<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>LEAPS 2.0 Chatbot</title>

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

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

<style>

body {

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

background-color: #f0f2f5;

display: flex;

justify-content: center;

align-items: center;

min-height: 100vh;

margin: 0;

}

.chat-container {

background-color: #ffffff;

border-radius: 1rem;

box-shadow: 0 10px 25px rgba(0, 0, 0, 0.1);

width: 100%;

max-width: 600px;

display: flex;

flex-direction: column;

overflow: hidden;

height: 80vh; /\* Set a fixed height for the chat window \*/

}

.chat-header {

background-color: navy; /\* Changed to navy blue \*/

color: white;

padding: 1rem;

text-align: center;

font-size: 1.25rem;

font-weight: 600;

border-top-left-radius: 1rem;

border-top-right-radius: 1rem;

display: flex; /\* Use flexbox for alignment \*/

align-items: center; /\* Vertically align items \*/

justify-content: center; /\* Center content horizontally \*/

}

.chat-messages {

flex-grow: 1;

padding: 1rem;

overflow-y: auto;

display: flex;

flex-direction: column;

gap: 0.75rem;

background-color: #87ceeb; /\* Light sky blue background \*/

/\* Background icons using SVG data URIs \*/

background-image:

url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg' width='100' height='100'%3E%3Ctext x='10' y='30' font-size='20' opacity='0.08'%3E%F0%9F%8E%B5%3C/text%3E%3Ctext x='40' y='60' font-size='20' opacity='0.08'%3E%E2%9A%BD%EF%B8%8F%3C/text%3E%3Ctext x='70' y='90' font-size='20' opacity='0.08'%3E%F0%9F%92%BB%3C/text%3E%3Ctext x='5' y='80' font-size='20' opacity='0.08'%3E%F0%9F%9B%A1%EF%B8%8F%3C/text%3E%3C/svg%3E"); /\* Added shield icon \*/

background-repeat: repeat;

background-size: 150px 150px; /\* Adjust size to control density of the pattern \*/

background-position: 0 0; /\* Start from top-left \*/

}

.message {

max-width: 80%;

padding: 0.75rem 1rem;

border-radius: 0.75rem;

word-wrap: break-word;

}

.user-message {

align-self: flex-end;

background-color: #dbeafe;

color: #1e40af;

border-bottom-right-radius: 0.25rem;

}

.bot-message {

align-self: flex-start;

background-color: #ffffff;

color: #333;

border-bottom-left-radius: 0.25rem;

box-shadow: 0 2px 5px rgba(0, 0, 0, 0.05);

}

.chat-input-container {

display: flex;

flex-wrap: wrap; /\* Allow wrapping for buttons \*/

padding: 1rem;

border-top: 1px solid #e2e8f0;

background-color: #ffffff;

border-bottom-left-radius: 1rem;

border-bottom-right-radius: 1rem;

}

.chat-input {

flex-grow: 1;

padding: 0.75rem 1rem;

border: 1px solid #cbd5e1;

border-radius: 0.5rem;

font-size: 1rem;

outline: none;

transition: border-color 0.2s;

margin-bottom: 0.5rem; /\* Space below input \*/

}

.chat-input:focus {

border-color: #4f46e5;

}

.button-group {

display: flex;

width: 100%; /\* Take full width \*/

justify-content: flex-end; /\* Align buttons to the right \*/

gap: 0.5rem; /\* Space between buttons \*/

}

.send-button, .llm-feature-button {

background-color: #4f46e5;

color: white;

padding: 0.75rem 1.25rem;

border-radius: 0.5rem;

cursor: pointer;

transition: background-color 0.2s;

border: none;

font-weight: 600;

flex-shrink: 0; /\* Prevent buttons from shrinking \*/

}

.send-button:hover, .llm-feature-button:hover {

background-color: #4338ca;

}

.loading-indicator {

align-self: flex-start;

background-color: #ffffff;

color: #333;

padding: 0.75rem 1rem;

border-radius: 0.75rem;

border-bottom-left-radius: 0.25rem;

box-shadow: 0 2px 5px rgba(0, 0, 0, 0.05);

}

</style>

</head>

<body>

<div class="chat-container">

<div class="chat-header">

Ask Alvin Anything about LEAPS

</div>

<div class="chat-messages" id="chat-messages">

<div class="bot-message">Hello! I'm Alvin. What would you like to know about CCA LEAPS 2.0?</div>

</div>

<div class="chat-input-container">

<input type="text" id="user-input" class="chat-input" placeholder="Ask a question or enter a domain/interest...">

<div class="button-group">

<button id="summarize-domain-button" class="llm-feature-button">Summarize Domain ✨</button>

<button id="suggest-cca-button" class="llm-feature-button">Suggest CCA Ideas ✨</button>

<button id="send-button" class="send-button">Send</button>

</div>

</div>

</div>

<script type="module">

const chatMessages = document.getElementById('chat-messages');

const userInput = document.getElementById('user-input');

const sendButton = document.getElementById('send-button');

const summarizeDomainButton = document.getElementById('summarize-domain-button');

const suggestCcaButton = document.getElementById('suggest-cca-button');

