<!DOCTYPE html>

<!-- Coding By JV Codes - youtube.com/@jvcodes -->

<html lang="en">

  <head>

    <meta charset="UTF-8" />

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

    <title>AI Chatbot | JVCodes</title>

    <!-- Linking Google fonts for icons -->

    <link rel="stylesheet" href="https://fonts.googleapis.com/css2?family=Material+Symbols+Outlined:opsz,wght,FILL,GRAD@48,400,0,0&family=Material+Symbols+Rounded:opsz,wght,FILL,GRAD@48,400,1,0" />

</head>

<style>

    /\* Importing Google Fonts - Inter \*/

@import url('https://fonts.googleapis.com/css2?family=Inter:opsz,wght@14..32,100..900&display=swap');

\* {

  margin: 0;

  padding: 0;

  box-sizing: border-box;

  font-family: "Inter", sans-serif;

}

body {

  width: 100%;

  min-height: 100vh;

  background: linear-gradient(#fdb99f, #ff5e14);

}

#chatbot-toggler {

  position: fixed;

  bottom: 30px;

  right: 35px;

  border: none;

  height: 50px;

  width: 50px;

  display: flex;

  cursor: pointer;

  align-items: center;

  justify-content: center;

  border-radius: 50%;

  background: #00abb1;

  box-shadow: 0 0 20px rgba(0, 0, 0, 0.1);

  transition: all 0.2s ease;

}

body.show-chatbot #chatbot-toggler {

  transform: rotate(90deg);

}

#chatbot-toggler span {

  color: #fff;

  position: absolute;

}

#chatbot-toggler span:last-child,

body.show-chatbot #chatbot-toggler span:first-child {

  opacity: 0;

}

body.show-chatbot #chatbot-toggler span:last-child {

  opacity: 1;

}

.chatbot-popup {

  position: fixed;

  right: 35px;

  bottom: 90px;

  width: 420px;

  overflow: hidden;

  background: #ececec;

  border-radius: 15px;

  opacity: 0;

  pointer-events: none;

  transform: scale(0.2);

  transform-origin: bottom right;

  box-shadow: 0 0 128px 0 rgba(0, 0, 0, 0.1),

    0 32px 64px -48px rgba(0, 0, 0, 0.5);

  transition: all 0.1s ease;

}

body.show-chatbot .chatbot-popup {

  opacity: 1;

  pointer-events: auto;

  transform: scale(1);

}

.chat-header {

  display: flex;

  align-items: center;

  padding: 15px 22px;

  background: #00abb1;

  justify-content: space-between;

}

.chat-header .header-info {

  display: flex;

  gap: 10px;

  align-items: center;

}

.header-info .chatbot-logo {

  width: 35px;

  height: 35px;

  padding: 6px;

  fill: #00abb1;

  flex-shrink: 0;

  background: #fff;

  border-radius: 50%;

}

.header-info .logo-text {

  color: #fff;

  font-weight: 600;

  font-size: 1.31rem;

  letter-spacing: 0.02rem;

}

.chat-header #close-chatbot {

  border: none;

  color: #fff;

  height: 40px;

  width: 40px;

  font-size: 1.9rem;

  margin-right: -10px;

  padding-top: 2px;

  cursor: pointer;

  border-radius: 50%;

  background: none;

  transition: 0.2s ease;

}

.chat-header #close-chatbot:hover {

  background: #005f63;

}

.chat-body {

  padding: 25px 22px;

  gap: 20px;

  display: flex;

  height: 460px;

  overflow-y: auto;

  margin-bottom: 82px;

  flex-direction: column;

  scrollbar-width: thin;

  scrollbar-color: #b9fdff transparent;

}

.chat-body,

.chat-form .message-input:hover {

  scrollbar-color: #b9fdff transparent;

}

.chat-body .message {

  display: flex;

  gap: 11px;

  align-items: center;

}

.chat-body .message .bot-avatar {

  width: 35px;

  height: 35px;

  padding: 6px;

  fill: #fff;

  flex-shrink: 0;

  margin-bottom: 2px;

  align-self: flex-end;

  border-radius: 50%;

  background: #ffba6a;

}

.chat-body .message .message-text {

  padding: 12px 16px;

  max-width: 75%;

  font-size: 0.95rem;

}

.chat-body .bot-message.thinking .message-text {

  padding: 2px 16px;

}

.chat-body .bot-message .message-text {

  background: #ececec;

  border-radius: 13px 13px 13px 3px;

