

EXP-1:

Create a student registration form with HTML5 validation for name, email, and age inputs:

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Student Registration Form</title>

  <script src="https://cdn.tailwindcss.com"></script>

  <!-- Importing the 'Poppins' font -->

  <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600;700&display=swap"
rel="stylesheet">

  <style>

    /* Applying the new font and custom validation colors */

    body {

      font-family: 'Poppins', sans-serif;

    }

    input:invalid {

      border-color: #f87171; /* red-400 */

    }

    input:valid {

      border-color: #34d399; /* emerald-400 */

    }

    /* Simple spinner for loading state */

    .spinner {

      border: 4px solid rgba(0, 0, 0, 0.1);

      width: 36px;

      height: 36px;

      border-radius: 50%;
```

```

        border-left-color: #14b8a6; /* teal-500 */

        animation: spin 1s ease infinite;
    }

    @keyframes spin {
        0% { transform: rotate(0deg); }
        100% { transform: rotate(360deg); }
    }
</style>
</head>
<body class="bg-gray-50 flex items-center justify-center min-h-screen p-4">
    <div class="w-full max-w-md mx-auto bg-white rounded-2xl shadow-xl p-8">
        <!-- Form Header -->
        <div class="text-center mb-8">
            <h1 class="text-3xl font-bold text-gray-800">Register for a Course</h1>
            <p class="text-gray-500 mt-2">Let's get you started</p>
        </div>

        <!-- Registration Form -->
        <form id="registrationForm" novalidate>
            <!-- Name Input -->
            <div class="mb-6">
                <label for="name" class="block text-sm font-medium text-gray-700 mb-2">Full Name</label>
                <input type="text" id="name" name="name"
                    class="w-full px-4 py-3 bg-gray-100 border-2 border-gray-200 rounded-lg focus:outline-
                    none focus:ring-2 focus:ring-teal-400 focus:border-teal-400 transition duration-200"
                    placeholder="e.g., Alex Doe"
                    required
                    minlength="3">
            </div>

            <!-- Email Input -->

```

```
<div class="mb-6">

    <label for="email" class="block text-sm font-medium text-gray-700 mb-2">Email
Address</label>

    <input type="email" id="email" name="email"

        class="w-full px-4 py-3 bg-gray-100 border-2 border-gray-200 rounded-lg focus:outline-
none focus:ring-2 focus:ring-teal-400 focus:border-teal-400 transition duration-200"

        placeholder="you@example.com"

        required>

</div>
```

```
<!-- Age Input -->

<div class="mb-6">

    <label for="age" class="block text-sm font-medium text-gray-700 mb-2">Age</label>

    <input type="number" id="age" name="age"

        class="w-full px-4 py-3 bg-gray-100 border-2 border-gray-200 rounded-lg focus:outline-
none focus:ring-2 focus:ring-teal-400 focus:border-teal-400 transition duration-200"

        placeholder="Enter your age"

        required

        min="16"

        max="100">

</div>
```

```
<!-- AI Course Suggestion Section (initially hidden) -->

<div id="course-suggestion-section" class="hidden mb-8 text-center">

    <button type="button" id="suggest-courses-btn" class="w-full bg-gray-700 text-white font-
semibold py-2 px-4 rounded-lg hover:bg-gray-800 focus:outline-none focus:ring-2 focus:ring-offset-2
focus:ring-gray-600 transition duration-300">

        Suggest Courses For Me

    </button>

</div>
```

```
<!-- Submit Button -->

<button type="submit"
```

```
        class="w-full bg-teal-500 text-white font-bold py-3 px-4 rounded-lg hover:bg-teal-600
focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-teal-500 transition duration-300 ease-in-
out shadow-lg hover:shadow-xl">
```

```
            Register Now
```

```
        </button>
```

```
    </form>
```

```
</div>
```

```
<!-- Modal for Gemini API Response -->
```

```
<div id="gemini-modal" class="fixed inset-0 bg-black bg-opacity-50 flex items-center justify-center
p-4 hidden z-50">
```

