

## AI-Assisted Coding

### Lab Assignment -14.1

**Name:** Neha Shazneen

**Roll no:**2403A510A1

**Batch no:**05

**CSE 2<sup>nd</sup> Year**

#### **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 portfolio website with these features:

Sections: **About Me**, **Projects**, and **Contact**

- Use AI to suggest:
  - A modern **color palette** and **typography**
  - A **responsive layout** using **Grid/Flexbox**
  - **Smooth scrolling** navigation between sections

Add simple hover effects, a contact form, and make sure the site looks good on both desktop and mobile.

#### **Code Generated:**

```
① 14.1.task1.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>My Portfolio</title>
7     <style>
8       /* Typography: Using Google Fonts */
9       @import url('https://fonts.googleapis.com/css2?family=Montserrat:wght@400;700&family=Open+Sans:wght@400;600&display=swap');
10
11    :root {
12      /* Color Palette: AI Suggested (modern + professional) */
13      --primary-color: #2C3E50;
14      --accent-color: #E67E22;
15      --bg-color: #F4F4F4;
16      --text-color: #333;
17    }
18
19    * {
20      margin: 0;
21      padding: 0;
22      box-sizing: border-box;
23      scroll-behavior: smooth;
24    }
25
26
27  e\Desktop\AI Assisted Coding\task2.py
28  body {
29    font-family: 'Open Sans', sans-serif;
30    background-color: var(--bg-color);
31    color: var(--text-color);
32    line-height: 1.6;
33  }
34
35  header {
36    background: var(--primary-color);
37    color: white;
38    padding: 1rem 2rem;
39    position: fixed;
40    top: 0;
41    width: 100%;
42    z-index: 1000;
43  }
44
45  nav {
46    display: flex;
47    justify-content: space-between;
48    align-items: center;
49  }
50
51  nav a {
52    color: white;
53    margin-left: 1.5rem;
54    text-decoration: none;
55    font-weight: 600;
56  }
57
58  nav a:hover {
59    color: var(--accent-color);
60  }
61
62  section {
63    padding: 100px 20px 60px;
64    max-width: 1000px;
65    margin: auto;
66  }
67
68  h2 {
69    font-family: 'Montserrat', sans-serif;
70    font-size: 2rem;
71    margin-bottom: 1rem;
72  }
```

```
70
71     .about,
72     .projects,
73     .contact {
74         display: grid;
75         gap: 1rem;
76     }
77
78
79     .projects .project {
80         background: white;
81         padding: 1rem;
82         border: 1px solid #ddd;
83         border-radius: 5px;
84         transition: transform 0.3s ease;
85     }
86
87     .projects .project:hover {
88         transform: scale(1.02);
89     }
90
91     form {
92         display: flex;
93         flex-direction: column;
94         gap: 1rem;
95     }
96
97     input, textarea {
98         padding: 0.75rem;
99         border: 1px solid #ccc;
100        border-radius: 4px;
101        font-size: 1rem;
102    }
103
```

```
104     input[type="submit"] {
105         background-color: var(--accent-color);
106         color: white;
107         border: none;
108         cursor: pointer;
109     }
110
111     input[type="submit"]:hover {
112         background-color: #cf6e17;
113     }
114
115     /* Responsive Layout */
116     @media (min-width: 768px) {
117         .projects {
118             grid-template-columns: repeat(2, 1fr);
119         }
120     }
121
122     @media (min-width: 1024px) {
123         .projects {
124             grid-template-columns: repeat(3, 1fr);
125         }
126     }
127
128     /* Space at top to account for fixed header */
129     .spacer {
130         height: 80px;
131     }
132     </style>
133 </head>
134 <body>
```

```

135
136 <header>
137   <nav>
138     <div><strong>My Portfolio</strong></div>
139     <div>
140       <a href="#about">About Me</a>
141       <a href="#projects">Projects</a>
142       <a href="#contact">Contact</a>
143     </div>
144   </nav>
145 </header>
146
147 <div class="spacer"></div>
148
149 <section id="about" class="about">
150   <h2>About Me</h2>
151   <p>Hello! I'm a passionate developer with experience in web design and development. I love turning ideas into real, working applications.</p>
152 </section>
153
154 <section id="projects" class="projects">
155   <h2>Projects</h2>
156   <div class="project">
157     <h3>Project One</h3>
158     <p>A web app built using HTML, CSS, and JavaScript.</p>
159   </div>
160   <div class="project">
161     <h3>Project Two</h3>
162     <p>An eCommerce site created with React and Firebase.</p>
163   </div>
164   <!-- Add more projects here -->
165 </section>
166
167 <section id="contact" class="contact">
168   <h2>Contact</h2>
169   <form>
170     <input type="text" placeholder="Your Name" required />
171     <input type="email" placeholder="Your Email" required />
172     <textarea rows="5" placeholder="Your Message" required></textarea>
173     <input type="submit" value="Send Message" />
174   </form>
175 </section>
176
177 </body>
178 </html>
179

