) with JavaScript (Frontend) for real-time data processing.</p>
            <a href="#" class="btn">View Case Study</a>
          </div>
          <div class="project-card">
            <h3>FullStack E-Commerce Platform</h3>
            <p>Details about a complete web application built using modern frameworks (e.g., Node.js, React) and database management.</p>
            <a href="#" class="btn">View Code</a>
          </div>
          <div class="project-card">
            <h3>C/Java Algorithm Design</h3>
            <p>Showcase of complex problem-solving or data structure implementation using C or Java.</p>
            <a href="#" class="btn">View Repository</a>
          </div>
        </div>
      </section>
    </main>
  </body>

```

```

48   <h2>Skills</h2>
49   <ul class="skills-list">
50     <li>HTML, CSS, JavaScript</li>
51     <li>Python, Java, C</li>
52     <li>FullStack Development</li>
53     <li>AI/ML Engineering</li>
54   </ul>
55 </div>
56
57 <div class="details-section">
58   <h2>Hobbies</h2>
59   <ul class="hobbies-list">
60     <li>Coding</li>
61     <li>Innovation</li>
62   </ul>
63 </div>
64 </div>
65
66 </section>
67
68 <!-- Projects Section (Unchanged) -->
69 <section id="projects" class="section-container">
70   <h2>Featured Works</h2>
71   <div class="project-grid">
72
73     <div class="project-card">
74       <h3>AI-Powered Web App</h3>
75       <p>A project demonstrating integration of Python (AI Backend) with JavaScript (Frontend) for real-time data processing.</p>
76       <a href="#" class="btn">View Case Study</a>
77     </div>
78
79     <div class="project-card">
80       <h3>FullStack E-Commerce Platform</h3>
81       <p>Details about a complete web application built using modern frameworks (e.g., Node.js, React) and database management.</p>
82       <a href="#" class="btn">View Code</a>
83     </div>
84
85     <div class="project-card">
86       <h3>C/Java Algorithm Design</h3>
87       <p>Showcase of complex problem-solving or data structure implementation using C or Java.</p>
88       <a href="#" class="btn">View Repository</a>
89     </div>
90   </div>
91 </section>
92

```

```

93
94    <!-- Contact Section (Unchanged) -->
95    <section id="contact" class="section-container">
96        <h2>Get in Touch</h2>
97        <p>I am available for collaborations and new opportunities. You can reach me through the following:</p>
98
99        <div class="contact-info">
100            <p><strong>Email:</strong> <a href="mailto:kundanverma321@gmail.com">kundanverma321@gmail.com</a></p>
101            <p><strong>LinkedIn:</strong> <a href="#">Abhishek Verma</a> (Link Placeholder)</p>
102            <p><strong>Contact No.:</strong> +918765716460</p>
103        </div>
104
105        <form class="contact-form">
106            <input type="text" placeholder="Your Name" required>
107            <input type="email" placeholder="Your Email" required>
108            <textarea placeholder="Your Message"></textarea>
109            <button type="submit" class="btn">Send Message</button>
110        </form>
111    </section>
112 </main>
113
114 <footer>
115     <p>&copy; 2025 Abhishek Verma Portfolio</p>
116 </footer>
117
118 <script src="script.js"></script>
119 </body>
120 </html>

```

CSS

```

portfolio > style.css > ...
1 /* Color Palette (High Contrast Blue) and Typography (Modern Simplicity) */
2 :root {
3     --color-dark: #001C2B; /* Dark Navy (Background) */
4     --color-light: #E0E0E0; /* Light Gray (Text) */
5     --color-accent: #42A0F5; /* Sky Blue (Highlight/Accent) */
6     --color-card-bg: #FAFAE4; /* Slightly lighter dark for cards */
7
8     --font-heading: 'Montserrat', sans-serif;
9     --font-body: 'Open Sans', sans-serif;
10 }
11
12 /* Global Reset and Smooth Scrolling */
13 *
14 * {
15     box-sizing: border-box;
16     margin: 0;
17     padding: 0;
18 }
19 /* Smooth Scrolling Navigation */
20 html {
21     scroll-behavior: smooth;
22 }
23
24 body {
25     font-family: var(--font-body);
26     color: var(--color-light);
27     background-color: var(--color-dark);
28     line-height: 1.6;
29 }
30
31 h1, h2, h3 {
32     font-family: var(--font-heading);
33     margin-bottom: 0.5em;
34     color: var(--color-accent);
35 }
36
37 .section-container {
38     padding: 80px 5%;
39     min-height: 100vh;
40     display: flex;
41     flex-direction: column;
42     align-items: center;
43     text-align: center;
44 }
45
46 /* --- Header and Navigation (FLEXBOX) --- */
47 .main-header {
48     position: sticky;
49     top: 0;
50     width: 100%;
51     background-color: rgba(13, 28, 43, 0.95); /* Semi-transparent dark background */
}

```

The screenshot shows a code editor window with the 'style.css' file open. The file contains CSS declarations for colors, fonts, and layout. It includes a global reset, smooth scrolling for the html element, and a section container with a sticky header. The code editor has a dark theme, and the status bar at the bottom shows file details like 'Ln 259, Col 54' and 'Port: 5500'.

