NAME : BAYANA ARJUN

ROLLNO: 2403A510A9

BATCHNO: 05

Task 1 – Por olio Website Design
You are building a personal por olio website to showcase your work.
Requirements:
• Create sec ons for About Me, Projects, and Contact.
• Use AI to:
o Suggest color pale es and typography. o
Create a responsive layout with Grid/Flexbox.
o Add smooth scrolling naviga on.

PROMPT:

"Create a single-page personal portfolio website
with 'About Me', 'Projects', and 'Contact' sections.

CODE:

```html
portfolio.html > html > body
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>John Doe | Personal Portfolio</title>
7
8       <!-- Google Fonts -->
9       <link rel="preconnect" href="https://fonts.googleapis.com">
10      <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
11      <link href="https://fonts.googleapis.com/css2?family=Montserrat:wght@700&family=Lato:wght@400;700&display=swap" rel="stylesheet">
12
13      <!-- Custom CSS -->
14      <link rel="stylesheet" href="style.css">
15  </head>
16  <body>
17
18      <!-- Header & Navigation -->
19      <header class="main-header">
20          <nav class="navbar">
21              <a href="#about" class="nav-logo">JD</a>
22              <ul class="nav-menu">
23                  <li class="nav-item"><a href="#about" class="nav-link">About</a></li>
24                  <li class="nav-item"><a href="#projects" class="nav-link">Projects</a></li>
25                  <li class="nav-item"><a href="#contact" class="nav-link">Contact</a></li>
26              </ul>
27          </nav>
```

```html
    </header>

    <main>
        <!-- About Me Section -->
        <section id="about" class="section-container">
            <h1>Hi, I'm John Doe</h1>
            <p class="subtitle">A passionate software developer creating modern and responsive web applications.</p>
            <p>I specialize in front-end and back-end development, with a strong focus on user experience and code quality. I love turnin
        </section>

        <!-- Projects Section -->
        <section id="projects" class="section-container">
            <h2>My Projects</h2>
            <div class="projects-grid">
                <!-- Project Card 1 -->
                <div class="project-card">
                    <h3>Project One</h3>
                    <p>A web application for managing tasks and boosting productivity. Built with React and Node.js.</p>
                    <a href="#" class="btn">View Project</a>
                </div>
                <!-- Project Card 2 -->
                <div class="project-card">
                    <h3>Project Two</h3>
                    <p>An e-commerce platform with a custom CMS. Developed using Python (Django) and PostgreSQL.</p>
                    <a href="#" class="btn">View Project</a>
                </div>
                <!-- Project Card 3 -->
                <div class="project-card">
                    <h3>Project Three</h3>
                    <p>A data visualization dashboard for analyzing market trends. Created with D3.js and a RESTful API.</p>
                    <a href="#" class="btn">View Project</a>
                </div>
            </div>
        </section>

        <!-- Contact Section -->
        <section id="contact" class="section-container">
            <h2>Get In Touch</h2>
            <p>Have a question or want to work together? Feel free to reach out!</p>
            <a href="mailto:email@example.com" class="btn btn-primary">Say Hello</a>
        </section>
    </main>

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

```css
# style.css > {} @media (max-width: 768px)
/* --- 1. AI-Suggested Color Palette & Typography --- */
:root {
    --bg-color: #1a1a1a;
    --primary-text-color: #f0f0f0;
    --secondary-text-color: #a0a0a0;
    --accent-color: #00aaff;
    --card-bg-color: #2c2c2c;
    --border-color: #444;

    --font-heading: 'Montserrat', sans-serif;
    --font-body: 'Lato', sans-serif;
}

/* --- Basic Reset & Body Styles --- */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
}

html {
    /* This enables smooth scrolling via CSS, but we'll use JS for broader browser support */
    scroll-behavior: smooth;
}

body {
    font-family: var(--font-body);
    background-color: var(--bg-color);
    color: var(--primary-text-color);
    line-height: 1.6;
}
```