}

.chat-body .user-message {

  flex-direction: column;

  align-items: flex-end;

}

.chat-body .user-message .message-text {

  color: #fff;

  background: #00abb1;

  border-radius: 13px 13px 3px 13px;

}

.chat-body .user-message .attachment {

  width: 50%;

  margin-top: -7px;

  border-radius: 13px 3px 13px 13px;

}

.chat-body .bot-message .thinking-indicator {

  display: flex;

  gap: 4px;

  padding-block: 15px;

}

.chat-body .bot-message .thinking-indicator .dot {

  height: 7px;

  width: 7px;

  opacity: 0.7;

  border-radius: 50%;

  background: #00abb1;

  animation: dotPulse 1.8s ease-in-out infinite;

}

.chat-body .bot-message .thinking-indicator .dot:nth-child(1) {

  animation-delay: 0.2s;

}

.chat-body .bot-message .thinking-indicator .dot:nth-child(2) {

  animation-delay: 0.3s;

}

.chat-body .bot-message .thinking-indicator .dot:nth-child(3) {

  animation-delay: 0.4s;

}

@keyframes dotPulse {

  0%,

  44% {

    transform: translateY(0);

  }

  28% {

    opacity: 0.4;

    transform: translateY(-4px);

  }

  44% {

    opacity: 0.2;

  }

}

.chat-footer {

  position: absolute;

  bottom: 0;

  width: 100%;

  background: #ececec;

  padding: 15px 22px 20px;

}

.chat-footer .chat-form {

  display: flex;

  align-items: center;

  position: relative;

  background: #fff;

  border-radius: 32px;

  outline: 1px solid #CCCCE5;

  box-shadow: 0 0 8px rgba(0, 0, 0, 0.06);

  transition: 0s ease, border-radius 0s;

}

.chat-form:focus-within {

  outline: 2px solid #00abb1;

}

.chat-form .message-input {

  width: 100%;

  height: 47px;

  outline: none;

  resize: none;

  border: none;

  max-height: 180px;

  scrollbar-width: thin;

  border-radius: inherit;

  font-size: 0.95rem;

  padding: 14px 0 12px 18px;

  scrollbar-color: transparent transparent;

}

.chat-form .chat-controls {

  gap: 3px;

  height: 47px;

  display: flex;

  padding-right: 6px;

  align-items: center;

  align-self: flex-end;

}

.chat-form .chat-controls button {

  height: 35px;

  width: 35px;

  border: none;

  cursor: pointer;

  color: #00abb1;

  border-radius: 50%;

  font-size: 1.15rem;

  background: none;

  transition: 0.2s ease;

}

.chat-form .chat-controls button:hover,

body.show-emoji-picker .chat-controls #emoji-picker {

  color: #00abb1;

  background: #f1f1ff;

}

.chat-form .chat-controls #send-message {

  color: #fff;

  display: none;

  background: #00abb1;

}

.chat-form .chat-controls #send-message:hover {

  background: #00abb1;

}

.chat-form .message-input:valid~.chat-controls #send-message {

  display: block;

}

.chat-form .file-upload-wrapper {

  position: relative;

  height: 35px;

  width: 35px;

}

.chat-form .file-upload-wrapper :where(button, img) {

  position: absolute;

}

.chat-form .file-upload-wrapper img {

  height: 100%;

  width: 100%;

  object-fit: cover;

  border-radius: 50%;

}

.chat-form .file-upload-wrapper #file-cancel {

  color: #ff0000;

  background: #fff;

}

.chat-form .file-upload-wrapper :where(img, #file-cancel),

.chat-form .file-upload-wrapper.file-uploaded #file-upload {

  display: none;

}

.chat-form .file-upload-wrapper.file-uploaded img,

.chat-form .file-upload-wrapper.file-uploaded:hover #file-cancel {

  display: block;

}

em-emoji-picker {

  position: absolute;

  left: 50%;

  top: -337px;

  width: 100%;

  max-width: 350px;

  visibility: hidden;

  max-height: 330px;

  transform: translateX(-50%);

}

body.show-emoji-picker em-emoji-picker {

  visibility: visible;

}

/\* Responsive media query for mobile screens \*/

@media (max-width: 520px) {

  #chatbot-toggler {

    right: 20px;

    bottom: 20px;

  }

  .chatbot-popup {

    right: 0;

    bottom: 0;

    height: 100%;

    border-radius: 0;

    width: 100%;

