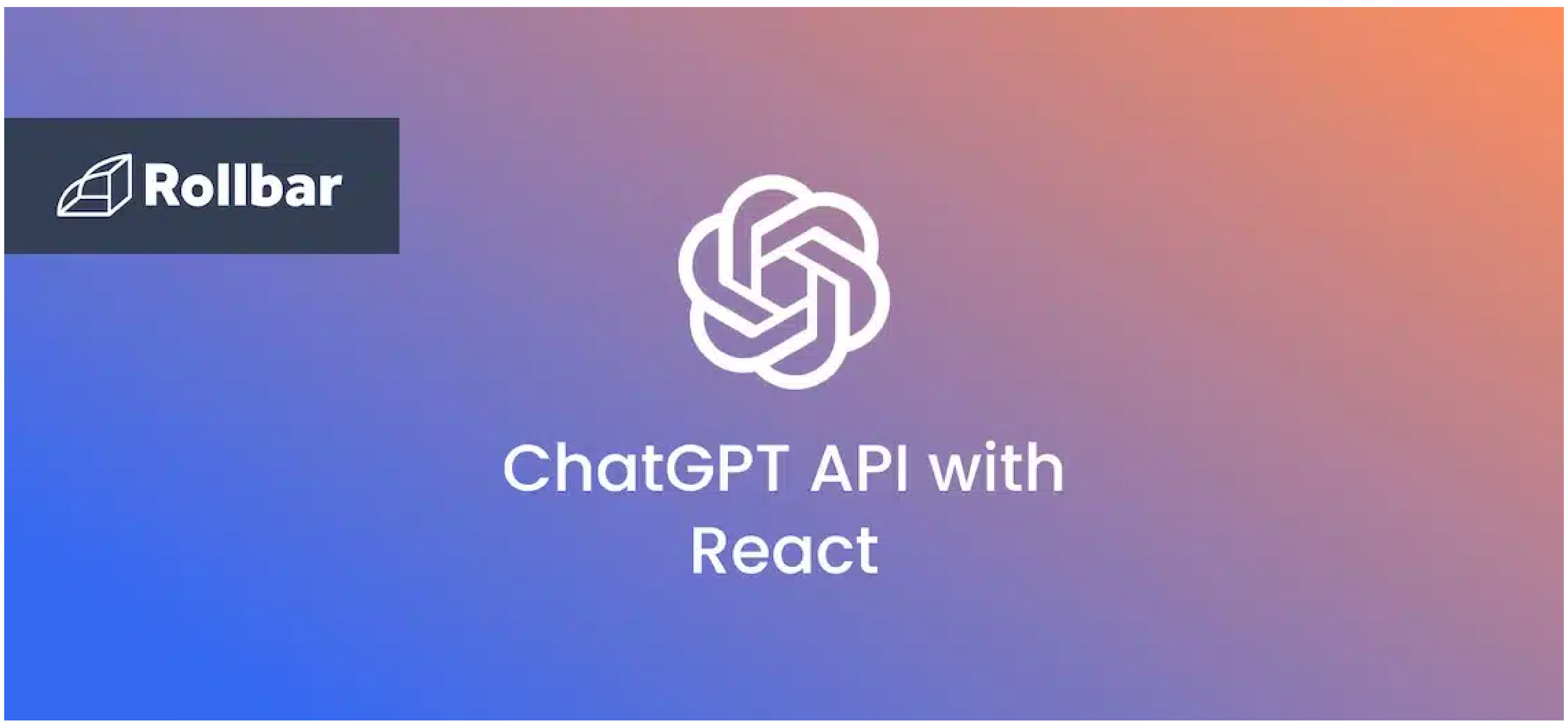
How to Integrate ChatGPT with React

- https://rollbar.com/blog/how-to-integrate-chatgpt-with-react/

- 這篇文章前半段主要描述**如何整合 ChatGPT 和 React**，這部分可以參考；
  最後一段是他在推廣他們家產品（
  Rollbar），可以用來處理 ChatGPT API 回傳的 error

- menu

If you're not thinking about integrating AI into your apps, you're missing out. In this tutorial, we will walk you through how to set up a React app that harnesses the vast knowledge of ChatGPT via the OpenAI API, allowing you to take your UI components to a whole new level.