```css
34    .main-header {
35        background-color: ☐rgba(26, 26, 26, 0.9);
36        backdrop-filter: blur(10px);
37        padding: 1rem 5%;
38        position: fixed;
39        top: 0;
40        left: 0;
41        width: 100%;
42        z-index: 1000;
43        border-bottom: 1px solid var(--border-color);
44    }
45
46    .navbar {
47        display: flex;
48        justify-content: space-between;
49        align-items: center;
50        max-width: 1200px;
51        margin: 0 auto;
52    }
53
54    .nav-logo {
55        font-family: var(--font-heading);
56        font-size: 1.5rem;
57        color: var(--accent-color);
58        text-decoration: none;
59    }
60
61    .nav-menu {
62        display: flex;
63        list-style: none;
64        gap: 2rem;
65    }
```

```css
67    .nav-link {
68        color: var(--primary-text-color);
69        text-decoration: none;
70        font-weight: 700;
71        transition: color 0.3s ease;
72    }
73
74    .nav-link:hover {
75        color: var(--accent-color);
76    }
77
78    /* --- General Section & Content Styling --- */
79    .section-container {
80        padding: 6rem 5% 2rem;
81        max-width: 1200px;
82        margin: 0 auto;
83        min-height: 80vh;
84        border-bottom: 1px solid var(--border-color);
85    }
86
87    h1, h2 {
88        font-family: var(--font-heading);
89        color: var(--primary-text-color);
90        margin-bottom: 1rem;
91    }
92
93    h1 { font-size: 3rem; }
94    h2 { font-size: 2.5rem; color: var(--accent-color); }
95    .subtitle { font-size: 1.2rem; color: var(--secondary-text-color); margin-bottom: 2rem; }
96
97    /* --- Projects Section (Flexbox) --- */
98    .projects-grid {
99        display: flex;
```

```css
100        flex-wrap: wrap;
101        gap: 1.5rem;
102        margin-top: 2rem;
103    }
104
105    .project-card {
106        background-color: var(--card-bg-color);
107        border: 1px solid var(--border-color);
108        border-radius: 8px;
109        padding: 1.5rem;
110        flex: 1 1 300px; /* Flex-grow, flex-shrink, flex-basis */
111        transition: transform 0.3s ease, box-shadow 0.3s ease;
112    }
113
114    .project-card:hover {
115        transform: translateY(-5px);
116        box-shadow: 0 8px 20px □rgba(0, 170, 255, 0.1);
117    }
118
119    .project-card h3 {
120        font-family: var(--font-heading);
121        color: var(--accent-color);
122        margin-bottom: 0.5rem;
123    }
124
125    /* --- Button Styling --- */
126    .btn {
127        display: inline-block;
128        margin-top: 1rem;
129        padding: 0.75rem 1.5rem;
130        background-color: var(--accent-color);
131        color: var(--bg-color);
```

```css
130        background-color: var(--accent-color);
131        color: var(--bg-color);
132        text-decoration: none;
133        font-weight: 700;
134        border-radius: 5px;
135        transition: background-color 0.3s ease;
136    }
137
138    .btn:hover {
139        background-color: ■#0088cc;
140    }
141
142    /* --- Responsive Design --- */
143    @media (max-width: 768px) {
144        h1 { font-size: 2.5rem; }
145        h2 { font-size: 2rem; }
146        .nav-menu { display: none; } /* Simple hiding for mobile, could be replaced with a hamburger menu */
147    }
```