```
51 .main-header {
52   position: relative;
53   z-index: 1000;
54
55   /* Flexbox setup for Header */
56   display: flex;
57   justify-content: space-between;
58   align-items: center;
59   border-bottom: 1px solid var(--color-card-bg);
60 }
61
62 .logo {
63   font-family: var(--font-heading);
64   font-size: 1.5rem;
65   font-weight: bold;
66 }
67
68 nav a {
69   color: var(--color-light);
70   text-decoration: none;
71   margin-left: 25px;
72   font-weight: 400;
73   transition: color 0.3s ease;
74 }
75
76 nav a:hover {
77   color: var(--color-accent);
78 }
79
80 /* --- About Me Content Styling --- */
81 .intro-name {
82   font-size: 2.5rem;
83   margin-bottom: 5px;
84 }
85 .subtitle {
86   font-size: 1.1rem;
87   margin-bottom: 30px;
88   color: var(--color-light);
89 }
90
91 .bio-content {
92   max-width: 800px;
93   width: 100%;
94   margin-top: 20px;
95   text-align: left;
96 }
97
98 .profile-summary {
99   margin-bottom: 40px;
100  font-size: 1.1rem;
101  text-align: center;
102 }
```

```
.details-section {
  margin: 30px 0;
  padding: 20px;
  border: 1px solid var(--color-card-bg);
  border-radius: 5px;
  background-color: #12253a;
}

.details-section h2 {
  text-align: center;
  margin-bottom: 15px;
}

.education-list, .skills-list, .hobbies-list {
  list-style-type: none;
  padding-left: 0;
  display: flex;
  flex-wrap: wrap;
  justify-content: center;
}

.skills-list li, .education-list li, .hobbies-list li {
  background-color: var(--color-dark);
  padding: 8px 15px;
  margin: 5px;
  border-radius: 20px;
  border: 1px solid var(--color-accent);
}
/* If stacked list is preferred, remove display: flex and use traditional list-style-type: disc; */

/* --- Projects Section (CSS GRID) --- */
.project-grid {
  display: grid;
  gap: 30px;
  width: 100%;
  max-width: 1200px;
  margin-top: 40px;
}

.project-card {
  background-color: var(--color-card-bg);
  padding: 25px;
  border-radius: 8px;
  text-align: left;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.2);
  transition: transform 0.3s ease;
}

.project-card:hover {
```

```
167  /* Grid Responsiveness */
168
169  /* Mobile First (1 column) */
170  .project-grid {
171      grid-template-columns: 1fr;
172  }
173
174  /* Tablet Layout (2 columns) */
175  @media (min-width: 768px) {
176      .project-grid {
177          grid-template-columns: repeat(2, 1fr);
178      }
179  }
180
181  /* Desktop Layout (3 columns) */
182  @media (min-width: 1200px) {
183      .project-grid {
184          grid-template-columns: repeat(3, 1fr);
185      }
186  }
187
188
189  /* --- Contact Section Styling --- */
190  .contact-info {
191      margin: 20px auto 40px;
192      line-height: 2;
193      text-align: left;
194      max-width: 500px;
195      padding: 0 10px;
196  }
197
198  .contact-info a {
199      color: var(--color-accent);
200      text-decoration: none;
201  }
202
203  .contact-form {
204      display: flex;
205      flex-direction: column;
206      gap: 15px;
207      width: 100%;
208      max-width: 500px;
209  }
210
211  .contact-form input[type="text"],
212  .contact-form input[type="email"],
213  .contact-form textarea {
214      padding: 15px;
215      border: none;
216      border-radius: 5px;
217      background-color: var(--color-card-bg);
```

```

215     padding: 15px;
216     border: none;
217     border-radius: 5px;
218     background-color: var(--color-card-bg);
219     color: var(--color-light);
220     font-family: var(--font-body);
221   }
222 
223   .contact-form textarea {
224     resize: vertical;
225     min-height: 150px;
226   }
227 
228   .contact-form button.btn {
229     border: none;
230     cursor: pointer;
231     transition: background-color 0.3s ease;
232   }
233 
234   .contact-form button.btn:hover {
235     background-color: #64B5F6; /* Slightly lighter accent on hover */
236   }
237 
238   /* --- Footer --- */
239   footer {
240     padding: 20px;
241     text-align: center;
242     background-color: var(--color-dark);
243     border-top: 1px solid var(--color-card-bg);
244     font-size: 0.9rem;
245   }
246   /* --- Added Photo Styling --- */
247 
248   .profile-photo {
249     width: 150px; /* Set fixed size for the image */
250     height: 150px;
251     object-fit: cover; /* Ensures the image covers the area without distortion */
252     border-radius: 50%; /* Makes the image circular */
253     border: 4px solid var(--color-accent); /* Optional border using the accent color */
254     margin: 20px 0 40px 0;
255     box-shadow: 0 0 15px rgba(66, 165, 245, 0.5); /* Soft glow effect */
256   }
257 
258   /* --- About Me Content Styling (Existing styles) --- */
259   /* ... (rest of the CSS code remains the same) ... */

```

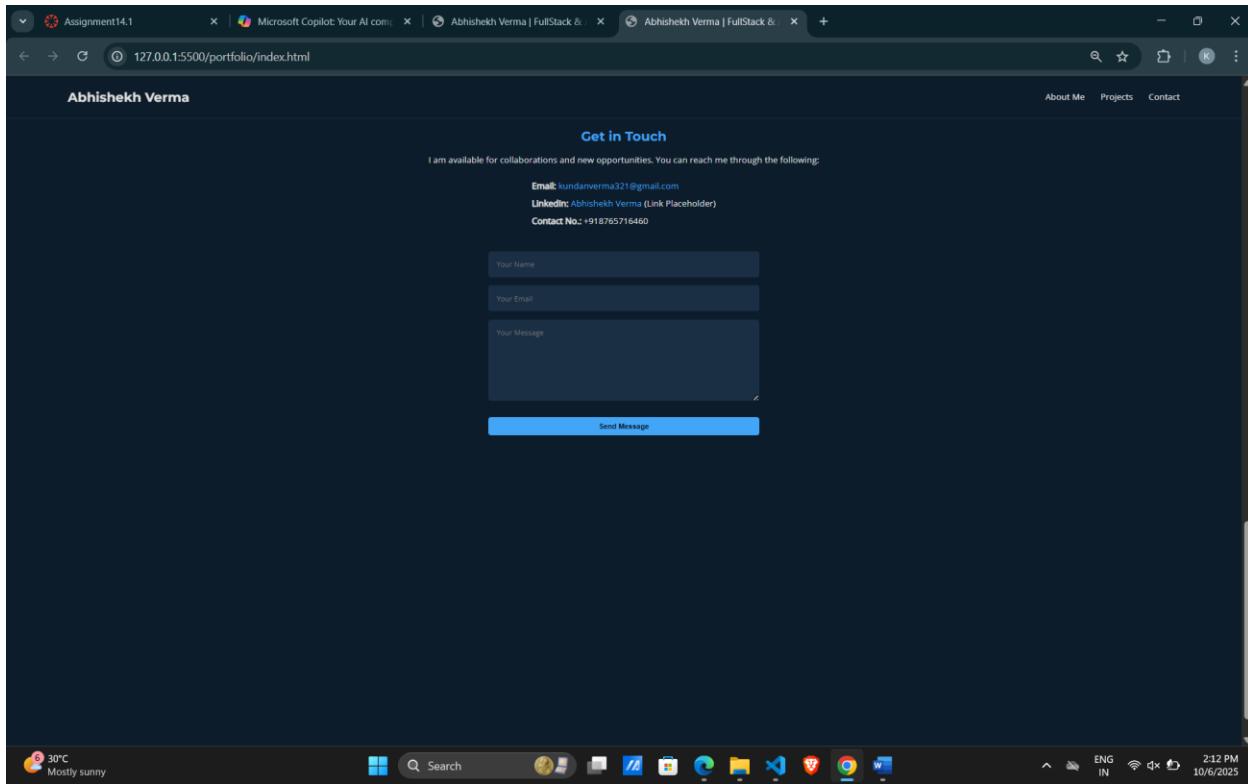
JS

