

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: B. Tech		Assignment Type: Lab	
Course Coordinator Name		Venkataramana Veeramsetty	
Instructor(s) Name		Dr. V. Venkataramana (Co-ordinator) Dr. T. Sampath Kumar Dr. Pramoda Patro Dr. Brij Kishor Tiwari Dr.J.Ravichander Dr. Mohammand Ali Shaik Dr. Anirodh Kumar Mr. S.Naresh Kumar Dr. RAJESH VELPULA Mr. Kundhan Kumar Ms. Ch.Rajitha Mr. M Prakash Mr. B.Raju Intern 1 (Dharma teja) Intern 2 (Sai Prasad) Intern 3 (Sowmya) NS_2 (Mounika)	
Course Code	24CS002PC215	Course Title	AI Assisted Coding
Year/Sem	II/I	Regulation	R24
Date and Day of Assignment	Week7 - Monday	Time(s)	
Duration	2 Hours	Applicable to Batches	
AssignmentNumber: 14.1(Present assignment number)/ 24 (Total number of assignments)			
Q.No	Question		Expected Time to complete
1	Lab 14: Web Design Application – AI-Assisted HTML/CSS/JS Generation Lab Objectives <ul style="list-style-type: none"> Design functional, visually appealing web applications using HTML, CSS, and JavaScript with AI assistance. Apply responsive, accessible, and interactive design principles. 		Week 7 - Monday

- Create **practical UI components** for real-world web applications.
- Use AI to optimize **layout, UX, and performance**.

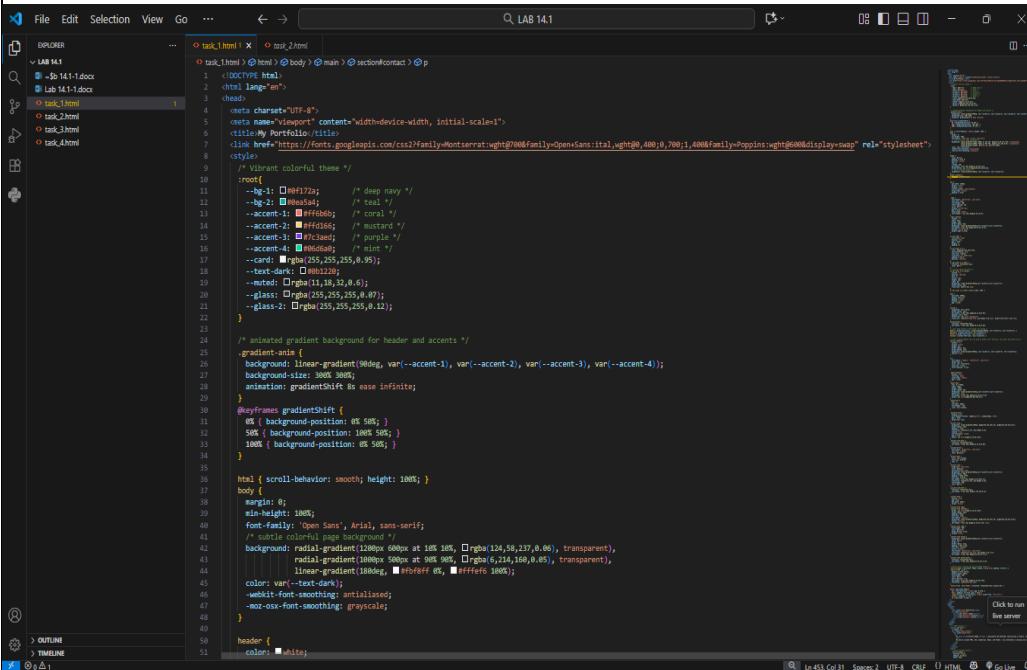
Task 1 – Portfolio Website Design

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: Generate a responsive portfolio layout with About, Projects, and Contact sections using Flexbox/Grid. Include smooth scrolling and apply the suggested color palette and typography.