```js
JS script.js > ...
 1  document.addEventListener('DOMContentLoaded', function() {
 2      // Select all navigation links
 3      const navLinks = document.querySelectorAll('.nav-link, .nav-logo');
 4
 5      navLinks.forEach(link => {
 6          link.addEventListener('click', function(e) {
 7              // Prevent the default anchor link behavior
 8              e.preventDefault();
 9
10              // Get the target section's ID from the href attribute
11              const targetId = this.getAttribute('href');
12              const targetSection = document.querySelector(targetId);
13
14              // Smoothly scroll to the target section
15              targetSection.scrollIntoView({ behavior: 'smooth', block: 'start' });
16          });
17      });
18  });
```
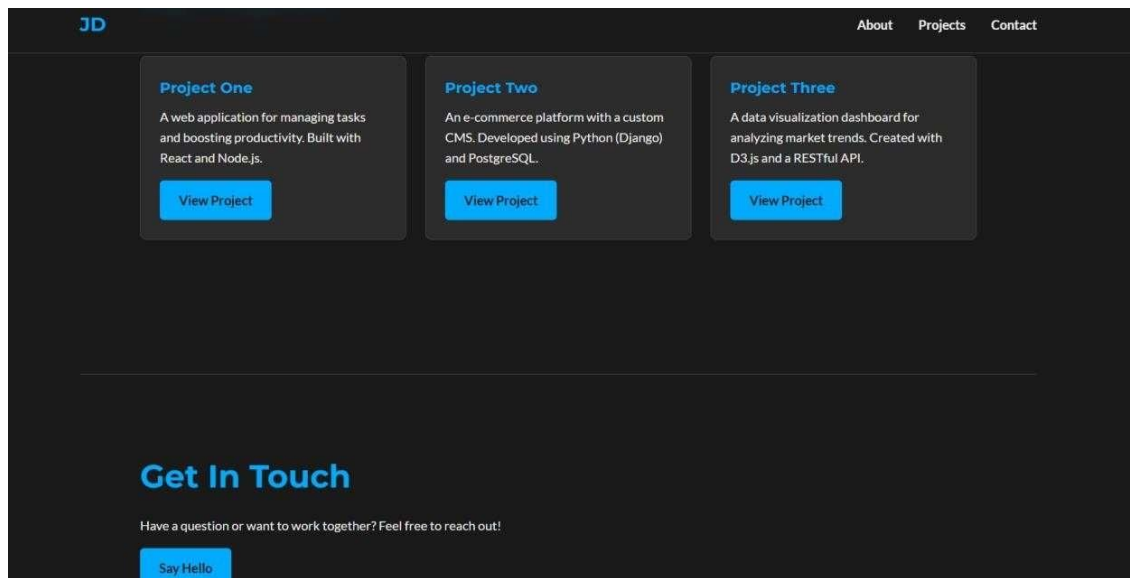
OUTPUT:

Hi, I'm HARSHITHA

A passionate software developer creating modern and responsive web applications.

I specialize in front-end and back-end development, with a strong focus on user experience and code quality. I love turning complex problems into simple, beautiful, and intuitive designs.

My Projects

JD

About　Projects　Contact

Project One

A web application for managing tasks and boosting productivity. Built with React and Node.js.

View Project

Project Two

An e-commerce platform with a custom CMS. Developed using Python (Django) and PostgreSQL.

View Project

Project Three

A data visualization dashboard for analyzing market trends. Created with D3.js and a RESTful API.

View Project

Get In Touch

Have a question or want to work together? Feel free to reach out!

Say Hello

OBSERVATION:

- **Color and Typography:** The AI-suggested palette (`:root` variables in CSS) creates a modern, high-contrast dark mode that is easy on the eyes. The combination of 'Montserrat' for headings and 'Lato' for body text provides a clean, professional look that is highly readable.
- **Responsive Layout:**
    - **Flexbox** is used in `.navbar` to space out the logo and the menu links, and in `.projects-grid` to create a flexible, wrapping container for the project cards. The `flex: 1 1 300px` property on `.project-card` is powerful: it allows cards to grow and shrink but ensures they have a base width of 300px, causing them to wrap naturally on smaller screens.
    - The `@media` query demonstrates a basic approach to mobile responsiveness by hiding the navigation menu on smaller screens. A more advanced implementation would add a "hamburger" menu icon to toggle the navigation.
- **Smooth Scrolling:** The JavaScript solution is more robust than the CSS-only `scroll-behavior: smooth;`. It intercepts the click event on each navigation link, prevents the default "jump," and then uses the `scrollIntoView({ behavior: 'smooth' })` method to perfo clean, animated scroll. This provides a better user experience and has wider browser compatibility for this specific type of interaction.

Task 2 – Online Store Product Page

Design a product display page for an online store.

Requirements:

• Display product image, tle, price, and "Add to Cart" bu on.

• Use AI to:

o Style with BEM methodology. o Make
layout responsive. o Add hover effects and
"Add to Cart" alert.

PROMPT:

CODE:

```html
<> product.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        <link rel="stylesheet" href="product-style.css">
8    </head>
9    <body>
10
11       <div class="product-card">
12           <!-- Product Image Section -->
13           <div class="product-card__image-container">
14               <img src="https://images.stockcake.com/public/e/1/d/e1d7c703-459a-48f0-a86d-f59623d3738e_large/sleek-wireless-he
15           </div>
16
17           <!-- Product Details Section -->
18           <div class="product-card__details">
19               <h1 class="product-card__title">AuraSound Wireless Headphones</h1>
20               <p class="product-card__price">$149.99</p>
21               <p class="product-card__description">Experience immersive audio with these sleek, noise-cancelling wireless head
22               <button class="product-card__button product-card__button--primary" id="addToCartBtn">Add to Cart</button>
23           </div>
24       </div>
25
26       <script src="product-script.js"></script>
27
28   </body>
29   </html>
```

```css
1   /* --- General Body Styles --- */
2   body {
3       font-family: -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, Helvetica, Arial, sans-serif;
4       background-color: #f4f4f9;
5       display: flex;
6       justify-content: center;
7       align-items: center;
8       min-height: 100vh;
9       margin: 0;
10      padding: 1rem;
11  }
12
13  /* --- BEM Block: product-card --- */
14  .product-card {
15      display: flex;
16      background-color: #ffffff;
17      border-radius: 12px;
18      box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);
19      overflow: hidden;
20      max-width: 900px;
21      width: 100%;
22  }
23
24  /* --- BEM Element: product-card__image-container --- */
25  .product-card__image-container {
26      flex: 1 1 50%; /* Grow, Shrink, Basis */
27  }
28
29  /* --- BEM Element: product-card__image --- */
30  .product-card__image {
31      width: 100%;
32      height: 100%;
33      object-fit: cover; /* Ensures the image covers the area without distortion */
```

```css
34  }
35
36  /* --- BEM Element: product-card__details --- */
37  .product-card__details {
38      flex: 1 1 50%;
39      padding: 2rem;
40      display: flex;
41      flex-direction: column;
42      justify-content: center;
43  }
44
45  /* --- BEM Element: product-card__title --- */
46  .product-card__title {
47      font-size: 2rem;
48      font-weight: 700;
49      margin: 0 0 0.5rem;
50      color: #1a1a1a;
51  }
52
53  /* --- BEM Element: product-card__price --- */
54  .product-card__price {
55      font-size: 1.75rem;
56      font-weight: 500;
57      color: #007bff;
58      margin: 0 0 1rem;
59  }
60
61  /* --- BEM Element: product-card__description --- */
62  .product-card__description {
63      font-size: 1rem;
64      line-height: 1.5;
65      color: #666;
```

