```
// Initial chat state with the first message from the bot
const messages = [
  {
    sender: "bot",
    message: "Hi there! How can I help you today?",
  },
];

// Select necessary DOM elements for the chat interface
const chatBody = document.querySelector(".chat-body");
const chatFooter = document.querySelector(".chat-footer");
const input = document.querySelector("#input-message");
const sendButton = document.querySelector("#send-btn");

// Initial render of messages
renderMessages();

// Function to render all messages inside the chat body
function renderMessages() {
  chatBody.innerHTML = "";
  messages.forEach((message) => {
    chatBody.innerHTML += getMessageDiv(message.sender, message.message);
  });
  chatBody.scrollTop = chatBody.scrollHeight;
}

// Function to create and return a message div for each chat message
function getMessageDiv(sender, message) {
  const htmlText = `<div class="message ${sender}">${message}</div>`;
  return htmlText;
}

// check if the user pressed the enter key
function handleKeyPress(event) {
  if (input.value !== "" && event.key === "Enter") {
    sendMessage();
  }
}

// API key for making requests to the Gemini AI API
const API_KEY = "AIzaSyDzGP3mJ3HXdYkU7VhIsaFyl6LOV-W06Vc";

// Function to send the user's message to the bot and handle the response
const sendMessage = async () => {
  // Get the message input from the user
  const message = input.value;

  // Add the user's message to the messages array
  messages.push({ sender: "user", message });

  // Render the updated messages
  renderMessages();

  // Clear the input field after sending the message
  input.value = "";

  // Add typing effect for the bot response
  typingEffect();

  // change the text of the button to "Sending..."
  sendButton.innerHTML = "Sending...";
  sendButton.disabled = true;

  // Send the user's message to the Gemini AI API
  try {
    const response = await fetch(
      `https://generativelanguage.googleapis.com/v1beta/models/gemini-1.5-
flash-latest:generateContent?key=${API_KEY}`,
      {
        method: "POST",
        headers: {
          "Content-Type": "application/json",
        },
        body: JSON.stringify({
          contents: [
            {
              role: "user",
              parts: [{ text: message }],
            },
          ],
        }),
      }
    );

    // Parse the response from the API
    const data = await response.json();

    // Extract and clean up the API response message
    const apiResponse = data.candidates[0].content.parts[0].text.replace(
      /\*\*(.*?)\*\*/g,
      "$1"
    );

    // Remove the "Typing..." placeholder and add the bot's response
    messages.pop();
    messages.push({ sender: "bot", message: apiResponse });

    // button text back to "Send"
    sendButton.innerHTML = "Send";
    sendButton.disabled = false;

    // Render the final messages after the bot's response
    renderMessages();
  } catch (error) {
    // If there is an error, log it and display a default error message
    console.error("Error fetching bot response:", error);
    messages.pop();
    messages.push({
      sender: "bot",
      message: "Sorry, there was an error processing your request.",
    });
    renderMessages();
  }
};

// Function to add a typing effect for the bot response
const typingEffect = () => {
  // Add a placeholder message for "Typing..."
  messages.push({ sender: "bot", message: "Typing..." });
  renderMessages();
};
```