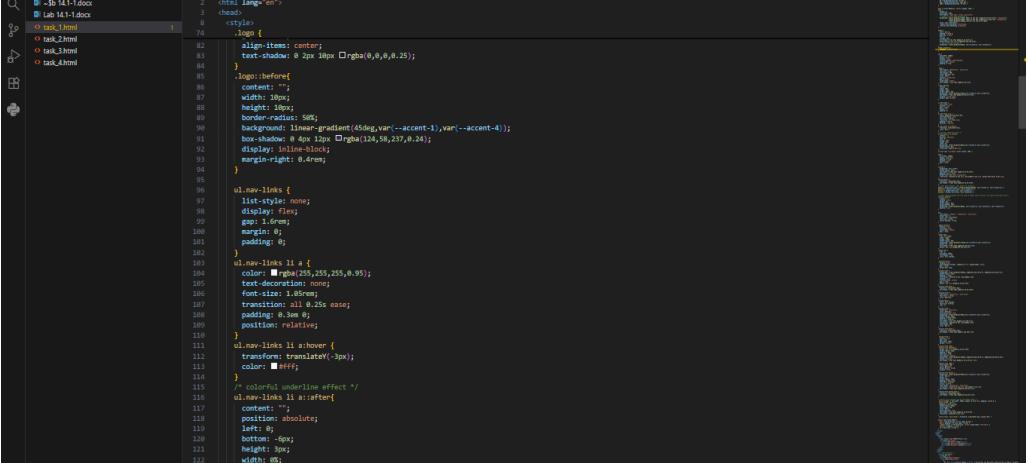
The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows files: task1.html, task2.html, task3.html, task4.html, and Lab 14.1-1.docx.
- Code Editor:** Displays the following code:

```

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>My Portfolio</title>
    <link href="https://fonts.googleapis.com/css2?family=Montserrat:wght@400;600&family=Poppins:wght@400;600&display=swap" rel="stylesheet">
  </head>
  <body>
    <div class="root" style="background-color: #f0f0f0; height: 100vh; width: 100%; display: flex; align-items: center; justify-content: center; font-family: 'Poppins', sans-serif; margin: 0; padding: 0; position: relative; font-size: 16px; line-height: 1.5; color: #333; transition: background-color 0.3s ease; z-index: 1;>
      <div style="background-color: #fff; padding: 20px; border-radius: 10px; text-align: center; width: fit-content; position: relative; z-index: 2;>
        <h1>Welcome to My Portfolio!</h1>
        <p>This is a simple responsive portfolio website I created for practice. It features a clean design with a color palette of deep navy, teal, coral, mustard, lime, mint, and grey. The layout uses a combination of Flexbox and Grid for a modern look. Smooth scrolling is implemented using CSS scroll-behavior: smooth; for the main content area. The header has a subtle color gradient background that shifts as you scroll down the page. The footer includes links to GitHub, LinkedIn, and a contact form. I hope you like it!</p>
        <div style="text-align: right; margin-top: 10px; position: absolute; bottom: 0; right: 0; font-size: 0.8em; transform: rotate(-45deg);>
          Click to run live server
        </div>
      </div>
    </div>
    <div style="position: absolute; top: 0; left: 0; width: 100%; height: 100%; background-color: #fff; z-index: 0; transition: background-color 0.3s ease; >
    </div>
  </body>