```

## Output:

## About Me

Hello! I'm a passionate developer with experience in web design and development. I love turning ideas into real, working applications.

## Projects

### Project One

A web app built using HTML, CSS, and JavaScript.

### Project Two

An eCommerce site created with React and Firebase.

## Contact

Your Name

Your Email

Your Message

Send Message

### Observation:

#### 1. Section Creation

- **About Me, Projects, and Contact** sections are clearly defined using `<section>` elements with IDs for navigation.

#### 2. AI-Enhanced Design Suggestions

- **Color Palette:** A clean, professional palette with a dark blue (#2C3E50), accent orange (#E67E22), and a light gray background (#F4F4F4)—suggested for modern portfolios.
- **Typography:** Google Fonts Montserrat (for headings) and Open Sans (for body text) are used for clean, readable design.

#### 3. Responsive Layout

- **CSS Grid** is used for the .projects section.
- Media queries adjust the grid layout for tablets and desktops.
- The layout adapts to different screen sizes, maintaining readability and structure.

#### 4. Smooth Scrolling Navigation

- `scroll-behavior: smooth` is applied globally using the `* selector`.
- The nav links use anchor tags (`href="#section-id"`) to allow smooth transitions when clicked.

## 5. Flexbox for Navigation

- Flexbox is used in the nav element to align the brand title and navigation links horizontally and responsively.

### 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

Design a responsive product display page for an online store.

Requirements:

- Show product image, title, price, and an "Add to Cart" button.
- Use AI to:
  - Apply BEM methodology for CSS class naming.
  - Make layout responsive with Flexbox/Grid.
  - Add hover effects and show an alert when "Add to Cart" is clicked.