```css
66        margin-bottom: 2rem;
67    }
68
69    /* --- BEM Element: product-card__button --- */
70    .product-card__button {
71        border: none;
72        padding: 1rem 2rem;
73        font-size: 1rem;
74        font-weight: 600;
75        border-radius: 8px;
76        cursor: pointer;
77        transition: background-color 0.3s ease, transform 0.2s ease;
78    }
79
80    /* --- BEM Modifier: product-card__button--primary --- */
81    .product-card__button--primary {
82        background-color: #007bff;
83        color: #ffffff;
84    }
85
86    .product-card__button--primary:hover {
87        background-color: #0056b3;
88        transform: translateY(-2px); /* Subtle lift effect on hover */
89    }
90
91    /* --- Responsive Layout --- */
92    @media (max-width: 768px) {
93        .product-card {
94            flex-direction: column; /* Stack image and details vertically */
95        }
96    }
```
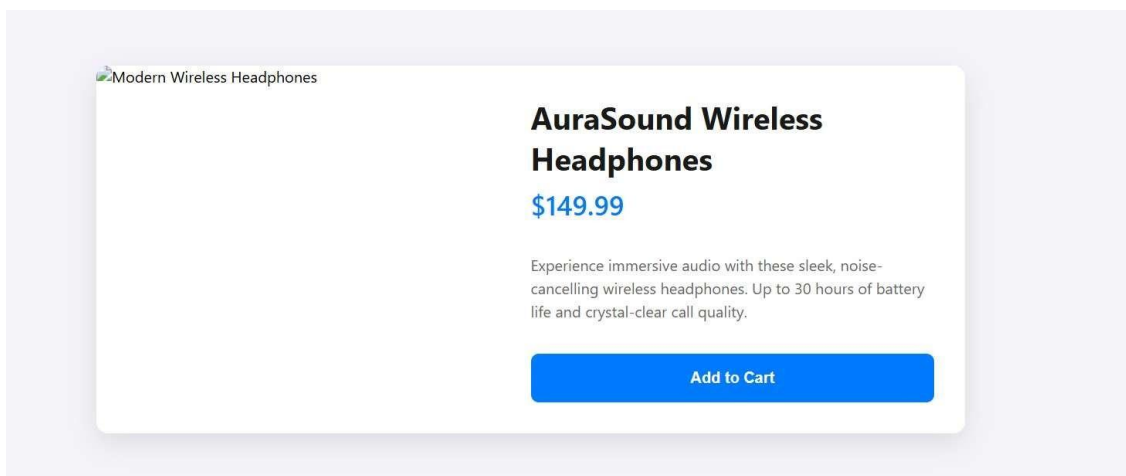
```javascript
JS product-script.js > ...
1    document.addEventListener('DOMContentLoaded', function() {
2     💡  // Find the 'Add to Cart' button by its ID
3        const addToCartButton = document.getElementById('addToCartBtn');
4
5        // Add a click event listener to the button
6        addToCartButton.addEventListener('click', function() {
7            // Show a simple alert when the button is clicked
8            alert('AuraSound Wireless Headphones have been added to your cart!');
9        });
10   });
```

OUTPUT:

AuraSound Wireless Headphones

$149.99

Experience immersive audio with these sleek, noise-cancelling wireless headphones. Up to 30 hours of battery life and crystal-clear call quality.

Add to Cart

OBSERVATION:

Task 3 – Event Registra on Form

Build an event registra on form for a conference.

Requirements:

• Collect name, email, phone number, and session selec on.

• Use AI to:

o Add form valida on with JavaScript. o Make
the form accessible with labels and ARIA. o
Style with a professional look.

PROMPT:

"Create the HTML, CSS, and JavaScript for an event registration form for a tech conference.

Form Fields:

1. Full Name (text input)
2. Email Address (email input)
3. Phone Number (tel input)
4. Session Selection (a dropdown/select menu with at least 3 options and a disabled default option).

CODE:

```html
form.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>Event Registration</title>
7       <link rel="stylesheet" href="form-style.css">
8   </head>
9   <body>
10
11      <div class="form-container">
12          <h1 class="form-title">Conference Registration</h1>
13          <form id="registrationForm" novalidate>
14              <!-- Name Field -->
15              <div class="form-group">
16                  <label for="fullName">Full Name</label>
17                  <input type="text" id="fullName" name="fullName" aria-required="true">
18                  <div class="error-message" id="nameError"></div>
19              </div>
20
21              <!-- Email Field -->
22              <div class="form-group">
23                  <label for="email">Email Address</label>
24                  <input type="email" id="email" name="email" aria-required="true">
25                  <div class="error-message" id="emailError"></div>
26              </div>
27
28              <!-- Phone Field -->
29              <div class="form-group">
30                  <label for="phone">Phone Number</label>
31                  <input type="tel" id="phone" name="phone" aria-required="true">
32                  <div class="error-message" id="phoneError"></div>
33              </div>
```