```



```
task_1.html
task_1.html > task_1.html
task_1.html > task_1.html > main > section>contact > p

1 <p>Contact</p>
2 <div lang="en">
3   <div>
4     <div>
5       <div>
6         .logo {
7           align-items: center;
8           text-shadow: @2px 10px #000000;
9         }
10        .Logo::before{
11          content: '';
12          width: 10px;
13          height: 10px;
14          border-radius: 50%;
15          background: linear-gradient(45deg, var(--accent-1), var(--accent-4));
16          background-size: 4px 1px #000000;
17          display: inline-block;
18          margin-right: 0.4rem;
19        }
20        ul.nav-links {
21          list-style: none;
22          display: flex;
23          gap: 1em;
24          margin: 0;
25          padding: 0;
26        }
27        ul.nav-links li {
28          color: #000000;
29          text-decoration: none;
30          font-size: 1.05em;
31          transition: width 0.25s ease;
32          padding: 0.8em 0;
33          position: relative;
34        }
35        ul.nav-links li a:hover {
36          transform: translateX(-3px);
37          color: #fff;
38        }
39        /* colorful underline effect */
40        ul.nav-links li a::after{
41          content: "";
42          position: absolute;
43          left: 0;
44          top: 0.8px;
45          height: 3px;
46          width: 0%;
47          background: linear-gradient(90deg, var(--accent-2), var(--accent-1));
48          border-radius: 8px;
49          transition: width 0.25s ease;
50        }
51        ul.nav-links li a::before{
52          width: 100%;
```

The screenshot shows a code editor interface with the following details:

- File Path:** EXPLORER / task_1.html
- Code Content:** The code is for a portfolio website. It includes sections for projects (Portfolio Website, Sports Registration Portal), skills (Python, HTML, CSS, JavaScript), and contact information (Name: Pranith Reddy, Email: pranith.reddy@example.com, Phone: (123) 456-7890).

```
<html lang="en">
  <body>
    <main>
      <script>
        </script>
      <section id="projects">
        <h2>Projects</h2>
        <div class="projects-grid">
          <div class="project-card">
            <span class="project-title">Portfolio Website</span>
            <div class="project-desc">A modern personal website with a blog and showcase gallery. Built with React and deployed with Netlify.</div>
            <a class="project-link" href="#" target="_blank" rel="noopener noreferrer">View Project</a>
          </div>
          <div class="project-card">
            <span class="project-title">Sports Registration Portal</span>
            <div class="project-desc">A mini sports registration website where users can register for sports events, featuring drag & drop and persistent storage.</div>
            <a class="project-link" href="#" target="_blank" rel="noopener noreferrer">View Project</a>
          </div>
        </div>
      </section>
      <!-- Skills Section -->
      <section id="skills">
        <h2>Skills</h2>
        <div class="skills-wrapper" aria-label="Scrolling list of skills">
          <div class="skills-track" role="list">
            <span class="skill" role="listitem">Python</span>
            <span class="skill" role="listitem">C</span>
            <span class="skill" role="listitem">HTML</span>
            <span class="skill" role="listitem">CSS</span>
            <span class="skill" role="listitem">JavaScript</span>
            <!-- Duplicate for seamless scrolling -->
            <span class="skill" role="listitem">Python</span>
            <span class="skill" role="listitem">C</span>
            <span class="skill" role="listitem">HTML</span>
            <span class="skill" role="listitem">CSS</span>
            <span class="skill" role="listitem">JavaScript</span>
          </div>
        </div>
      </section>
      <!-- Contact Section -->
      <section id="contact">
        <h3>Contact Details</h3>
        <p>Name: Pranith Reddy</p>
        <p>Email: pranith.reddy@example.com</p>
        <p>Phone: (123) 456-7890</p>
      </section>
    </main>
  </body>
</html>
```

- Bottom Status Bar:** Shows the current file is "task_1.html", line 453, column 31. It also includes buttons for "Click to run live server", "HTML", "Go Live", and other standard editor controls.

The image displays two screenshots of a portfolio website titled "MyPortfolio".

Screenshot 1: About Section

The "About" section features a pink header with the title "MyPortfolio". Below it is a purple header with navigation links: "About", "Projects", and "Contact". A "School" icon is visible in the top right corner. The main content area has a light blue background and contains the following text:

Hi! I'm **Pranith Reddy**, a passionate web developer specializing in modern, responsive front-end and user experiences. I have an interest in exploring new knowledge about modern technology & apps.

My skills include HTML, CSS, JavaScript, React, and Python. I am interested in learning new technologies and collaborating on interesting projects!