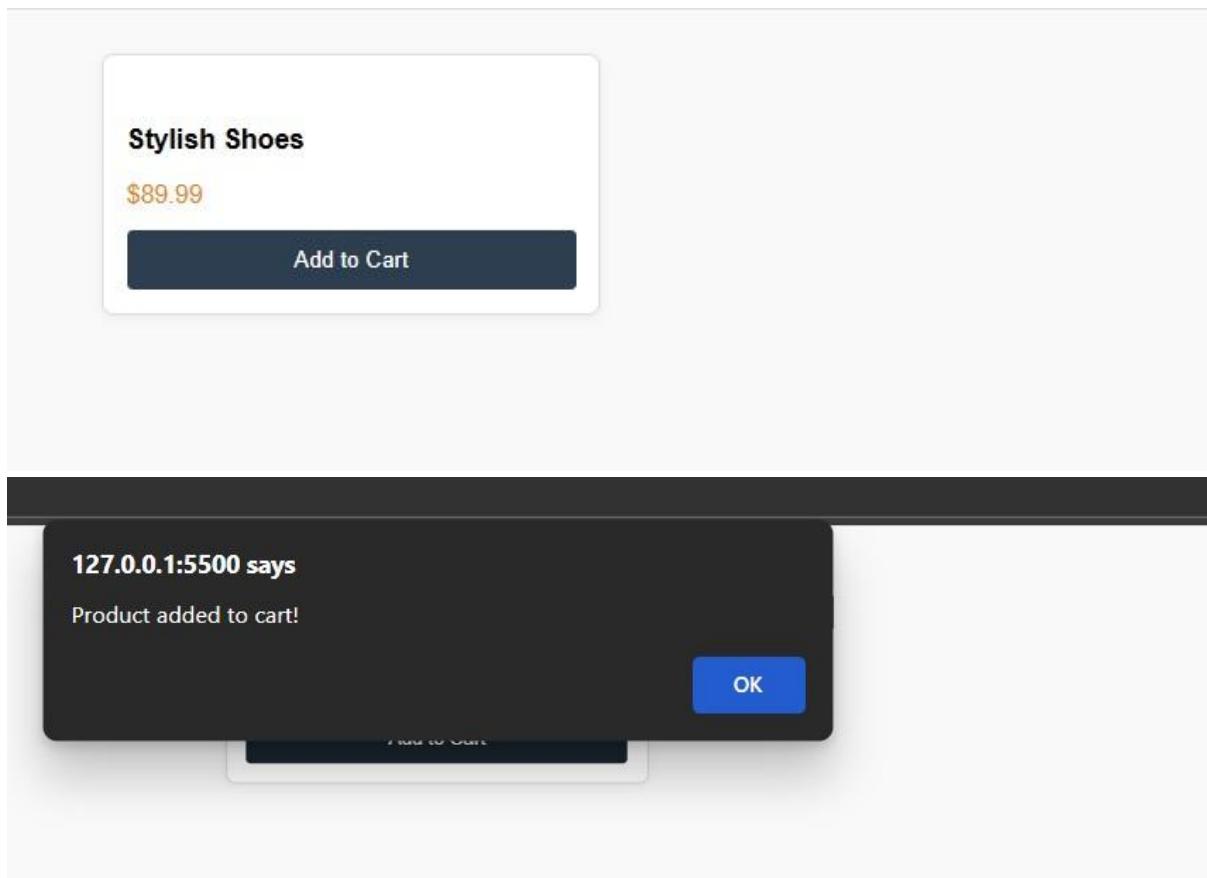
#### Code Generated:

```
④ 14.1.task2.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>Product Page</title>
7  <style>
8  /* BEM-style CSS and AI-Suggested Styling */
9  body {
10 |  font-family: Arial, sans-serif;
11 |  margin: 0;
12 |  padding: 0;
13 |  background-color: #f9f9f9;
14 }
15
16 .product-page {
17 |  display: flex;
18 |  flex-direction: column;
19 |  align-items: center;
20 |  padding: 2rem;
21 }
22
23 .product-card {
24 |  background-color: #fff;
25 |  border: 1px solid #ddd;
26 |  border-radius: 8px;
27 |  max-width: 350px;
28 |  width: 100%;
29 |  box-shadow: 0 2px 8px rgba(0, 0, 0, 0.05);
30 |  overflow: hidden;
31 |  transition: transform 0.3s ease;
32 }
33
34 .product-card:hover {
35 |  transform: scale(1.02);
36 }
37
```

```
37
38     .product-card__image {
39         width: 100%;
40         height: auto;
41     }
42
43     .product-card__content {
44         padding: 1rem;
45     }
46
47     .product-card__title {
48         font-size: 1.2rem;
49         font-weight: bold;
50         margin-bottom: 0.5rem;
51     }
52
53     .product-card__price {
54         color: #e67e22;
55         font-size: 1.1rem;
56         margin-bottom: 1rem;
57     }
58
59     .product-card__button {
60         background-color: #2c3e50;
61         color: #fff;
62         padding: 0.75rem 1rem;
63         border: none;
64         border-radius: 4px;
65         cursor: pointer;
66         font-size: 1rem;
67         width: 100%;
68         transition: background-color 0.3s;
69     }
70
```

```
71  .product-card__button:hover {  
72    background-color: #1a252f;  
73  }  
74  
75  /* Responsive */  
76  @media (min-width: 768px) {  
77    .product-page {  
78      flex-direction: row;  
79      justify-content: center;  
80    }  
81  }  
82  </style>  
83  </head>  
84  <body>  
85  
86  <main class="product-page">  
87    <div class="product-card">  
88        
89      <div class="product-card__content">  
90        <h2 class="product-card__title">Stylish Shoes</h2>  
91        <p class="product-card__price">$89.99</p>  
92        <button class="product-card__button" onclick="addToCart()">Add to Cart</button>  
93      </div>  
94    </div>  
95  </main>  
96  
97  <script>  
98    function addToCart() {  
99      alert("Product added to cart!");  
100    }  
101  </script>  
  
102  
103  </body>  
104  </html>  
105
```