```

portfolio > scripts > ...
1  /**
2   * script.js
3   * Portfolio interactivity and setup
4   */
5 
6 document.addEventListener('DOMContentLoaded', () => {
7   console.log("Abhishek Verma Portfolio website loaded successfully.");
8 
9   // Note: Smooth scrolling is handled by CSS (scroll-behavior: smooth).
10 
11  // Example: Basic form submission listener (prevents default page reload)
12  const contactForm = document.querySelector('.contact-form');
13  if (contactForm) {
14    contactForm.addEventListener('submit', function(event) {
15      event.preventDefault();
16 
17      // In a real application, this data would be sent to a server/service.
18      alert('Message simulated as sent! Thank you for contacting Abhishek Verma.');
19 
20      // Clear the form fields after submission (optional)
21      contactForm.reset();
22    });
23  }
24 
```

Output:

The screenshot displays a dark-themed portfolio website for Abhishek Verma. At the top, there's a navigation bar with links for 'About Me', 'Projects', and 'Contact'. Below the header, the name 'Abhishek Verma' is prominently displayed, followed by the title 'FullStack Developer and AI engineer | Nawalparasi, Nepal'. A circular profile picture of a young man is centered below the title. A brief bio states: 'I love coding! As a FullStack Developer and AI Engineer, I focus on building robust, scalable solutions using modern web technologies and applying machine learning principles. My goal is to merge development expertise with innovative AI applications.' Under the bio, there are sections for 'Education' and 'Skills'. The 'Education' section shows academic records: 'B.Tech CSE++ 9.5/10', '12th class: 3.28/4', '11th class: 3.27/4', and '10th class: 3.95/4'. The 'Skills' section lists technical competencies: 'HTML, CSS, JavaScript', 'Python, Java, C', 'FullStack Development', and 'AI/ML Engineering'. The bottom half of the page features a 'Hobbies' section with 'Coding' and 'Innovation' interests. The 'Featured Work' section highlights three projects: 'AI-Powered Web App', 'FullStack E-Commerce Platform', and 'C/Java Algorithm Design', each with a 'View Case Study' or 'View Repository' button. The website is viewed in a Microsoft Edge browser on a Windows 10 desktop, with the taskbar visible at the bottom showing various open applications like Microsoft Word, Excel, and Visual Studio Code.



Observation:

The code provides a **modern, responsive dark-mode portfolio** for Abhishek Verma.

It is well-organized using:

1. **HTML** for structure and personal details (skills, education).
2. **CSS** for aesthetics (dark blue theme, clean fonts) and layout, using **Flexbox** for navigation and **CSS Grid** for a responsive project gallery.
3. **Smooth Scrolling** is enabled via CSS for a polished user experience.
4. **JavaScript** is minimal, only handling the contact form submission gracefully.

Task 2 – Online Store Product Page

Design a product display page for an online store.

Requirements:

- Display product image, title, price, and "Add to Cart" button.
- Use AI to:
 - Style with BEM methodology.
 - Make layout responsive.
 - Add hover effects and "Add to Cart" alert

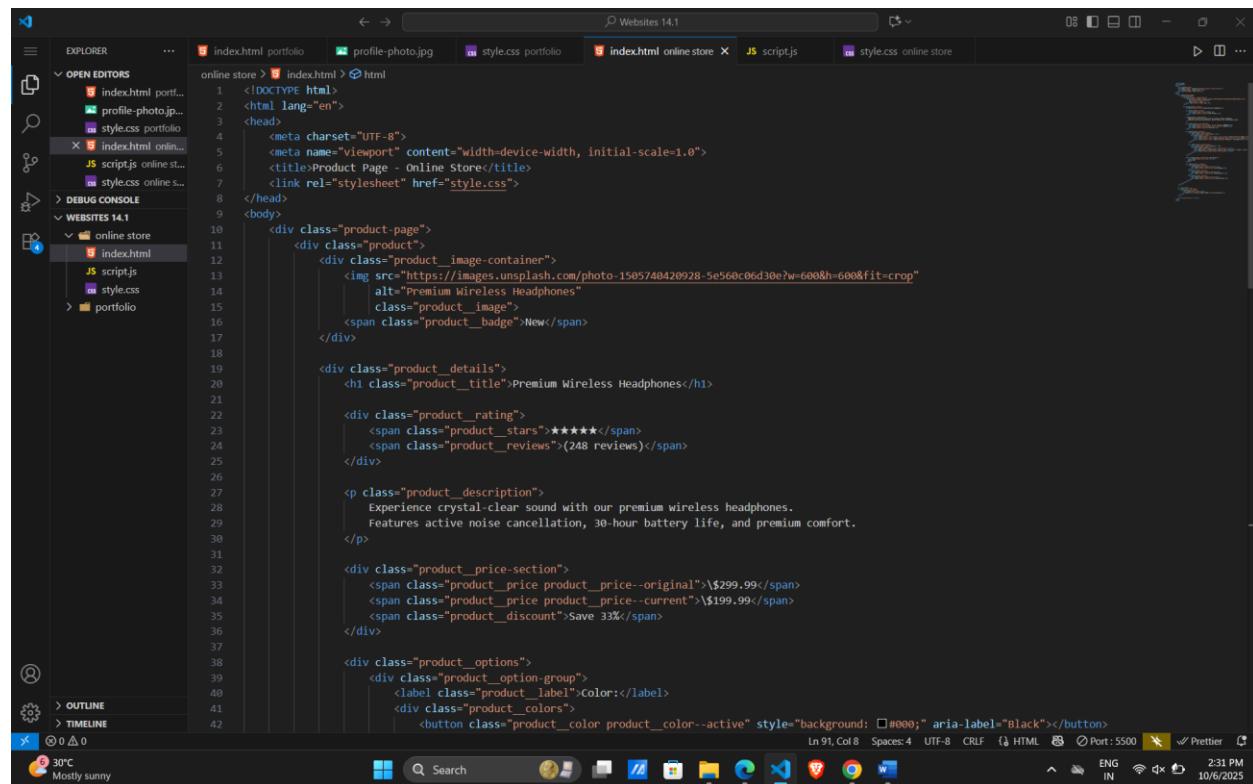
Prompt:

Generate the code for a responsive product page with an image, title, price, and an "Add to Cart" button.

Style everything using the BEM methodology and add interactive hover effects.

Make the "Add to Cart" button trigger a JavaScript alert when clicked.

Code:



The screenshot shows a code editor interface with the following files open:

- index.html (online store)
- script.js (online store)
- style.css (online store)

The index.html file contains the following code:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Product Page - Online Store</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <div class="product-page">
      <div class="product">
        <div class="product__image-container">
          
          <span class="product__badge">New</span>
        </div>
        <div class="product__details">