```
    <div class="bg-white rounded-xl shadow-2xl w-full max-w-lg max-h-[90vh] flex flex-col">
```

```
        <div class="flex justify-between items-center p-4 border-b">
```

```
            <h3 id="modal-title" class="text-lg font-semibold text-gray-800">AI Response</h3>
```

```
            <button id="modal-close-btn" class="p-2 rounded-full hover:bg-gray-200 text-gray-500
hover:text-gray-800">
```

```
                &times;
```

```
            </button>
```

```
        </div>
```

```
    <div id="modal-content" class="p-6 overflow-y-auto">
```

```
        <!-- Loading spinner or content will be shown here -->
```

```
    </div>
```

```
    <div id="modal-footer" class="p-4 border-t text-right">
```

```
        <button id="modal-action-btn" class="bg-teal-500 text-white font-bold py-2 px-4 rounded-lg
hover:bg-teal-600 transition">Close</button>
```

```
    </div>
```

```
</div>
```

```
</div>
```

```
<script>
```

```
    // --- DOM Elements ---
```

```
    const form = document.getElementById('registrationForm');
```

```
    const ageInput = document.getElementById('age');
```

```

const courseSuggestionSection = document.getElementById('course-suggestion-section');
const suggestCoursesBtn = document.getElementById('suggest-courses-btn');
const geminiModal = document.getElementById('gemini-modal');
const modalTitle = document.getElementById('modal-title');
const modalContent = document.getElementById('modal-content');
const modalCloseBtn = document.getElementById('modal-close-btn');
const modalActionBtn = document.getElementById('modal-action-btn');

// --- Event Listeners ---

// Show course suggestion button when a valid age is entered
ageInput.addEventListener('input', () => {
  if (ageInput.checkValidity()) {
    courseSuggestionSection.classList.remove('hidden');
  } else {
    courseSuggestionSection.classList.add('hidden');
  }
});

// Handle form submission
form.addEventListener('submit', async function(event) {
  event.preventDefault();
  if (form.checkValidity()) {
    const studentName = document.getElementById('name').value.split(' ')[0];
    showModal('Registration Successful!');

    modalContent.innerHTML = `<div class="flex justify-center items-center h-32"><div
class="spinner"></div></div>`;

    const prompt = `Generate a short, friendly, and encouraging welcome message for a new
student named ${studentName} who just registered for a course. Mention that their registration is
confirmed.`;

    const responseText = await callGeminiAPI(prompt);

```

```

        modalContent.innerHTML = `<div class="prose max-w-
none">${formatResponse(responseText)}</div>`;

        modalActionBtn.textContent = 'Finish';

        modalActionBtn.onclick = () => {

            hideModal();

            form.reset();

            courseSuggestionSection.classList.add('hidden');

        };

    } else {

        console.log('Form is invalid');

    }

});

// Handle course suggestion button click
suggestCoursesBtn.addEventListener('click', async () => {

    const studentAge = ageInput.value;

    showModal('Suggesting Courses...');

    modalContent.innerHTML = `<div class="flex justify-center items-center h-32"><div
class="spinner"></div></div>`;

    const prompt = `Based on a student's age of ${studentAge}, suggest three interesting and
relevant courses they could take. Provide a one-sentence description for each course. Format the
response as a simple list.`;

    const responseText = await callGeminiAPI(prompt);

    modalContent.innerHTML = `<div class="prose max-w-
none">${formatResponse(responseText)}</div>`;

    modalActionBtn.textContent = 'Close';

    modalActionBtn.onclick = hideModal;

});

```

```

// --- Modal Functions ---

const showModal = (title) => {
  modalTitle.textContent = title;
  geminiModal.classList.remove('hidden');
};

const hideModal = () => {
  geminiModal.classList.add('hidden');
};

modalCloseBtn.addEventListener('click', hideModal);
modalActionBtn.addEventListener('click', hideModal); // Default action