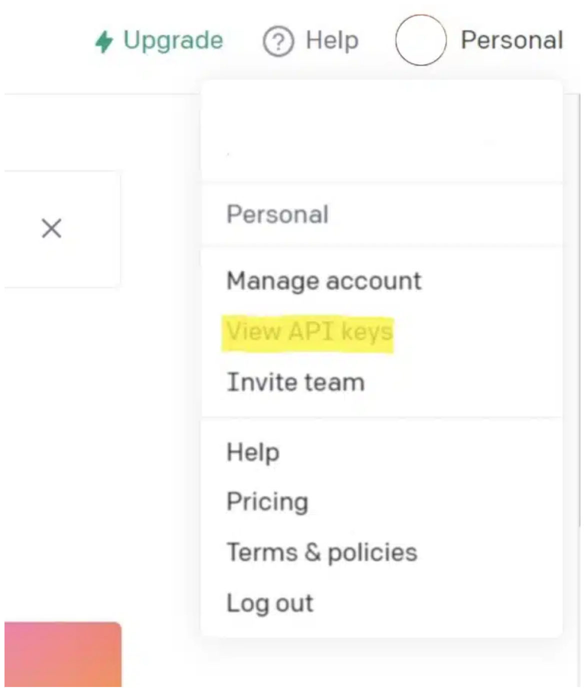
如果你還沒有考慮將人工智慧整合到你的應用程式中，那你正在錯失機會。在這個教程中，我們將指導你如何設置一個React應用程式，通過OpenAI API利用ChatGPT的龐大知識，使你的UI元件達到一個全新的水平。

Step 1: Sign up for the OpenAI API

To use the ChatGPT language model in a React app, first go to https://beta.openai.com/signup/ and sign up to get an API key that you will use to authenticate your requests.

步驟1：註冊 OpenAI API

要在React應用程式中使用ChatGPT語言模型，首先訪問 https://beta.openai.com/signup/ 並註冊以獲取一個API金鑰，你將使用它來驗證你的請求。



Sign up for the OpenAI API

Step 2: Setup the development environment

步驟2：設置開發環境

1. Create an empty folder, for instance ‘chat-gpt-app’, and open it in an IDE like VSCode.

   創建一個空文件夾，例如 'chat-gpt-app'，並在像VSCode這樣的IDE中打開它。

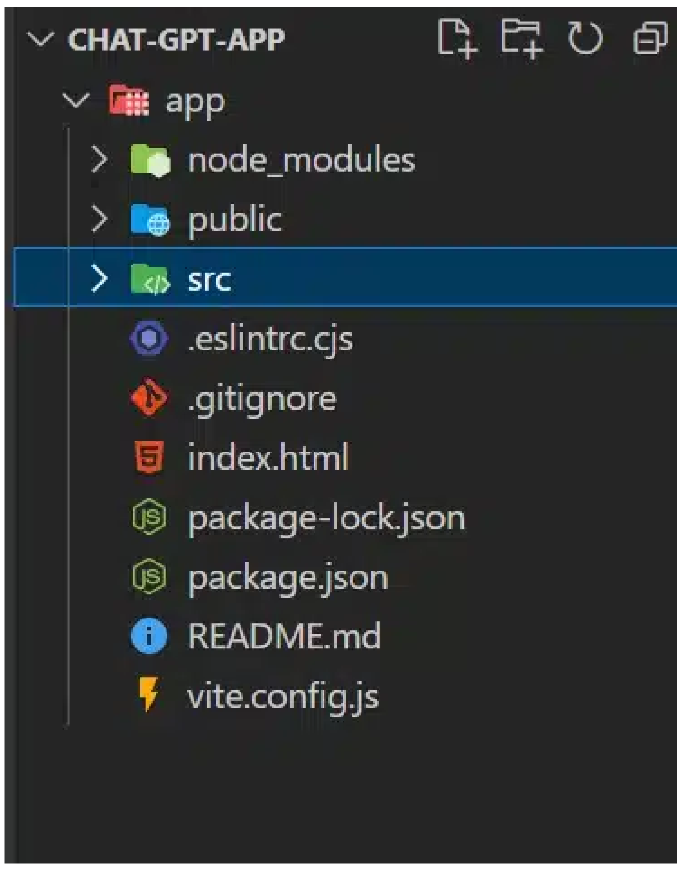
2. Now open the terminal in VSCode and type the below command to create a Vite app with a React template.

   現在在VSCode中打開終端並輸入以下命令來使用React模板創建一個Vite應用程式。

   ```
   npm create vite@latest app -- --template react
   ```

   The below folder structure will be created:

   將創建以下文件夾結構：



Setup ChatGPT development environment
設置 ChatGPT 開發環境

3. Now enter the following command in the VSCode console to access the `app` folder:

   現在在VSCode控制台中輸入以下命令以進入應用程式文件夾：

   ```
   cd app
   ```

4. Run the `npm install`.

```
npm install
```

5. To build your chat interface, you'll also need to install the chatscope UI kit.

   為了構建你的聊天界面，你還需要安裝`chatscope` UI套件。 ⇒ check npm website ✅

```
npm install@chatscope/chat-ui-kit-react
```