  }

  .chatbot-popup .chat-header {

    padding: 12px 15px;

  }

  .chat-body {

    height: calc(90% - 55px);

    padding: 25px 15px;

  }

  .chat-footer {

    padding: 10px 15px 15px;

  }

  .chat-form .file-upload-wrapper.file-uploaded #file-cancel {

    opacity: 0;

  }

}

</style>

<body>

  <!-- Chatbot Toggler -->

  <button id="chatbot-toggler">

    <span class="material-symbols-rounded">mode\_comment</span>

    <span class="material-symbols-rounded">close</span>

  </button>

  <div class="chatbot-popup">

    <!-- Chatbot Header -->

    <div class="chat-header">

      <div class="header-info">

        <img class="chatbot-logo" src="client-2.jpg" alt="Chatbot Logo" width="50" height="50">

        </img>

        <h2 class="logo-text">Chatbot</h2>

      </div>

      <button id="close-chatbot" class="material-symbols-rounded">keyboard\_arrow\_down</button>

    </div>

    <!-- Chatbot Body -->

    <div class="chat-body">

      <div class="message bot-message">

        <img class="bot-avatar" src="robotic.png" alt="Chatbot Logo" width="50" height="50">

        </img>

        <!-- prettier-ignore -->

        <div class="message-text"> Hey there  <br /> How can I help you today? </div>

      </div>

    </div>

    <!-- Chatbot Footer -->

    <div class="chat-footer">

      <form action="#" class="chat-form">

        <textarea placeholder="Message..." class="message-input" required></textarea>

        <div class="chat-controls">

          <button type="button" id="emoji-picker" class="material-symbols-outlined">sentiment\_satisfied</button>

          <div class="file-upload-wrapper">

            <input type="file" accept="image/\*" id="file-input" hidden />

            <img src="#" />

            <button type="button" id="file-upload" class="material-symbols-rounded">attach\_file</button>

            <button type="button" id="file-cancel" class="material-symbols-rounded">close</button>

          </div>

          <button type="submit" id="send-message" class="material-symbols-rounded">arrow\_upward</button>

        </div>

      </form>

    </div>

  </div>

  <!-- Linking Emoji Mart script for emoji picker -->

  <script src="https://cdn.jsdelivr.net/npm/emoji-mart@latest/dist/browser.js"></script>

  <!-- Linking custom script -->

  <script>

    const chatBody = document.querySelector(".chat-body");

const messageInput = document.querySelector(".message-input");

const sendMessage = document.querySelector("#send-message");

const fileInput = document.querySelector("#file-input");

const fileUploadWrapper = document.querySelector(".file-upload-wrapper");

const fileCancelButton = fileUploadWrapper.querySelector("#file-cancel");

const chatbotToggler = document.querySelector("#chatbot-toggler");

const closeChatbot = document.querySelector("#close-chatbot");

// API setup

const API\_KEY = "AIzaSyCQXdM8mF1o7j7KlC2ue75X37ZIU\_cDTVk";

const API\_URL = `https://generativelanguage.googleapis.com/v1/models/gemini-1.5-pro:generateContent?key=${API\_KEY}`;

// Initialize user message and file data

const userData = {

  message: null,

  file: {

    data: null,

    mime\_type: null,

  },

};

// Store chat history

const chatHistory = [];

const initialInputHeight = messageInput.scrollHeight;

// Create message element with dynamic classes and return it

const createMessageElement = (content, ...classes) => {

  const div = document.createElement("div");

  div.classList.add("message", ...classes);

  div.innerHTML = content;

  return div;

};

// Generate bot response using API

const generateBotResponse = async (incomingMessageDiv) => {

  const messageElement = incomingMessageDiv.querySelector(".message-text");

  // Add user message to chat history

  chatHistory.push({

    role: "user",

    parts: [{ text: userData.message }, ...(userData.file.data ? [{ inline\_data: userData.file }] : [])],

  });

  // API request options

  const requestOptions = {

    method: "POST",

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

    body: JSON.stringify({

      contents: chatHistory,

    }),

  };

  try {

    // Fetch bot response from API

    const response = await fetch(API\_URL, requestOptions);

    const data = await response.json();

    if (!response.ok) throw new Error(data.error.message);

    // Extract and display bot's response text

    const apiResponseText = data.candidates[0].content.parts[0].text.replace(/\\*\\*(.\*?)\\*\\*/g, "$1").trim();

    messageElement.innerText = apiResponseText;