// --- Gemini API Call ---

async function callGeminiAPI(prompt, retries = 3, delay = 1000) {
  const apiKey = ""; // This will be handled by the environment
  const apiUrl = `https://generativelanguage.googleapis.com/v1beta/models/gemini-2.5-flash-preview-05-20:generateContent?key=${apiKey}`;
  const payload = { contents: [{ role: "user", parts: [{ text: prompt }] }] };

  try {
    const response = await fetch(apiUrl, {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(payload)
    });

    if (!response.ok) throw new Error(`HTTP error! status: ${response.status}`);

    const result = await response.json();
    const text = result.candidates?.[0]?.content?.parts?.[0]?.text;

```

```

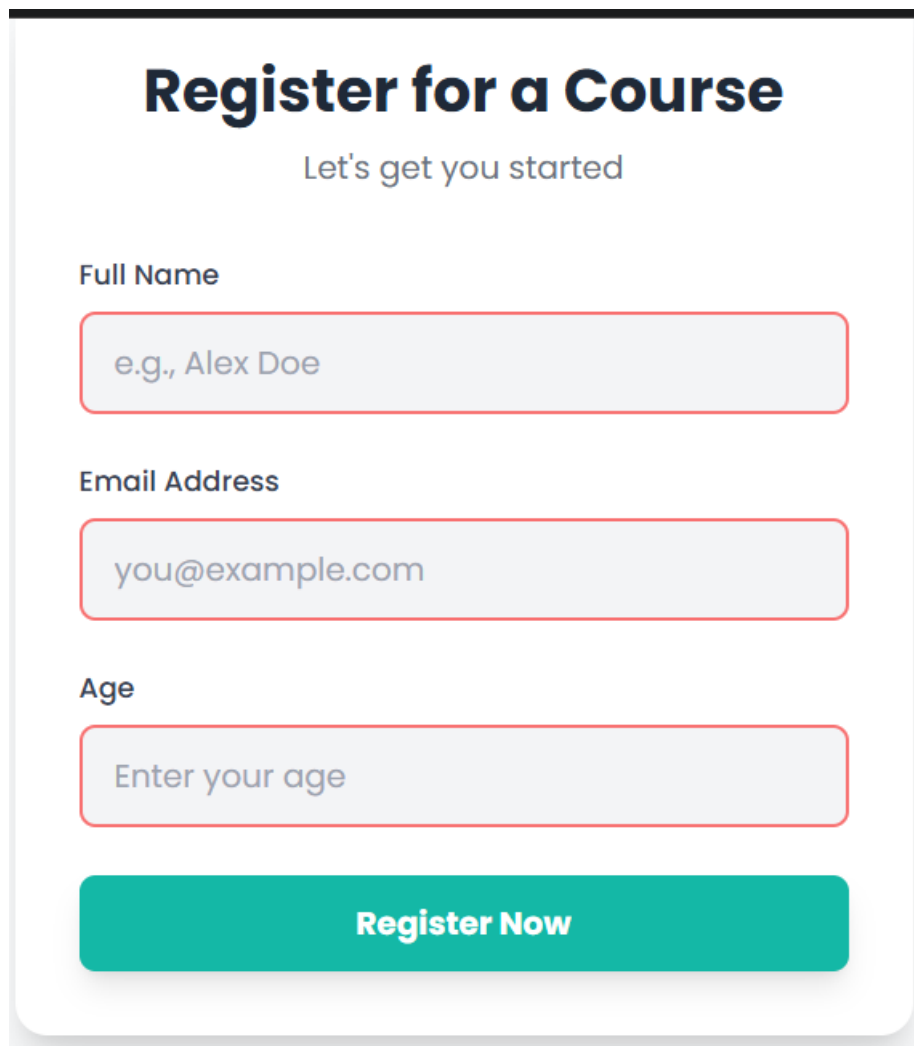
    if (text) {
        return text;
    } else {
        console.error("Unexpected API response structure:", result);
        return "Sorry, the AI returned an unexpected response format.";
    }
} catch (error) {
    console.error("Error calling Gemini API:", error);
    if (retries > 0) {
        await new Promise(res => setTimeout(res, delay));
        return callGeminiAPI(prompt, retries - 1, delay * 2); // Exponential backoff
    }
    return "An error occurred while contacting the AI. Please try again later.";
}
}