Screenshot 2: Projects Section

The "Projects" section has a green header with the title "MyPortfolio". Below it is a purple header with navigation links: "About", "Projects", and "Contact". The main content area shows two projects:

- Portfolio Website**: A modern personal website with a blog and showcase gallery. Built with React and deployed with Netlify.
- Sports Registration Portal**: A mini sports registration website where users can register for sports events, featuring drag & drop and payment storage.

Skills Section

The "Skills" section has an orange header with the title "MyPortfolio". Below it is a purple header with navigation links: "About", "Projects", and "Contact". The main content area shows a list of skills:

- cript
- Python
- C
- HTML
- CSS
- JavaScript

Contact Details Section

The "Contact Details" section has a purple header with the title "MyPortfolio". Below it is a green header with navigation links: "About", "Projects", and "Contact". The main content area shows contact information:

Name: Pranith Reddy
Email: pranith.reddy@example.com
Phone: (123) 456-7890

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.

	<ul style="list-style-type: none">• Use AI to:<ul style="list-style-type: none">◦ Style with BEM methodology.◦ Make layout responsive.◦ Add hover effects and "Add to Cart" alert. <p>Prompt: Create a responsive product card using BEM CSS. Include image, title, price, and an 'Add to Cart' button with hover effects and alert</p> <td></td>	
--	---	--


```

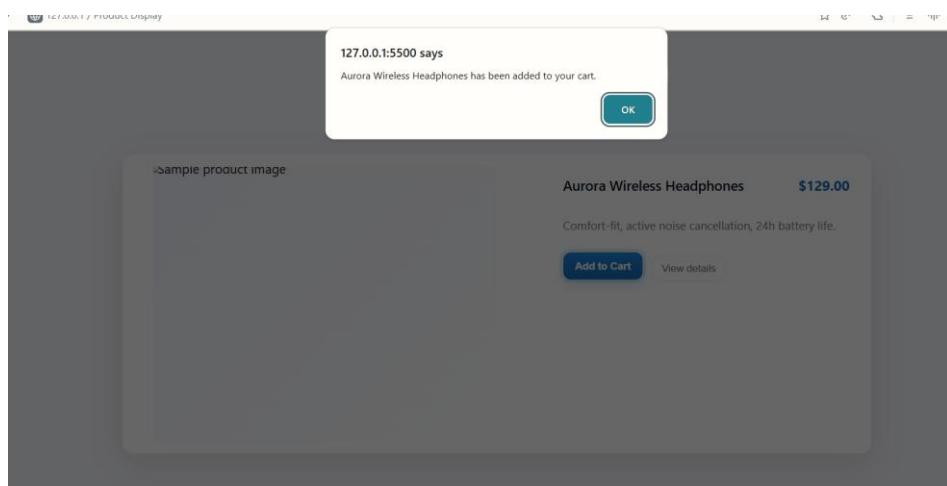
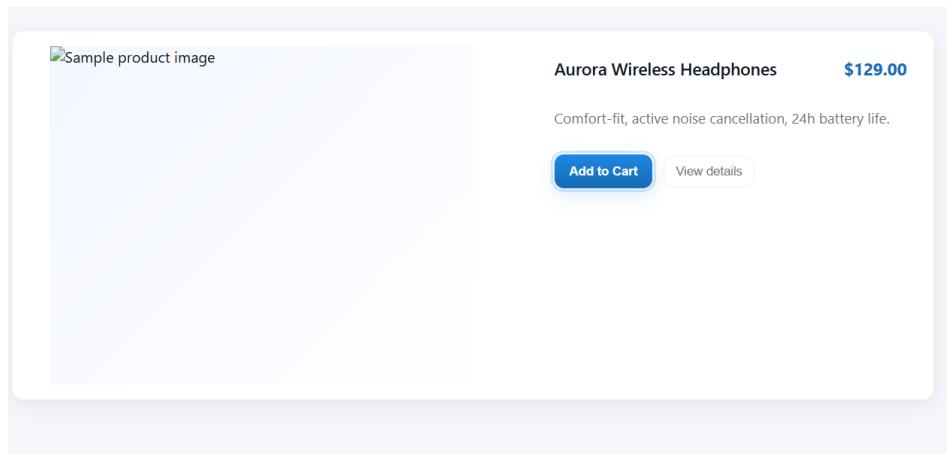
1  .product__title {
2    margin:0;
3    font-size:1.125rem;
4    line-height:1.2;
5    font-weight:600;
6    color:□#0f1724;
7  }
8
9  .product__price {
10   font-weight:700;
11   color:var(--accent-strong);
12   font-size:1.125rem;
13 }
14
15 .product__meta {
16   color:var(--muted);
17   font-size:.95rem;
18 }
19
20 .product__actions {
21   display:flex;
22   gap:12px;
23   align-items:center;
24 }
25
26 .product__button {
27   appearance:none;
28   border:0;
29   background:linear-gradient(180deg,var(--accent),var(--accent-strong));
30   color:□white;
31   padding:10px 16px;
32   font-weight:600;
33   border-radius:10px;
34   cursor:pointer;
35   transition:transform .15s ease, box-shadow .15s ease, filter .15s ease;
36   box-shadow:8 6px 18px □rgba(38,136,229,0.18);
37 }
38 .product__button:hover { transform:translateY(-3px); filter:brightness(.98); }
39 .product__button:active { transform:translateY(0); }
40 .product__button:focus { outline:3px solid □rgba(38,136,229,0.18); outline-offset:2px; }
41
42 .product__secondary {
43   background:transparent;
44   color:var(--muted);
45   padding:8px 12px;
46   border-radius:10px;
47   border:1px solid □rgba(15,23,36,0.06);
48   cursor:pointer;
49 }
50
51 /* Responsive: desktop layout with image left, info right */
52 @media (min-width:720px){
53   .product__card {
54     flex-direction:row;
55     align-items:stretch;
56   }
57   .product__media {
58     flex:0 0 46%;
59     min-height:360px;
60   }
61   .product__info {
62     flex:1;
63     padding:28px;
64   }
65 }
66
67 /* Smaller tweaks */
68 @media (max-width:420px){
69   .product__title { font-size:1rem; }
70   .product__price { font-size:1rem; }
71 }
72 </style>
73 </head>
74 <body>
75   <main class="product" role="main">
76     <article class="product__card" aria-label="Product card">
77       <figure class="product__media" aria-hidden="false">
78         