Output:



### Observation:

#### Required Elements Present

- Shows product **image**, **title**, **price**, and **Add to Cart** button.

#### 2. BEM Methodology Used

- Class names follow BEM (Block\_\_Element format), e.g., product-card\_\_title, product-card\_\_button.

#### 3. Responsive Layout

- Mobile-first design with Flexbox.
- Switches to horizontal layout on wider screens (@media breakpoint at 768px).

#### 4. Interactivity

- Hover effect on the card (transform: scale(1.02)).
- Button hover changes background.
- Clicking "**Add to Cart**" triggers a simple JavaScript alert.

### 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:

o Add form validation with JavaScript.

o Make the form accessible with labels and ARIA.

o Style with a professional look. **Prompt:**

Create an **event registration form** for a conference.

Requirements:

- Collect: **Name, Email, Phone Number**, and **Session Selection**
- Use AI to:
  - Add **JavaScript form validation**
  - Ensure **accessibility** with proper labels and ARIA attributes
  - Apply **professional CSS styling**

### **Code Generated:**

```
> 14.1.task3.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     <style>
8       body {
9         font-family: 'Segoe UI', sans-serif;
10        background-color: #f4f4f4;
11        padding: 2rem;
12      }
13
14      .form-container {
15        background: #fff;
16        max-width: 500px;
17        margin: auto;
18        padding: 2rem;
19        border-radius: 8px;
20        box-shadow: 0 4px 12px rgba(0,0,0,0.1);
21      }
22
23      .form-container h2 {
24        text-align: center;
25        margin-bottom: 1.5rem;
26      }
27
28      .form-group {
29        margin-bottom: 1rem;
30      }
31
32      label {
33        display: block;
34        margin-bottom: 0.5rem;
35        font-weight: bold;
36      }
37
38      input, select {
39        width: 100%;
```

```
40     padding: 0.75rem;
41     border: 1px solid #ccc;
42     border-radius: 4px;
43     font-size: 1rem;
44   }
45
46   input:focus, select:focus {
47     border-color: #0077cc;
48     outline: none;
49   }
50
51   .error-message {
52     color: red;
53     font-size: 0.9rem;
54     display: none;
55   }
56
57   button {
58     width: 100%;
59     padding: 0.75rem;
60     background-color: #0077cc;
61     color: white;
62     font-size: 1rem;
63     border: none;
64     border-radius: 4px;
65     cursor: pointer;
66   }
67
68   button:hover {
69     background-color: #005fa3;
70   }
71 </style>
72 </head>
73 <body>
74
75 <div class="form-container" role="form" aria-labelledby="form-title">
76   <h2 id="form-title">Event Registration Form</h2>
77   <form id="registrationForm" novalidate>
78     <div class="form-group">
79       <label for="name">Full Name *</label>
80       <input type="text" id="name" name="name" aria-required="true" required />
81       <div class="error-message" id="nameError">Please enter your name.</div>
82     </div>
83
84     <div class="form-group">
85       <label for="email">Email Address *</label>
86       <input type="email" id="email" name="email" aria-required="true" required />
87       <div class="error-message" id="emailError">Please enter a valid email.</div>
88     </div>
89
90     <div class="form-group">
91       <label for="phone">Phone Number *</label>
92       <input type="tel" id="phone" name="phone" pattern="[0-9]{10}" aria-required="true" required />
93       <div class="error-message" id="phoneError">Enter a 10-digit phone number.</div>
94     </div>
95
96     <div class="form-group">
97       <label for="session">Select a Session *</label>
98       <select id="session" name="session" aria-required="true" required>
99         <option value="">-- Select a Session --</option>
100        <option value="session1">Session 1: Web Development</option>
101        <option value="session2">Session 2: AI & ML</option>
102        <option value="session3">Session 3: Cybersecurity</option>
103      </select>
104      <div class="error-message" id="sessionError">Please select a session.</div>
105    </div>
106
107    <button type="submit">Register</button>
108  </form>
109 </div>
```

```
109 </div>
110 <script>
111   const form = document.getElementById('registrationForm');
112
113   Complexity is 6 It's time to do something...
114   form.addEventListener('submit', function (e) { █
115     e.preventDefault();
116     let valid = true;
117
118     // Name validation
119     const name = document.getElementById('name');
120     const nameError = document.getElementById('nameError');
121     if (name.value.trim() === '') {
122       nameError.style.display = 'block';
123       valid = false;
124     } else {
125       nameError.style.display = 'none';
126     }
127
128     // Email validation
129     const email = document.getElementById('email');
130     const emailError = document.getElementById('emailError');
131     const emailRegex = /^[^@\s]+@[^\s@]+\.[^\s@]+$/;
132     if (!emailRegex.test(email.value)) {
133       emailError.style.display = 'block';
134       valid = false;
135     } else {
136       emailError.style.display = 'none';
137     }
138
139     // Phone validation
140     const phone = document.getElementById('phone');
141     const phoneError = document.getElementById('phoneError');
142     const phoneRegex = /^\d{10}$/;
143     if (!phoneRegex.test(phone.value)) {
144       phoneError.style.display = 'block';
145
146       valid = false;
147     } else {
148       phoneError.style.display = 'none';
149     }
150
151     // Session validation
152     const sessionError = document.getElementById('sessionError');
153     if (session.value === '') {
154       sessionError.style.display = 'block';
155       valid = false;
156     } else {
157       sessionError.style.display = 'none';
158     }
159
160     // Final submission
161     if (valid) {
162       alert("Registration successful!");
163       form.reset();
164     }
165   });
166 </script>
167 </body>
168 </html>
169
```

Output:

## Event Registration Form

**Full Name \***

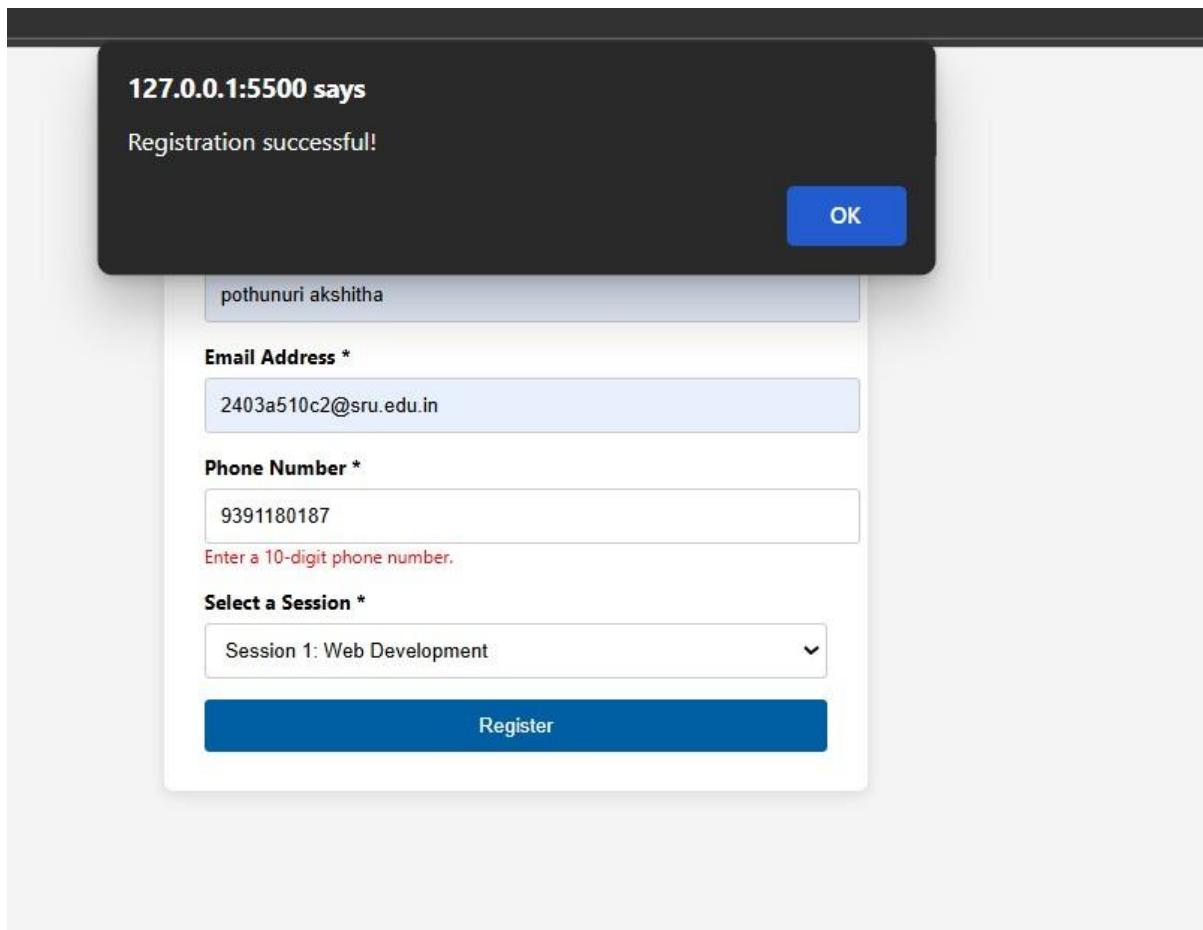
**Email Address \***

**Phone Number \***

**Select a Session \***

-- Select a Session --

**Register**



### Observation:

#### 1. Form Requirements Met

- Captures: **Name, Email, Phone, Session Selection** using input and select fields.
- Each input has a <label> for accessibility and ARIA roles where applicable.

#### 2. Form Validation with JavaScript

- Validates all fields:
  - Name: not empty
  - Email: valid format
  - Phone: 10-digit number
  - Session: option selected
- Shows error messages for invalid input.
- Displays alert("Registration successful!") on valid submit.

#### 3. Accessibility Features

- Labels are correctly linked to inputs via for and id.
- ARIA attributes like aria-required and aria-labelledby used.
- role="form" used for screen readers.

#### 4. Professional Styling

- Clean, modern UI with spacing, borders, shadows, and focus states.
- Uses responsive, accessible input styles and button hover effect.

**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: o Ask AI to write fetch() logic, create DOM nodes safely, and add skeleton/loading text.

### Prompt

Write code to **fetch JSON data** from an API and **render a list of items** to the DOM.

Use AI to:

- Write clean fetch() logic with proper **error handling**
- Create **DOM elements safely** • Show **loading text or skeleton UI** while data is loading
- Show an **error message** if fetch fails