          <h1 class="product__title">Premium Wireless Headphones</h1>
          <div class="product__rating">
            <span class="product__stars">★★★★★</span>
            <span class="product__reviews">(248 reviews)</span>
          </div>
          <p class="product__description">
            Experience crystal-clear sound with our premium wireless headphones.
            Features active noise cancellation, 30-hour battery life, and premium comfort.
          </p>
          <div class="product__price-section">
            <span class="product__price product__price--original">$299.99</span>
            <span class="product__price product__price--current">$199.99</span>
            <span class="product__discount">Save 33%</span>
          </div>
          <div class="product__options">
            <div class="product__option-group">
              <label class="product__label">Color:</label>
              <div class="product__colors">
                <button class="product__color product__color--active" style="background: #000; aria-label="Black"></button>
                <button class="product__color product__color--active" style="background: #fff; aria-label="White"></button>
                <button class="product__color product__color--active" style="background: #ccc; aria-label="Grey"></button>
                <button class="product__color product__color--active" style="background: #f00; aria-label="Red"></button>
                <button class="product__color product__color--active" style="background: #0f0; aria-label="Green"></button>
                <button class="product__color product__color--active" style="background: #00f; aria-label="Blue"></button>
                <button class="product__color product__color--active" style="background: #999; aria-label="Silver"></button>
                <button class="product__color product__color--active" style="background: #666; aria-label="Dark Grey"></button>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </body>
</html>
```

```

42         <button class="product_color product_color--active" style="background: #000; aria-label="Black"></button>
43         <button class="product_color" style="background: #fff; border: 1px solid #ddd; aria-label="White"></button>
44     </div>
45   </div>
46
47   <div class="product_option_group">
48     <label class="product_label">Quantity:</label>
49     <div class="product_quantity">
50       <button class="product_quantity-btn" id="decrease"><-></button>
51       <input type="number" class="product_quantity-input" id="quantity" value="1" min="1" max="10">
52       <button class="product_quantity-btn" id="increase"><+></button>
53     </div>
54   </div>
55
56 </div>
57
58 <button class="product__add-to-cart" id="addToCart">
59   <span class="product__cart-icon">🛒</span>
60   Add to Cart
61 </button>
62
63 <div class="product__features">
64   <div class="product__feature">
65     <span class="product__feature-icon">🚚</span>
66     <span class="product__feature-text">Free Shipping</span>
67   </div>
68   <div class="product__feature">
69     <span class="product__feature-icon">>Returns</span>
70     <span class="product__feature-text">30-Day Returns</span>
71   </div>
72   <div class="product__feature">
73     <span class="product__feature-icon">✓</span>
74     <span class="product__feature-text">2-Year Warranty</span>
75   </div>
76 </div>
77 </div>
78
79 <!-- Alert Modal -->
80 <div class="alert" id="alert">
81   <div class="alert_content">
82     <span class="alert_icon">✓</span>
83     <p class="alert_message">Product added to cart successfully!</p>
84   </div>
85 </div>
86
87 <script src="script.js"></script>
88
89 </body>
90 </html>

```

Css



The screenshot shows a Microsoft Edge browser window displaying a dark-themed website. The address bar shows 'Websites 14.1'. The page content includes a header with a logo and navigation links, followed by a main section with a large image and descriptive text. The footer contains social media links and a copyright notice.