// Function to add a message to the chat

function addMessage(text, sender) {

const messageDiv = document.createElement('div');

messageDiv.classList.add('message');

messageDiv.classList.add(sender === 'user' ? 'user-message' : 'bot-message');

messageDiv.innerHTML = text; // Use innerHTML to render bullet points

chatMessages.appendChild(messageDiv);

chatMessages.scrollTop = chatMessages.scrollHeight; // Scroll to the bottom

}

// Function to show a loading indicator

function showLoadingIndicator() {

const loadingDiv = document.createElement('div');

loadingDiv.classList.add('loading-indicator');

loadingDiv.id = 'loading-indicator';

loadingDiv.textContent = 'Typing...';

chatMessages.appendChild(loadingDiv);

chatMessages.scrollTop = chatMessages.scrollHeight;

}

// Function to remove the loading indicator

function removeLoadingIndicator() {

const loadingDiv = document.getElementById('loading-indicator');

if (loadingDiv) {

loadingDiv.remove();

}

}

// Function to send a message to the LLM

async function sendMessageToLLM(prompt) {

showLoadingIndicator(); // Show loading indicator

let chatHistory = [];

// Add instructions to the prompt to limit the LLM's knowledge base and response length

const fullPrompt = `Using ONLY the information from the provided MOE document titled "LEAPS 2.0 (School Detailed Version).pdf" and the MOE website link (Co-curricular activities (CCA) | MOE), answer the following question. Do NOT include any source file details or mention the document/website. Ensure that if you provide a list, each item is a bullet point on a new line. Limit your response to 2-3 key pointers. Focus solely on details relevant to Singapore's LEAPS 2.0 and Co-curricular achievements.

Question: ${prompt}`;

chatHistory.push({ role: "user", parts: [{ text: fullPrompt }] });

const payload = { contents: chatHistory };

const apiKey = ""; // If you want to use models other than gemini-2.5-flash-preview-05-20 or imagen-3.0-generate-002, provide an API key here. Otherwise, leave this as-is.

const apiUrl = `https://generativelanguage.googleapis.com/v1beta/models/gemini-2.5-flash-preview-05-20:generateContent?key=${apiKey}`;

// Implement exponential backoff for API calls

let retries = 0;

const maxRetries = 5;

const initialDelay = 1000; // 1 second

while (retries < maxRetries) {

try {

const response = await fetch(apiUrl, {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify(payload)

});

if (!response.ok) {

// If response is not OK, throw an error to trigger retry

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

}

const result = await response.json();

removeLoadingIndicator(); // Remove loading indicator on successful response

if (result.candidates && result.candidates.length > 0 &&

result.candidates[0].content && result.candidates[0].content.parts &&

result.candidates[0].content.parts.length > 0) {

const text = result.candidates[0].content.parts[0].text;

// Replace markdown bullet points with HTML list items for display

const formattedText = text.replace(/^- /gm, '• ').replace(/\n• /g, '<br>• ');

addMessage(formattedText, 'bot');

} else {

addMessage("I'm sorry, I couldn't get a clear response. Please try again.", 'bot');

}

return; // Exit loop on success

} catch (error) {

retries++;

if (retries < maxRetries) {

const delay = initialDelay \* Math.pow(2, retries - 1);

await new Promise(resolve => setTimeout(resolve, delay));

} else {

removeLoadingIndicator(); // Remove loading indicator on final failure

addMessage("I'm sorry, I'm having trouble connecting right now. Please try again later.", 'bot');

}

}

}

}

// Event listener for the main send button

sendButton.addEventListener('click', () => {

const question = userInput.value.trim();

if (question) {

addMessage(question, 'user');

sendMessageToLLM(question);

userInput.value = ''; // Clear input

}

});

// Event listener for the "Summarize Domain" button

summarizeDomainButton.addEventListener('click', () => {

const domain = userInput.value.trim();

if (domain) {

addMessage(`Summarize the ${domain} domain.`, 'user');

const prompt = `Summarize the '${domain}' domain of LEAPS 2.0.`;

sendMessageToLLM(prompt);

userInput.value = ''; // Clear input

} else {

addMessage("Please enter a domain (e.g., Participation, Leadership) to summarize.", 'bot');

}

});

// Event listener for the "Suggest CCA Ideas" button

suggestCcaButton.addEventListener('click', () => {

const interests = userInput.value.trim();

if (interests) {

addMessage(`Suggest CCA ideas for interests: ${interests}.`, 'user');

const prompt = `Suggest CCA types under LEAPS 2.0 that align with the following interests: '${interests}'.`;

sendMessageToLLM(prompt);

userInput.value = ''; // Clear input

} else {

addMessage("Please enter your child's interests to get CCA suggestions.", 'bot');

}

});

// Event listener for pressing Enter in the input field

userInput.addEventListener('keypress', (event) => {

if (event.key === 'Enter') {

sendButton.click();

}

});

</script>

</body>

</html>