    // Add bot response to chat history

    chatHistory.push({

      role: "model",

      parts: [{ text: apiResponseText }],

    });

  } catch (error) {

    // Handle error in API response

    console.log(error);

    messageElement.innerText = error.message;

    messageElement.style.color = "#ff0000";

  } finally {

    // Reset user's file data, removing thinking indicator and scroll chat to bottom

    userData.file = {};

    incomingMessageDiv.classList.remove("thinking");

    chatBody.scrollTo({ top: chatBody.scrollHeight, behavior: "smooth" });

  }

};

// Handle outgoing user messages

const handleOutgoingMessage = (e) => {

  e.preventDefault();

  userData.message = messageInput.value.trim();

  messageInput.value = "";

  messageInput.dispatchEvent(new Event("input"));

  fileUploadWrapper.classList.remove("file-uploaded");

  // Create and display user message

  const messageContent = `<div class="message-text"></div>

                          ${userData.file.data ? `<img src="data:${userData.file.mime\_type};base64,${userData.file.data}" class="attachment" />` : ""}`;

  const outgoingMessageDiv = createMessageElement(messageContent, "user-message");

  outgoingMessageDiv.querySelector(".message-text").innerText = userData.message;

  chatBody.appendChild(outgoingMessageDiv);

  chatBody.scrollTo({ top: chatBody.scrollHeight, behavior: "smooth" });

  // Simulate bot response with thinking indicator after a delay

  setTimeout(() => {

    const messageContent = `<img class="bot-avatar" src="robotic.png" alt="Chatbot Logo" width="50" height="50">

          </img>

          <div class="message-text">

            <div class="thinking-indicator">

              <div class="dot"></div>

              <div class="dot"></div>

              <div class="dot"></div>

            </div>

          </div>`;

    const incomingMessageDiv = createMessageElement(messageContent, "bot-message", "thinking");

    chatBody.appendChild(incomingMessageDiv);

    chatBody.scrollTo({ top: chatBody.scrollHeight, behavior: "smooth" });

    generateBotResponse(incomingMessageDiv);

  }, 600);

};

// Adjust input field height dynamically

messageInput.addEventListener("input", () => {

  messageInput.style.height = `${initialInputHeight}px`;

  messageInput.style.height = `${messageInput.scrollHeight}px`;

  document.querySelector(".chat-form").style.borderRadius = messageInput.scrollHeight > initialInputHeight ? "15px" : "32px";

});

// Handle Enter key press for sending messages

messageInput.addEventListener("keydown", (e) => {

  const userMessage = e.target.value.trim();

  if (e.key === "Enter" && !e.shiftKey && userMessage && window.innerWidth > 768) {

    handleOutgoingMessage(e);

  }

});

// Handle file input change and preview the selected file

fileInput.addEventListener("change", () => {

  const file = fileInput.files[0];

  if (!file) return;

  const reader = new FileReader();

  reader.onload = (e) => {

    fileInput.value = "";

    fileUploadWrapper.querySelector("img").src = e.target.result;

    fileUploadWrapper.classList.add("file-uploaded");

    const base64String = e.target.result.split(",")[1];

    // Store file data in userData

    userData.file = {

      data: base64String,

      mime\_type: file.type,

    };

  };

  reader.readAsDataURL(file);

});

// Cancel file upload

fileCancelButton.addEventListener("click", () => {

  userData.file = {};

  fileUploadWrapper.classList.remove("file-uploaded");

});

// Initialize emoji picker and handle emoji selection

const picker = new EmojiMart.Picker({

  theme: "light",

  skinTonePosition: "none",

  previewPosition: "none",

  onEmojiSelect: (emoji) => {

    const { selectionStart: start, selectionEnd: end } = messageInput;

    messageInput.setRangeText(emoji.native, start, end, "end");

    messageInput.focus();

  },

  onClickOutside: (e) => {

    if (e.target.id === "emoji-picker") {

      document.body.classList.toggle("show-emoji-picker");

    } else {

      document.body.classList.remove("show-emoji-picker");

    }

  },

});

document.querySelector(".chat-form").appendChild(picker);

sendMessage.addEventListener("click", (e) => handleOutgoingMessage(e));

document.querySelector("#file-upload").addEventListener("click", () => fileInput.click());

closeChatbot.addEventListener("click", () => document.body.classList.remove("show-chatbot"));

chatbotToggler.addEventListener("click", () => document.body.classList.toggle("show-chatbot"));

  </script>

</body>

</html>