```html
29                <div class="form-group">
30                    <label for="phone">Phone Number</label>
31                    <input type="tel" id="phone" name="phone" aria-required="true">
32                    <div class="error-message" id="phoneError"></div>
33                </div>
34
35                <!-- Session Selection Field -->
36                <div class="form-group">
37                    <label for="session">Choose a Session</label>
38                    <select id="session" name="session" aria-required="true">
39                        <option value="" disabled selected>Select a session...</option>
40                        <option value="ai-ml">AI & Machine Learning</option>
41                        <option value="web-dev">Modern Web Development</option>
42                        <option value="cloud">Cloud Native Architectures</option>
43                    </select>
44                    <div class="error-message" id="sessionError"></div>
45                </div>
46
47                <button type="submit" class="submit-btn">Register</button>
48            </form>
49            <div id="successMessage" class="success-message"></div>
50        </div>
51
52        <script src="form-script.js"></script>
53
54    </body>
55    </html>
```

```css
# form-style.css > ...
1    /* --- General Body & Font Styles --- */
2    body {
3        font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen, Ubuntu, Cantarell, 'Open Sans', '
4        background-color: #eef2f7;
5        display: flex;
6        justify-content: center;
7        align-items: center;
8        min-height: 100vh;
9        padding: 1rem;
10   }
11
12   /* --- Form Container Styling --- */
13   .form-container {
14       background-color: #ffffff;
15       padding: 2.5rem;
16       border-radius: 10px;
17       box-shadow: 0 4px 20px rgba(0, 0, 0, 0.08);
18       width: 100%;
19       max-width: 500px;
20   }
21
22   .form-title {
23       text-align: center;
24       margin-bottom: 2rem;
25       color: #333;
26       font-weight: 600;
27   }
28
29   /* --- Form Group & Label Styling --- */
30   .form-group {
31       margin-bottom: 1.5rem;
32   }
```

```css
35        display: block;
36        margin-bottom: 0.5rem;
37        font-weight: 500;
38        color: #555;
39    }
40
41    /* --- Input & Select Styling --- */
42    input[type="text"],
43    input[type="email"],
44    input[type="tel"],
45    select {
46        width: 100%;
47        padding: 0.75rem;
48        border: 1px solid #ccc;
49        border-radius: 5px;
50        font-size: 1rem;
51        transition: border-color 0.3s ease, box-shadow 0.3s ease;
52    }
53
54    input:focus,
55    select:focus {
56        outline: none;
57        border-color: #007bff;
58        box-shadow: 0 0 0 3px rgba(0, 123, 255, 0.2);
59    }
60
61    /* --- Validation & Error Styling --- */
62    .error-message {
63        color: #dc3545;
64        font-size: 0.875rem;
65        margin-top: 0.25rem;
66        min-height: 1em; /* Prevents layout shift */
```

```css
69    .success-message {
70        color: #28a745;
71        text-align: center;
72        font-weight: 500;
73        margin-top: 1rem;
74    }
75
76    /* Add a class for invalid inputs */
77    .invalid {
78        border-color: #dc3545;
79    }
80
81    .invalid:focus {
82        box-shadow: 0 0 0 3px rgba(220, 53, 69, 0.2);
83    }
84
85    /* --- Button Styling --- */
86    .submit-btn {
87        width: 100%;
88        padding: 0.85rem;
89        background-color: #007bff;
90        color: #fff;
91        border: none;
92        border-radius: 5px;
93        font-size: 1.1rem;
94        font-weight: 600;
95        cursor: pointer;
96        transition: background-color 0.3s ease;
97    }
98
99    .submit-btn:hover {
100       background-color: #0056b3;
```

OUTPUT:

Conference Registration

Full Name

Email Address

Phone Number

Choose a Session

Select a session...

Register

OBSERVATION:

- **Accessibility:** The form is highly accessible. Every input has a corresponding `<label>` with a `for` attribute, which is crucial for screen reader users to understand what each field is for. The `aria-required="true"` attribute further informs assistive technologies that the field must be filled out.
- **JavaScript Validation:** The `novalidate` attribute on the `<form>` tag disables the browser's default validation popups, allowing our custom JavaScript to take full control. The script provides a superior user experience by:
  - Checking all fields at once on submission.
  - Displaying clear, specific error messages directly on the page.
  - Using CSS (`.invalid` class) to visually highlight problematic fields.
- **Professional Styling:** The CSS creates a modern and professional look. Key features include:
  - A soft background color for the page and a clean white card for the form.
  - Subtle `box-shadow` to make the form "float."
  - Clear visual feedback on `:focus` for inputs, improving usability.
  - A responsive design that ensures the form looks great on both desktop and mobile devices.

Task Descrip on #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.

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

Deliverables (For All Tasks)

1. AI-generated prompts for code and test case genera on.

2. At least 3 assert test cases for each task.

3. AI-generated ini al code and execu on screenshots.

4. Analysis of whether code passes all tests.

5. Improved final version with inline comments and explana on.

6. Compiled report (Word/PDF) with prompts, test cases, asser ons, code, and output.

PROMPT:

"Create an HTML page that fetches a list of users from the JSONPlaceholder API (https://jsonplaceholder.typicode.com/users) and displays their names and emails.

CODE:

```html
<> data.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 Data</title>
7      <link rel="stylesheet" href="data-style.css">
8  </head>
9  <body>
10     <h1>User List</h1>
11     <div id="user-container" class="user-container">
12         <!-- Content will be dynamically inserted here -->
13     </div>
14     <script src="data-script.js"></script>
15 </body>
16 </html>
```

```css
body {
    font-family: sans-serif;
    background-color: ■#f0f2f5;
    color: □#333;
    padding: 2rem;
}

h1 {
    text-align: center;
    color: □#1d2129;
}

.user-container {
    max-width: 600px;
    margin: 0 auto;
    background-color: ■#fff;
    border-radius: 8px;
    padding: 1rem;
    box-shadow: 0 2px 8px □rgba(0, 0, 0, 0.1);
}

/* --- Success State: User Item Styling --- */
.user-item {
    padding: 1rem;
    border-bottom: 1px solid ■#ddd;
}

.user-item:last-child {
    border-bottom: none;
}

.user-name {
    font-weight: bold;
```

```css
33      font-weight: bold;
34      font-size: 1.1rem;
35      margin: 0 0 0.25rem;
36  }
37
38  .user-email {
39      color: #666;
40      font-size: 0.9rem;
41      margin: 0;
42  }
43
44  /* --- Loading State: Skeleton Screen Styling --- */
45  .skeleton {
46      padding: 1rem;
47      border-bottom: 1px solid #ddd;
48  }
49
50  .skeleton-text {
51      height: 1em;
52      background-color: #e0e0e0;
53      border-radius: 4px;
54      animation: pulse 1.5s infinite ease-in-out;
55  }
56
57  .skeleton-text.title {
58      width: 40%;
59      margin-bottom: 0.5rem;
60  }
61
62  .skeleton-text.subtitle {
63      width: 60%;
64  }
```

```css
55   }
56
57   .skeleton-text.title {
58       width: 40%;
59       margin-bottom: 0.5rem;
60   }
61
62   .skeleton-text.subtitle {
63       width: 60%;
64   }
65
66   @keyframes pulse {
67       0% { background-color: ■#e0e0e0; }
68       50% { background-color: □#f0f0f0; }
69       100% { background-color: □#e0e0e0; }
70   }
71
72   /* --- Error State Styling --- */
73   .error-message {
74       color: ■#d93025;
75       font-weight: bold;
76       text-align: center;
77       padding: 2rem;
78   }
```

```js
JS data-script.js > ...
  1  document.addEventListener('DOMContentLoaded', () => {
  2      const userContainer = document.getElementById('user-container');
  3      const API_URL = 'https://jsonplaceholder.typicode.com/users';
  4
  5      // --- UI Rendering Functions ---
  6
  7      /**
  8       * Renders the initial skeleton loading state.
  9       */
 10      function renderLoadingState() {
 11          userContainer.innerHTML = ''; // Clear previous content
 12          // Create 5 skeleton placeholders
 13          for (let i = 0; i < 5; i++) {
 14              const skeleton = document.createElement('div');
 15              skeleton.className = 'skeleton';
 16              skeleton.innerHTML = `
 17                  <div class="skeleton-text title"></div>
 18                  <div class="skeleton-text subtitle"></div>
 19              `;
 20              userContainer.appendChild(skeleton);
 21          }
 22      }
 23
 24      /**
 25       * Renders the list of users from the fetched data.
 26       * @param {Array} users - An array of user objects.
 27       */
 28      function renderSuccessState(users) {
 29          userContainer.innerHTML = ''; // Clear previous content
 30          users.forEach(user => {
 31              // Create elements safely to prevent XSS
 32              const userItem = document.createElement('div');
 33              userItem.className = 'user-item';
```