OPEN EDITORS

EXPLORER

WEBSITES

style.css online store

index.html portfolio profile-photo.jpg style.css portfolio index.html online store script.js

style.css online store

style.css

index.html

script.js

style.css

portfolio

outline

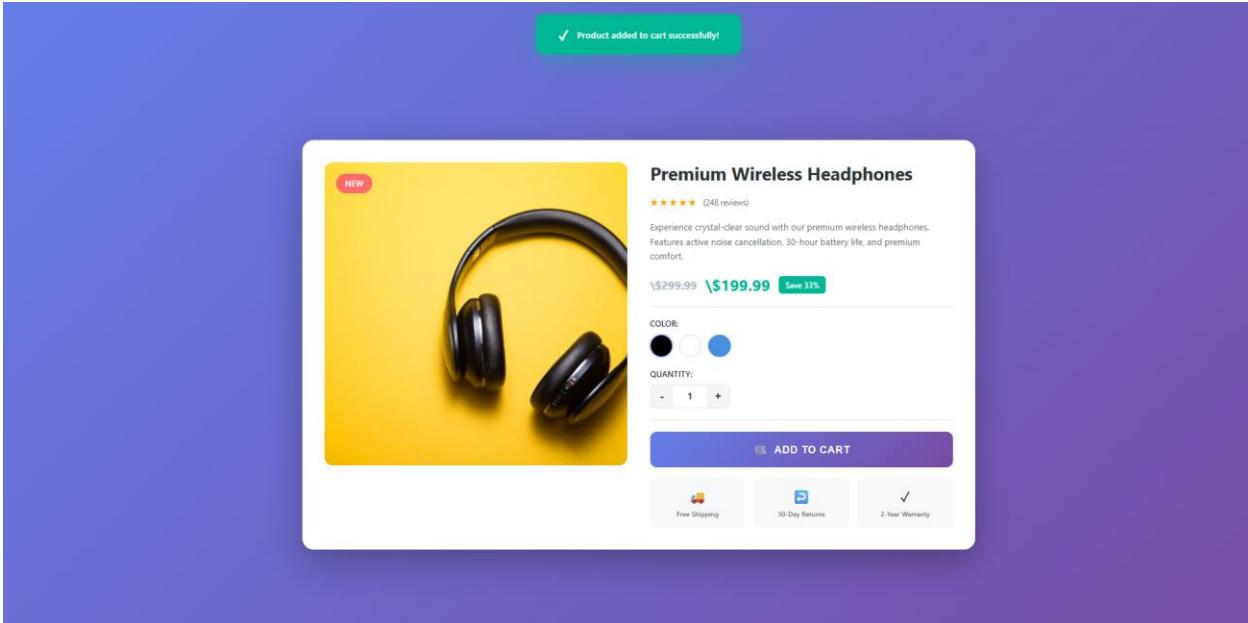
timeline

30°C Mostly sunny

Ln 380, Col 2 Spaces: 4 CRLF ⚙ CSS ⚙ Port: 5500 ENG IN 2:32 PM 10/6/2025

```
online store > style.css > {} @media (max-width: 640px)
1  /* style.css */
2
3  /* Reset & Base Styles */
4  *
5    margin: 0;
6    padding: 0;
7    box-sizing: border-box;
8
9
10 body {
11   font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
12   background: linear-gradient(135deg, #667eea 0%, #76ba2 100%);
13   min-height: 100vh;
14   padding: 20px;
15   display: flex;
16   align-items: center;
17   justify-content: center;
18 }
19
20 /* Product Block */
21 .product-page {
22   width: 100%;
23   max-width: 1200px;
24   margin: 0 auto;
25 }
26
27 /* Product Block */
28 .product {
29   background: #ffffff;
30   border-radius: 20px;
31   box-shadow: 0 20px 60px rgba(0, 0, 0, 0.3);
32   overflow: hidden;
33   display: grid;
34   grid-template-columns: 1fr 1fr;
35   gap: 40px;
36   padding: 40px;
37   transition: transform 0.3s ease;
38 }
39
40 .product:hover {
41   transform: translate(-5px);
42   box-shadow: 0 25px 70px rgba(0, 0, 0, 0.4);
}
30
31 .alert__icon {
32   font-size: 24px;
33   font-weight: 700;
34 }
35
36 .alert__message {
37   font-size: 16px;
38   font-weight: 600;
39 }
40
41 /* Responsive Design */
42 @media (max-width: 968px) {
43   .product {
44     grid-template-columns: 1fr;
45     gap: 30px;
46     padding: 30px;
47   }
48
49   .product__title {
50     font-size: 28px;
51   }
52 }
53
54 @media (max-width: 640px) {
55   body {
56     padding: 10px;
57   }
58
59   .product {
60     padding: 20px;
61     gap: 20px;
62   }
63
64   .product__title {
65     font-size: 24px;
66   }
67
68   .product__features {
69     grid-template-columns: 1fr;
70   }
71
72   .product__price-section {
73     flex-direction: column;
74     align-items: flex-start;
75   }
76
77   .alert {
78     width: 90%;
79     padding: 15px 20px;
80   }
}
```

Output:



Observation:

- The HTML structure is clean and uses the BEM methodology for class names, which makes the relationship between the markup and styling very clear.
- CSS is used effectively to create a modern, visually appealing card design, complete with subtle hover effects that improve the user experience.
- The design is fully responsive, using a media query to adjust the layout and ensure the product looks good on smaller mobile devices.
- JavaScript is minimal and efficient, using an event listener to provide immediate and direct feedback to the user upon a successful action.
- Overall, the code demonstrates a strong separation of concerns, with HTML for content, CSS for presentation, and JavaScript for behavior.

Task 3 – Event Registration Form

Build an event registration form for a conference.

Requirements:

- Collect name, email, phone number, and session selection.
- Use AI to:
 - Add form validation with JavaScript.
 - Make the form accessible with labels and ARIA.
 - Style with a professional look

Prompt: make an event registration form for a conference that collect name, email, phone number, and session selection style it in professional way via css add js also