### Code Generated:

```

14.1.task4.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>Fetch API List</title>
7       <style>
8         body {
9           font-family: Arial, sans-serif;
10          padding: 2rem;
11          background-color: #f9f9f9;
12        }
13
14        .container {
15          max-width: 600px;
16          margin: auto;
17        }
18
19        .item {
20          background-color: #fff;
21          padding: 1rem;
22          border: 1px solid #ddd;
23          border-radius: 4px;
24          margin-bottom: 1rem;
25          box-shadow: 0 2px 4px rgba(0,0,0,0.05);
26        }
27
28        .loading, .error {
29          text-align: center;
30          font-size: 1.2rem;
31          color: #555;
32          margin-top: 2rem;
33        }
34      </style>
35    </head>
36    <body>
37
38      <div class="container" id="data-container">
```

```
38  <div class="container" id="data-container">
39    <div class="loading" id="loading">Loading items...</div>
40  </div>
41
42  <script>
43    const container = document.getElementById('data-container');
44    const loading = document.getElementById('loading');
45
46    // Sample API for demo (can be replaced with any API)
47    const API_URL = 'https://jsonplaceholder.typicode.com/posts?\_limit=5';
48
49    function createElement(item) {
50      const div = document.createElement('div');
51      div.className = 'item';
52
53      const title = document.createElement('h3');
54      title.textContent = item.title;
55
56      const body = document.createElement('p');
57      body.textContent = item.body;
58
59      div.appendChild(title);
60      div.appendChild(body);
61
62      return div;
63    }
64
65  Complexity is 6 It's time to do something...
66  async function fetchData() { █
67    try {
68      // Show loading
69      loading.style.display = 'block';
70
71      const response = await fetch(API_URL);
72
73      if (!response.ok) {
74        if (!response.ok) {
75          throw new Error(`HTTP error! Status: ${response.status}`);
76        }
77
78        const data = await response.json();
79
80        // Remove loading
81        loading.style.display = 'none';
82
83        // Render items
84        data.forEach(item => {
85          const element = createElement(item);
86          container.appendChild(element);
87        });
88
89      } catch (error) {
90        loading.style.display = 'none';
91        const errorMessage = document.createElement('div');
92        errorMessage.className = 'error';
93        errorMessage.textContent = 'Failed to load data. Please try again later.';
94        container.appendChild(errorMessage);
95        console.error('Fetch error:', error);
96      }
97
98      fetchData();
99    } </script>
100 </body>
101 </html>
102
```

Output:

**sunt aut facere repellat provident occaecati excepturi optio  
reprehenderit**

quia et suscipit suscipit recusandae consequuntur expedita et cum  
reprehenderit molestiae ut ut quas totam nostrum rerum est autem sunt rem  
eveniet architecto

**qui est esse**

est rerum tempore vitae sequi sint nihil reprehenderit dolor beatae ea dolores  
neque fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis qui  
aperiam non debitis possimus qui neque nisi nulla

**ea molestias quasi exercitationem repellat qui ipsa sit aut**

et iusto sed quo iure voluptatem occaecati omnis eligendi aut ad voluptatem  
doloribus vel accusantium quis pariatur molestiae porro eius odio et labore et  
velit aut

**eum et est occaecati**

ullam et saepe reiciendis voluptatem adipisci sit amet autem assumenda  
provident rerum culpa quis hic commodi nesciunt rem tenetur doloremque  
ipsam iure quis sunt voluptatem rerum illo velit

**nesciunt quas odio**

repudiandae veniam quaerat sunt sed alias aut fugiat sit autem sed est  
voluptatem omnis possimus esse voluptatibus quis est aut tenetur dolor neque

Observation:

1. Fetch JSON Data

- Uses `fetch()` to get sample JSON data from [jsonplaceholder.typicode.com](https://jsonplaceholder.typicode.com) (a placeholder public API).
- You can replace the URL with any other API that returns JSON.

2. Loading UI

- Shows "Loading items..." text while the data is being fetched.
- Hidden after data is successfully loaded or an error occurs.

### **3. Safe DOM Creation**

- All DOM nodes (title, paragraph) are created using document.createElement, ensuring security (avoids innerHTML).
- Elements are appended cleanly into a container.

### **4. Error Handling**

- Uses try...catch block to catch fetch or parsing errors.
- Displays a user-friendly error message in the DOM if the fetch fails.

### **5. UI Styling**

- Clean, card-style layout for items.
- Text-centered loading and error messages with minimal styling.