```js
 34
 35              const nameEl = document.createElement('p');
 36              nameEl.className = 'user-name';
 37              nameEl.textContent = user.name; // Use textContent for safety
 38
 39              const emailEl = document.createElement('p');
 40              emailEl.className = 'user-email';
 41              emailEl.textContent = user.email; // Use textContent for safety
 42
 43              userItem.appendChild(nameEl);
 44              userItem.appendChild(emailEl);
 45              userContainer.appendChild(userItem);
 46          });
 47      }
 48
 49      /**
 50       * Renders an error message.
 51       * @param {string} message - The error message to display.
 52       */
 53      function renderErrorState(message) {
 54          userContainer.innerHTML = ''; // Clear previous content
 55          const errorEl = document.createElement('p');
 56          errorEl.className = 'error-message';
 57          errorEl.textContent = message;
 58          userContainer.appendChild(errorEl);
 59      }
```

```
65      */
66      async function fetchUsers() {
67          renderLoadingState();
68          // Test Case #1 Assertion: Check if loading state is rendered
69          console.assert(document.querySelector('.skeleton'), "Test 1 Failed: Skeleton loader should be visible.");
70
71          try {
72              const response = await fetch(API_URL);
73              if (!response.ok) {
74                  throw new Error(`Network response was not ok (Status: ${response.status})`);
75              }
76              const users = await response.json();
77              renderSuccessState(users);
78
79              // Test Case #2 Assertion: Check if data is rendered correctly
80              const userItems = document.querySelectorAll('.user-item');
81              console.assert(userItems.length === users.length, `Test 2 Failed: Expected ${users.length} user items, but found
82
83          } catch (error) {
84              console.error('Fetch error:', error);
85              renderErrorState('Failed to load users. Please try again later.');
86
87              // Test Case #3 Assertion (Manual Trigger): Check if error state is rendered
88              // To run this test, change API_URL to a bad URL like 'https://invalid.url'
89              console.assert(document.querySelector('.error-message'), "Test 3 Failed: Error message should be visible on fetc
90          }
91      }
92
93      // --- Initial Execution ---
94      fetchUsers();
95  });
```

OUTPUT:

User List

Leanne Graham
Sincere@april.biz

Ervin Howell
Shanna@melissa.tv

Clementine Bauch
Nathan@yesenia.net

Patricia Lebsack
Julianne.OConner@kory.org

Chelsey Dietrich
Lucio_Hettinger@annie.ca

Mrs. Dennis Schulist
Karley_Dach@jasper.info

Kurtis Weissnat
Telly.Hoeger@billy.biz

Nicholas Runolfsdottir V
Sherwood@rosamond.me

Glenna Reichert
Chaim_McDermott@dana.io

Observa on:

- **User Experience (UX):** The skeleton screen is a significant improvement over a simple "Loading..." text. It reduces perceived wait time and prevents page layout shifts when the real content loads. The clear error message is also crucial for a good user experience.
- **Code Quality & Security:**
  - The logic is cleanly separated into functions for each state (`renderLoadingState`, `renderSuccessState`, `renderErrorState`), making the code easy to read and maintain.
  - Using `async/await` with a `try...catch` block is the modern standard for handling asynchronous operations like `fetch`.
  - **The most critical aspect is security.** The code correctly uses `document.createElement` and sets `element.textContent`. This automatically sanitizes the input, preventing malicious HTML or script tags in the API data from being executed (an XSS attack). Using `element.innerHTML = ...` would have been insecure.
- **Testing:** The `console.assert()` statements serve as simple, effective integration tests. They verify that the correct UI is be rendered in response to different API outcomes, confirming that the core logic works as expected.