Code:

```
online event > 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>Conference Registration</title>
7       <link rel="stylesheet" href="style.css">
8     </head>
9     <body>
10
11    <main class="form-container">
12      <h1>Conference Registration</h1>
13      <p>Please fill out the form below to register for the event.</p>
14
15      <form id="registrationForm" novalidate>
16        <!-- Name Field -->
17        <div class="form-group">
18          <label for="fullName">Full Name</label>
19          <input type="text" id="fullName" name="fullName" placeholder="e.g., Jane Doe" required aria-required="true" aria-describedby="nameError">
20          <div id="nameError" class="error-message" aria-live="polite"></div>
21        </div>
22
23        <!-- Email Field -->
24        <div class="form-group">
25          <label for="email">Email Address</label>
26          <input type="email" id="email" name="email" placeholder="e.g., jane.doe@example.com" required aria-required="true" aria-describedby="emailError">
27          <div id="emailError" class="error-message" aria-live="polite"></div>
28        </div>
29
30        <!-- Phone Number Field -->
31        <div class="form-group">
32          <label for="phone">Phone Number</label>
33          <input type="tel" id="phone" name="phone" placeholder="e.g., (123) 456-7890" required aria-required="true" aria-describedby="phoneError">
34          <div id="phoneError" class="error-message" aria-live="polite"></div>
35        </div>
36
37        <!-- Session Selection Field -->
38        <div class="form-group">
39          <label for="session">Select a Session</label>
40          <select id="session" name="session" required aria-required="true" aria-describedby="sessionError">
41            <option value="" disabled selected>Choose your session...</option>
42            <option value="morning-keynote">Morning Keynote: The Future of AI</option>
```

Css

```
1  /* General Body Styles */
2  body {
3    font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, Helvetica, Arial, sans-serif;
4    background-color: #f4f7f6;
5    color: #333;
6    display: flex;
7    justify-content: center;
8    align-items: center;
9    min-height: 100vh;
10   margin: 0;
11   padding: 20px;
12   box-sizing: border-box;
13 }
14
15 /* Form Container */
16 .form-container {
17   background-color: #ffffff;
18   padding: 40px;
19   border-radius: 8px;
20   box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
21   width: 100%;
22   max-width: 500px;
23 }
24
25 h1 {
26   margin-top: 0;
27   margin-bottom: 10px;
28   font-size: 24px;
29   color: #2c3e50;
30 }
31
32 p {
33   margin-bottom: 30px;
34   color: #555;
35 }
36
37 /* Form Grouping */
38 .form-group {
39   margin-bottom: 20px;
40 }
41
42 /* Labels */
43 label {
44   display: block;
45   font-weight: 600;
46   margin-bottom: 8px;
47   color: #34495e;
48 }
49
50 /* Input and Select Fields */
51 input[type="text"],
52 input[type="email"],
53 input[type="tel"],
54 select {
55   width: 100%;
56   padding: 12px;
57   border: 1px solid #ccc;
58   border-radius: 4px;
59   box-sizing: border-box; /* Ensures padding doesn't affect width */
60   font-size: 16px;
61   transition: border-color 0.3s, box-shadow 0.3s;
62 }
```

```
64  /* Focus State for Inputs */
65  input:focus,
66  ✓ select:focus {
67    border-color: #3498db;
68    box-shadow: 0 0 0 3px rgba(52, 152, 219, 0.2);
69    outline: none;
70  }
71
72  /* Submit Button */
73  ✓ button {
74    width: 100%;
75    padding: 15px;
76    background-color: #3498db;
77    color: #ffffff;
78    border: none;
79    border-radius: 4px;
80    font-size: 18px;
81    font-weight: 600;
82    cursor: pointer;
83    transition: background-color 0.3s;
84  }
85
86  ✓ button:hover {
87    background-color: #2980b9;
88  }
89
90  /* Error Message Styling */
91  ✓ .error-message {
92    color: #e4c3c;
93    font-size: 14px;
94    margin-top: 5px;
95    min-height: 1em; /* Prevents layout shift when messages appear */
96  }
97
98  /* Input Error State */
99  input.input-error,
100  ✓ select.input-error {
101    border-color: #e74c3c;
102  }
103
104  input.input-error:focus,
105  ✓ select.input-error:focus {
106    box-shadow: 0 0 0 3px rgba(231, 76, 60, 0.2);
107  }
```