```

```

159 </figure>
160
161 <section class="product_info">
162   <div class="product_row">
163     <h1 class="product_title">Aurora Wireless Headphones</h1>
164     <div class="product_price" aria-hidden="False">$129.00</div>
165   </div>
166
167   <p class="product_meta">Comfort-fit, active noise cancellation, 24h battery life.</p>
168
169   <div class="product_actions" role="group" aria-label="Purchase actions">
170     <button type="button" class="product_button" id="addToCart" data-product-name="Aurora Wireless Headphones" aria-label="Add Aurora Wireless Headphones to cart">
171       Add to Cart
172     </button>
173     <button type="button" class="product_secondary" aria-label="View details">View details</button>
174   </div>
175 </section>
176 </article>
177 </main>
178
179 <script>
180   // Simple Add to Cart alert (accessible)
181   <function>()
182     const btn = document.getElementById('addToCart');
183     btn.addEventListener('click', function(e){
184       const name = this.getAttribute('data-product-name') || document.querySelector('.product_title').textContent.trim();
185       // Use setTimeout to ensure focus behavior doesn't get interrupted by alert on some browsers
186       setTimeout(() => { alert(name + ' has been added to your cart.'), 10);
187     });
188
189   // Keyboard "Enter" and "Space" handled automatically for button element
190   })();
191 </script>
192 </body>
193 </html>