// --- Formatting Helper ---
function formatResponse(text) {
    // Basic Markdown-like to HTML conversion
    return text
        .replace(/\*\*(.*?)\*\*/g, '<strong>$1</strong>')
        .replace(/^- (.*)$/gim, '<li class="ml-4 mb-2 list-disc">$1</li>')
        .replace(/\n/g, '<br>');
}
</script>
</body>
</html>

```

OUTPUTS:

EXP-1:



Register for a Course

Let's get you started

Full Name

e.g., Alex Doe

Email Address

you@example.com

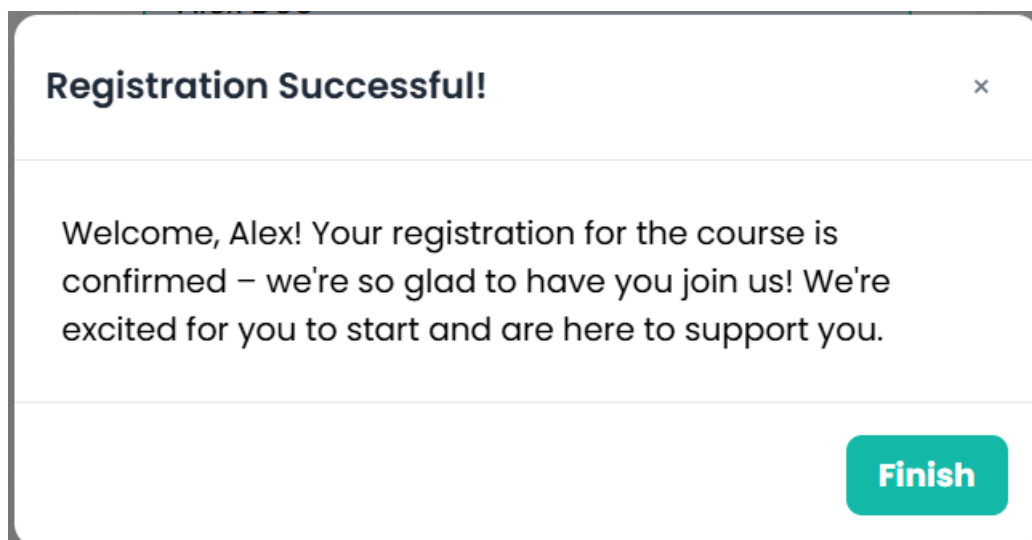
Age

Enter your age

Register Now

The registration form is a white card with rounded corners and a subtle drop shadow. It features a teal header with the title 'Register for a Course' and a subtitle 'Let's get you started'. Below this are three input fields: 'Full Name' with a placeholder 'e.g., Alex Doe', 'Email Address' with a placeholder 'you@example.com', and 'Age' with a placeholder 'Enter your age'. Each input field has a light gray background and a thin red border. At the bottom is a large teal button with the text 'Register Now' in white.

Figure 1 Registration



Registration Successful! x

Welcome, Alex! Your registration for the course is confirmed – we're so glad to have you join us! We're excited for you to start and are here to support you.

Finish

The confirmation message is a white card with rounded corners and a subtle drop shadow. It has a teal header with the title 'Registration Successful!' and a close button 'x' in the top right corner. The main body contains a welcome message: 'Welcome, Alex! Your registration for the course is confirmed – we're so glad to have you join us! We're excited for you to start and are here to support you.' At the bottom right is a teal button with the text 'Finish' in white.

Figure 2 Registration Successful