Javascript:

```
online event > Js script.js > ...
1  document.addEventListener('DOMContentLoaded', () => {
2    const form = document.getElementById('registrationForm');
3
4    // Input Fields
5    const fullName = document.getElementById('fullName');
6    const email = document.getElementById('email');
7    const phone = document.getElementById('phone');
8    const session = document.getElementById('session');
9
10   // Error Message Divs
11   const nameError = document.getElementById('nameError');
12   const emailError = document.getElementById('emailError');
13   const phoneError = document.getElementById('phoneError');
14   const sessionError = document.getElementById('sessionError');
15
16   form.addEventListener('submit', (event) => {
17     // Prevent default form submission to handle validation
18     event.preventDefault();
19
20     // Perform validation and check if the form is valid
21     const isValidForm = validateForm();
22
23     if (isValidForm) {
24       // In a real application, you would submit the data to a server here.
25       alert('Registration Successful! Thank you for registering.');
26       form.reset(); // Clear the form fields
27
28       // Clear any lingering error states
29       clearAllErrors();
30     }
31   });
32
33   function validateForm() {
34     // Run validation for each field and store the result
35     const isValidName = validateName();
36     const isValidEmail = validateEmail();
37     const isValidPhone = validatePhone();
38     const isValidSession = validateSession();
39
40     // Return true only if all fields are valid
41     return isValidName && isValidEmail && isValidPhone && isValidSession;
42   }
43
44   function setError(inputElement, errorElement, message) {
45     inputElement.classList.add('input-error');
46     inputElement.setAttribute('aria-invalid', 'true');
47     errorElement.textContent = message;
48   }
49
50   function clearError(inputElement, errorElement) {
51     inputElement.classList.remove('input-error');
52     inputElement.setAttribute('aria-invalid', 'false');
53     errorElement.textContent = '';
54   }
55
56   function clearAllErrors() {
57     clearError(fullName, nameError);
58     clearError(email, emailError);
59     clearError(phone, phoneError);
60     clearError(session, sessionError);
61   }
62
```

```

function validateName() {
  if (fullName.value.trim() === '') {
    setError(fullName, nameError, 'Full Name is required.');
    return false;
  }
  clearError(fullName, nameError);
  return true;
}

function validateEmail() {
  const emailPattern = /^[\w\.-]+@[^\w\.-]+\.[^\w\.-]+\$/;
  if (email.value.trim() === '') {
    setError(email, emailError, 'Email Address is required.');
    return false;
  }
  if (!emailPattern.test(email.value)) {
    setError(email, emailError, 'Please enter a valid email address.');
    return false;
  }
  clearError(email, emailError);
  return true;
}

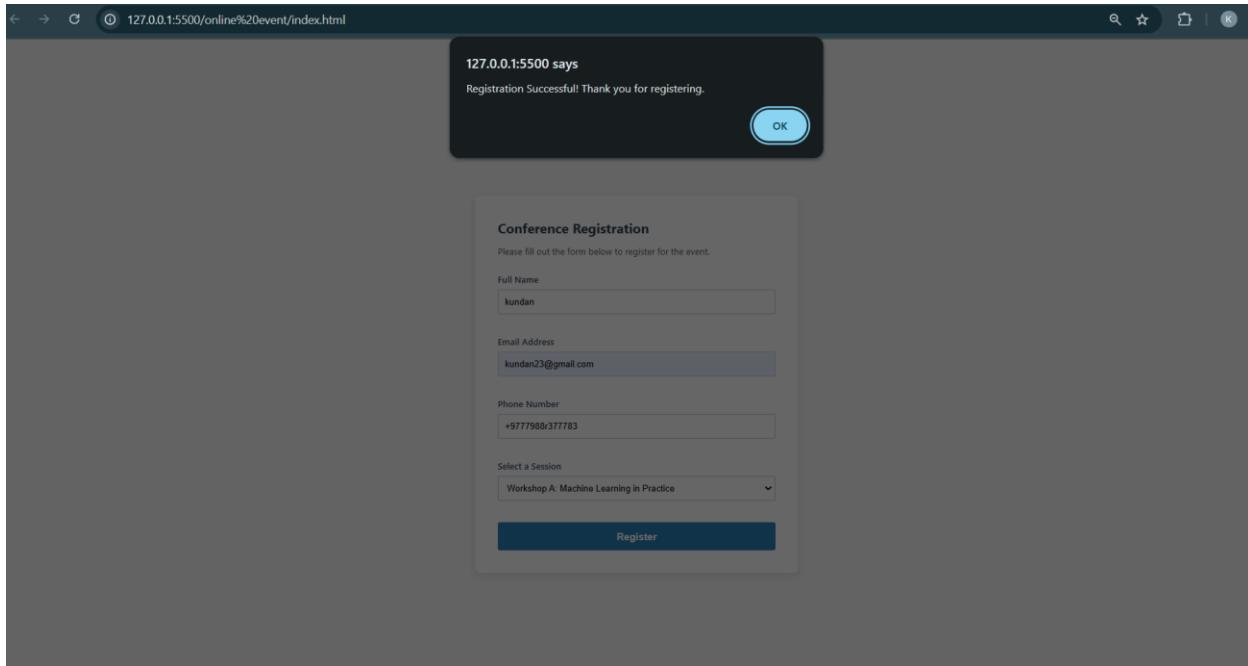
function validatePhone() {
  if (phone.value.trim() === '') {
    setError(phone, phoneError, 'Phone Number is required.');
    return false;
  }
  // Basic phone validation: at least 7 digits
  if (phone.value.replace(/\D/g, '').length < 7) {
    setError(phone, phoneError, 'Please enter a valid phone number.');
    return false;
  }
  clearError(phone, phoneError);
  return true;
}

function validateSession() {
  if (session.value === '') {
    setError(session, sessionError, 'Please select a session.');
    return false;
  }
  clearError(session, sessionError);
  return true;
}

```

Output:

The screenshot shows a registration form titled "Conference Registration" with the instruction "Please fill out the form below to register for the event." Below the title, there are four input fields: "Full Name" (containing "kundan"), "Email Address" (containing "kundan23@gmail.com"), "Phone Number" (containing "+977986377783"), and a dropdown menu for "Select a Session" which has "Workshop B: Advanced JavaScript Techniques" selected. A large blue "Register" button is at the bottom. The "Email Address" field has a red border, indicating it is invalid according to the validation rules shown in the code.



Observation:

- **HTML Structure:** Builds a semantic and accessible registration form with fields for name, email, phone, and a session dropdown menu.
- **CSS Styling:** Applies a modern, professional design with a clean layout, responsive adjustments, and clear focus states for better user experience.
- **JavaScript Validation:** Implements client-side validation to ensure all required fields are filled and the email format is correct before allowing submission.
- **User Feedback:** Provides real-time error messages next to invalid fields and uses CSS classes to visually highlight them, guiding the user to correct their input.
- **Submission Handling:** Prevents the default form submission, validates all fields at once, and displays a success alert message after all data is correctly entered.

Task Description #4 (Data – Fetch API & Render List with Loading/Error States)

- Task: Fetch JSON from an API and render items to the DOM with loading and error UI.
- Instructions:
 - Ask AI to write fetch() logic, create DOM nodes safely, and add skeleton/loading text

Prompt: build a website using html css js that Fetch JSON from an API and render items to the DOM with loading and error UI

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>API Data Fetcher</title>
7      <link rel="stylesheet" href="style.css">
8  </head>
9  <body>
10
11  <main class="container">
12      <h1>User Directory</h1>
13      <p>A list of users fetched from a public API.</p>
14
15      <!-- Loading State: Skeleton UI -->
16      <div id="loading-state" class="loading-container">
17          <div class="skeleton-card">
18              <div class="skeleton-line skeleton-title"></div>
19              <div class="skeleton-line skeleton-text"></div>
20              <div class="skeleton-line skeleton-text"></div>
21          </div>
22          <div class="skeleton-card">
23              <div class="skeleton-line skeleton-title"></div>
24              <div class="skeleton-line skeleton-text"></div>
25              <div class="skeleton-line skeleton-text"></div>
26          </div>
27          <div class="skeleton-card">
28              <div class="skeleton-line skeleton-title"></div>
29              <div class="skeleton-line skeleton-text"></div>
30              <div class="skeleton-line skeleton-text"></div>
31          </div>
32      </div>
33
34      <!-- Error State -->
35      <div id="error-state" class="error-container" style="display: none;" role="alert">
36          <!-- Error message will be injected here by JavaScript -->
37      </div>
38
39      <!-- Success State: User List -->
40      <div id="user-list" class="user-list-container">
41          <!-- User cards will be rendered here by JavaScript -->
42      </div>
43  </main>
44
45  <script src="script.js"></script>
46
47 </body>
</html>
```

Javascript:

```
1 document.addEventListener('DOMContentLoaded', () => {
2     // DOM Elements
3     const loadingState = document.getElementById('loading-state');
4     const errorState = document.getElementById('error-state');
5     const userList = document.getElementById('user-list');
6
7     // API Endpoint
8     const API_URL = 'https://jsonplaceholder.typicode.com/users';
9
10    // Main function to fetch and render users
11    async function fetchAndRenderUsers() {
12        // --- 1. Set Initial State (Show Loading) ---
13        showLoadingState();
14
15        try {
16            // --- 2. Fetch Data ---
17            const response = await fetch(API_URL);
18
19            // Handle HTTP errors (e.g., 404, 500)
20            if (!response.ok) {
21                throw new Error(`HTTP error! Status: ${response.status}`);
22            }
23
24            const users = await response.json();
25
26            // --- 3. Render Data (Success) ---
27            renderUserList(users);
28
29        } catch (error) {
30            // --- 4. Handle Errors ---
31            console.error('Failed to fetch users:', error);
32            showErrorMessage('Failed to load user data. Please try again later.');
33        }
34    }
35
36    function showLoadingState() {
37        loadingState.style.display = 'grid'; // Use grid to match its layout
38        errorState.style.display = 'none';
39        userList.style.display = 'none';
40    }
41
42    function showErrorMessage(message) {
43        loadingState.style.display = 'none';
44        errorState.textContent = message;
45        errorState.style.display = 'block';
46        userList.style.display = 'none';
47    }
48
49    function showSuccessState() {
50        loadingState.style.display = 'none';
51        errorState.style.display = 'none';
52    }
53}
```

```
47 }
48
49 function showSuccessState() {
50   loadingState.style.display = 'none';
51   errorState.style.display = 'none';
52   userList.style.display = 'grid'; // Use grid to match its layout
53 }
54
55 // Safely creates and appends user cards to the DOM
56 function renderUserList(users) {
57   // Clear any previous content
58   userList.innerHTML = '';
59
60   // Loop through each user and create a card
61   users.forEach(user => {
62     // Create elements
63     const card = document.createElement('div');
64     card.className = 'user-card';
65
66     const nameElement = document.createElement('h3');
67     const emailElement = document.createElement('p');
68     const websiteElement = document.createElement('p');
69
70     // --- SAFELY Set Content using .textContent ---
71     // This prevents XSS attacks by not parsing the string as HTML
72     nameElement.textContent = user.name;
73     emailElement.textContent = `Email: ${user.email}`;
74     websiteElement.textContent = `Website: ${user.website}`;
75
76     // Append elements to the card
77     card.appendChild(nameElement);
78     card.appendChild(emailElement);
79     card.appendChild(websiteElement);
80
81     // Append the card to the list container
82     userList.appendChild(card);
83   });
84
85   // Finally, show the populated list
86   showSuccessState();
87 }
88
89 // Initial call to start the process
90 fetchAndRenderUsers();
91 };
```

Output:

The screenshot shows a web browser window with the URL 127.0.0.1:5500/api%20fetch/index.html. The page title is "User Directory". Below it, a sub-header says "A list of users fetched from a public API.". There are nine user profiles displayed in a grid:

Name	Email	Website
Leanne Graham	Email: Leanne@april.biz	Website: leelleighard.org
Ervin Howell	Email: Ervin.Shanan@melissa.ca	Website: anastasia.net
Clementine Bauch	Email: Nathan@yesenia.net	Website: ramiro.info
Patricia Lebsack	Email: Julianne.OConner@kory.org	Website: kale.biz
Chelsey Dietrich	Email: Lucio_Hettinger@annie.ca	Website: demarcus.info
Mrs. Dennis Schulist	Email: Karley_Dach@jasper.info	Website: os.org
Kurtis Weissnat	Email: Telly.Hoeger@billy.biz	Website: elvis.io
Nicholas Runolfsdottir V	Email: SheneWood@rosanond.me	Website: jaynthe.com
Glenna Reichert	Email: Chain.McDermott@dana.io	Website: conrad.com
Clementina DuBuque	Email: Reg.Padberg@karina.biz	Website: ambrose.net

The browser's taskbar at the bottom shows several open tabs, including "GenAI for IT: Transforming IT with AI", "Udemy Course Completion Certificate", "Assignment14.1", "AI-assistant-coding-practical-1", and "API Data Fetcher". The system tray indicates the date as 10/6/2025, the time as 10:10 PM, and the weather as 25°C Mostly cloudy.

Observation:

- **State-Driven UI:** The application's core design is based on managing three states. The HTML provides dedicated containers for each state, and the JavaScript's primary job is to show the appropriate one based on the status of the API request.
- **Asynchronous Data Fetching:** It uses `async/await` with the `fetch()` API, which makes the asynchronous code for fetching data clean and easy to read.
- **Enhanced Loading Experience:** Instead of a simple "Loading..." message, the code implements a "skeleton screen." This provides a better user experience by showing a placeholder that mimics the final layout, reducing perceived load time.
- **Robust Error Handling:** The script correctly handles two types of potential failures:
 1. **Network Errors:** A `try...catch` block catches issues like a lost internet connection.
 2. **HTTP Errors:** It explicitly checks if `response.ok` is true, allowing it to handle server-side errors (like a 404 "Not Found" or 500 "Server Error").
- **Secure DOM Manipulation:** This is a critical security feature. The code avoids using `.innerHTML` to inject the fetched data. Instead, it uses `document.createElement()` and sets the content with `.textContent`. This prevents Cross-Site Scripting (XSS) attacks by ensuring that any malicious code within the API data is rendered as plain text, not executed as HTML.