```



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: Build an accessible event registration form with ARIA, labels, and JavaScript validation for name, email, phone, and session

```

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" />
6    <title>Event Registration</title>
7    <style>
8      :root{font-family:system-ui,-apple-system,Segoe UI,Roboto,"Helvetica Neue",Arial}
9      body{max-width:780px;margin:2rem auto;padding:1rem}
10     fieldset{border:1px solid #ccc;padding:1rem;margin-bottom:1rem}
11     legend{font-weight:600}
12     label{display:block;margin-top:.5rem}
13     input,select,button{font-size:1rem;padding:.45rem;margin-top:.25rem;width:100%;box-sizing:border-box}
14     .row{display:flex;gap:1rem}
15     .row > *{flex:1}
16     .error{color:#d00;font-size:95rem;margin-top:.25rem}
17     .is-invalid{outline:2px solid #ff0000}
18     .sr-only{position:absolute!important;width:1px;height:1px;padding:0; margin:-1px;overflow:hidden;clip:rect(0 0 0 0);white-space nowrap; border:0}
19     .success{background:#e0f2e0; border:1px solid #2aa34a; padding:.75rem; margin-top:1rem}
20   </style>
21 </head>
22 <body>
23   <h1>Event Registration</h1>
24
25   <form id="registrationForm" novalidate aria-describedby="formInstructions" role="form">
26     <p id="formInstructions">Fill in your details and choose a session. All fields are required.</p>
27
28     <fieldset>
29       <legend>Personal Information</legend>
30
31       <label for="name">Full name</label>
32       <input id="name" name="name" type="text" autocomplete="name" aria-required="true" aria-describedby="nameError" required />
33       <div id="nameHelp" class="sr-only">Enter your first and last name, at least 2 characters.</div>
34       <div id="nameError" class="error" role="alert" aria-live="assertive"></div>
35
36       <label for="email">Email address</label>
37       <input id="email" name="email" type="email" autocomplete="email" aria-required="true" aria-describedby="emailError" required />
38       <div id="emailError" class="error" role="alert" aria-live="assertive"></div>
39
40       <label for="phone">Phone number</label>
41       <input id="phone" name="phone" type="tel" inputmode="tel" aria-required="true" aria-describedby="phoneError" required />
42       <div id="phoneHelp" class="sr-only">Include country code if not local. Allowed characters: digits, spaces, parentheses, hyphens, plus sign.</div>
43       <div id="phoneError" class="error" role="alert" aria-live="assertive"></div>
44     </fieldset>
45
46     <fieldset>
47       <legend>Session selection</legend>
48
49       <label for="session">Choose a session</label>
50       <select id="session" name="session" aria-required="true" aria-describedby="sessionError" required>
51         <option value=""> Select a session </option>
52         <option value="am">Morning - Accessibility Basics</option>
53         <option value="pm">Afternoon - Advanced ARIA & Testing</option>
54         <option value="evening">Evening - Hands-on Workshop</option>
55       </select>

```

```
</select>
<div id="sessionError" class="error" role="alert" aria-live="assertive"></div>
</fieldset>

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

<div id="formMessage" class="success" role="status" aria-live="polite" hidden></div>
</form>

<script>
// GitHub Copilot
(function () {
    const form = document.getElementById('registrationForm');
    const fields = {
        name: document.getElementById('name'),
        email: document.getElementById('email'),
        phone: document.getElementById('phone'),
        session: document.getElementById('session')
    };
    const errors = {
        name: document.getElementById('nameError'),
        email: document.getElementById('emailError'),
        phone: document.getElementById('phoneError'),
        session: document.getElementById('sessionError')
    };
    const formMessage = document.getElementById('formMessage');

    function clearErrors() {
        Object.values(errors).forEach(e => {
            e.textContent = '';
        });
        Object.values(fields).forEach(f => {
            f.removeAttribute('aria-invalid');
            f.classList.remove('is-invalid');
        });
        formMessage.hidden = true;
        formMessage.textContent = '';
    }

    function focusFirstInvalid(invalidField) {
        if (invalidField && typeof invalidField.focus === 'function') {
            invalidField.focus();
            invalidField.scrollIntoView({behavior: 'smooth', block: 'center'});
        }
    }

    function validateName(value) {
        if (!value.trim()) return 'Name is required.';
        if (value.trim().length < 2) return 'Please enter at least 2 characters.';
        // allow letters, hyphens, apostrophes, spaces
        if (/^([A-Za-z]+(-[A-Za-z]+)*|[A-Za-z]+')\s*/.test(value)) return 'Name contains invalid characters';
    }
})()
```

```
103     if (value.trim().length < 2) return 'Please enter at least 2 characters.';
104     // allow letters, hyphens, apostrophes, spaces
105     if (!/^(?![\u00c0-\u00ff]).*\w+$/u.test(value.trim())) return 'Name contains invalid characters.';
106     return '';
107 }
108
109 function validateEmail(value) {
110     if (!value.trim()) return 'Email is required.';
111     // basic RFC-like check
112     const re = /^[\w\.-]+@[^\w\.-]+\.[^\w\.-]+$/;
113     if (!re.test(value.trim())) return 'Please enter a valid email address.';
114     return '';
115 }
116
117 function validatePhone(value) {
118     if (!value.trim()) return 'Phone number is required.';
119     // allow "+", digits, spaces, parentheses, hyphens; require 7-20 digits total
120     const digits = value.replace(/\D/g, '');
121     if (digits.length < 7 || digits.length > 20) return 'Enter a valid phone number with 7-20 digits.';
122     if (!/^\d{1,3}(\d{1,3}){1,3}\d{1,4}$/.test(digits)) return 'Phone number contains invalid characters.';
123     return '';
124 }
125
126 function validateSession(value) {
127     if (!value) return 'Please select a session.';
128     return '';
129 }
130
131 form.addEventListener('submit', function (ev) {
132     ev.preventDefault();
133     clearErrors();
134
135     const vName = fields.name.value;
136     const vEmail = fields.email.value;
137     const vPhone = fields.phone.value;
138     const vSession = fields.session.value;
139
140     const nameErr = validateName(vName);
141     const emailErr = validateEmail(vEmail);
142     const phoneErr = validatePhone(vPhone);
143     const sessionErr = validateSession(vSession);
144
145     const feedback = { name: nameErr, email: emailErr, phone: phoneErr, session: sessionErr };
146
147     let firstInvalid = null;
148     Object.keys(feedback).forEach(key => {
149         const message = feedback[key];
150         if (message) {
151             errors[key].textContent = message;
152             fields[key].setAttribute('aria-invalid', 'true');
```

```

152         fields[key].setAttribute('aria-invalid', 'true');
153         fields[key].classList.add('is-invalid');
154     }
155   });
156 });
157 if (firstInvalid) {
158   focusFirstInvalid(firstInvalid);
159   return;
160 }
161
162 // If all valid: simulate submission success
163 formMessage.hidden = false;
164 formMessage.textContent = 'Registration received. A confirmation email will be sent shortly.';
165 // Clear form (optional)
166 form.reset();
167 // move focus to confirmation for screen reader users
168 formMessage.focus?_();
169 });
170
171 // Live inline validation (on blur)
172 Object.keys(fields).forEach(key => {
173   fields[key].addEventListener('blur', () => {
174     let msg = '';
175     switch (key) {
176       case 'name': msg = validateName(fields.name.value); break;
177       case 'email': msg = validateEmail(fields.email.value); break;
178       case 'phone': msg = validatePhone(fields.phone.value); break;
179       case 'session': msg = validateSession(fields.session.value); break;
180     }
181     errors[key].textContent = msg;
182     if (msg) {
183       fields[key].setAttribute('aria-invalid', 'true');
184       fields[key].classList.add('is-invalid');
185     } else {
186       fields[key].removeAttribute('aria-invalid');
187       fields[key].classList.remove('is-invalid');
188     }
189   });
190 });
191 });
192 })();
193 </script>
194 </body>
195 </html>

```

127.0.0.1 / Event Registration

Event Registration

Fill in your details and choose a session. All fields are required.

Personal information

Full name	<input type="text"/>
Email address	<input type="text"/>
Phone number	<input type="text"/>

Session selection

Choose a session	<input type="text" value="— Select a session —"/>
------------------	---

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: Write fetch logic to get JSON data, show loading text, handle errors, and render list items to the DOM

```
task_4.html > <!DOCTYPE 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" />
6      <title>Fetch JSON and Render List</title>
7      <style>
8          body { font-family: system-ui, sans-serif; margin: 20px; }
9          #loading { color: #0a58ca; }
10         #error { color: #b02a37; margin-top: 12px; }
11         ul { padding-left: 20px; }
12         li { margin-bottom: 8px; }
13         button { margin-top: 8px; }
14     </style>
15 </head>
16 <body>
17     <h1>Users</h1>
18
19     <div id="loading" aria-live="polite" hidden>Loading...</div>
20     <div id="error" role="alert" hidden></div>
21
22     <ul id="list" aria-live="polite"></ul>
23
24     <script>
25         // Change to the API you want to fetch
26         const API_URL = 'https://jsonplaceholder.typicode.com/users';
27
28         const loadingEl = document.getElementById('loading');
29         const errorEl = document.getElementById('error');
30         const listEl = document.getElementById('list');
31
32         async function fetchAndRender(url = API_URL) {
33             showLoading(true);
34             showError(null);
35             clearList();
36
37             try {
38                 const res = await fetch(url, { cache: 'no-store' });
39                 if (!res.ok) {
40                     throw new Error(`Server responded with ${res.status} ${res.statusText}`);
41                 }
42
43                 const data = await res.json();
44
45                 if (!Array.isArray(data)) {
46                     throw new Error('Expected an array in response');
47                 }
48
49                 renderList(data);
50             } catch (err) {
51                 showError(err.message || 'An unknown error occurred');
52             } finally {
53                 showLoading(false);
54             }
55         }
56     </script>
```

```

0      </script>
1  
```

```

2  function showLoading(visible) {
3      loadingEl.hidden = !visible;
4  }
5
6  function showError(message) {
7      if (!message) {
8          errorEl.hidden = true;
9          errorEl.textContent = '';
10         return;
11     }
12     errorEl.hidden = false;
13     errorEl.textContent = message + ' ';
14     // add a retry button when there's an error
15     const retry = document.createElement('button');
16     retry.textContent = 'Retry';
17     retry.addEventListener('click', () => fetchAndRender());
18     // clear any previous children and append
19     errorEl.appendChild(retry);
20 }
21
22 function clearList() {
23     listEl.innerHTML = '';
24 }
25
26 function renderList(items) {
27     if (items.length === 0) {
28         listEl.innerHTML = '<li>No items found.</li>';
29         return;
30     }
31
32     const fragment = document.createDocumentFragment();
33
34     items.forEach(item => {
35         // adjust display fields depending on the JSON schema
36         // for JSONPlaceholder users we show name and email
37         const li = document.createElement('li');
38         const title = item.name || item.title || item.id || 'Unnamed item';
39         const subtitle = item.email ? ' - ' + item.email : (item.body ? ' - ' + String(item.body).slice(0, 60) + '...' : '');
40         li.textContent = title + subtitle;
41         fragment.appendChild(li);
42     });
43
44     listEl.appendChild(fragment);
45 }
46
47 // start on load
48 window.addEventListener('DOMContentLoaded', () => {
49     fetchAndRender();
50 });
51
52 </script>
53 </body>
54

```

Users

- 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
- Clementina DuBuque — Rey.Padberg@karina.biz

	<input checked="" type="checkbox"/> Deliverables (For All Tasks)	
--	--	--

1. AI-generated prompts for code and test case generation.
2. At least 3 assert test cases for each task.
3. AI-generated initial code and execution screenshots.
4. Analysis of whether code passes all tests.
5. Improved final version with inline comments and explanation.
6. Compiled report (Word/PDF) with prompts, test cases, assertions, code, and output.