Step 3: Write React code to connect to the OpenAI API

步驟3：編寫React代碼以連接到OpenAI API

Now enter the following code into the `App.jsx` file located within the `src` folder:

現在輸入以下代碼到`src`文件夾中的`App.jsx`文件：

```
import { useState, useEffect }  from 'react';
import './App.css';
import '@chatscope/chat-ui-kit-styles/dist/default/styles.min.css';
import {
  MainContainer,
  ChatContainer,
  MessageList,
  Message,
  MessageInput,
  TypingIndicator,
} from '@chatscope/chat-ui-kit-react';

const API_KEY ="YOUR_API_KEY_HERE"

const App = () => {
  const [messages, setMessages] = useState([
    {
      message: "Hello, I'm ChatGPT! Ask me anything!",
      sentTime: "just now",
      sender: "ChatGPT",
    },
  ]);
  const [isTyping, setIsTyping] = useState(false);

  const handleSendRequest = async (message) => {
    const newMessage = {
      message,
      direction: 'outgoing',
      sender: "user",
    };

    setMessages((prevMessages) => [...prevMessages, newMessage]);
    setIsTyping(true);

    try {
      const response = await processMessageToChatGPT([...messages, newMessage]);
      const content = response.choices[0]?.message?.content;
      if (content) {
        const chatGPTResponse = {
          message: content,
          sender: "ChatGPT",
        };
        setMessages((prevMessages) => [...prevMessages, chatGPTResponse]);
      }
    } catch (error) {
      console.error("Error processing message:", error);
    } finally {
      setIsTyping(false);
    }
  };

  async function processMessageToChatGPT(chatMessages) {
    const apiMessages = chatMessages.map((messageObject) => {
      const role = messageObject.sender === "ChatGPT" ? "assistant" : "user";
      return { role, content: messageObject.message };
    });

    const apiRequestBody = {
      "model": "gpt-3.5-turbo",
      "messages": [
        { role: "system", content: "I'm a Student using ChatGPT for learning" },
        ...apiMessages,
      ],
    };

    const response = await fetch("https://api.openai.com/v1/chat/completions", {
      method: "POST",
      headers: {
        "Authorization": "Bearer " + API_KEY,
        "Content-Type": "application/json",
      },
      body: JSON.stringify(apiRequestBody),
    });

    return response.json();
  }

  return (
    <div className="App">
      <div style={{ position:"relative", height: "800px", width: "700px"  }}>
        <MainContainer>
          <ChatContainer>
            <MessageList
              scrollBehavior="smooth"
              typingIndicator={isTyping ? <TypingIndicator content="ChatGPT is typing" /
            >
              {messages.map((message, i) => {
                console.log(message)
                return <Message key={i} model={message} />
              })}
            </MessageList>
            <MessageInput placeholder="Send a Message" onSend={handleSendRequest} />
          </ChatContainer>
        </MainContainer>
      </div>
    </div>
  )
}

export default App;
```

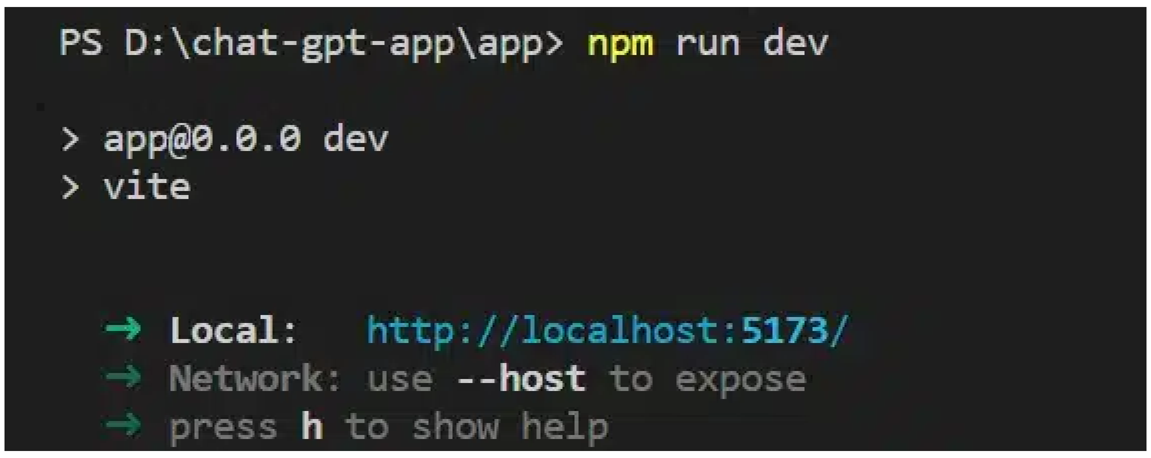
Step 4: Open your app and start chatting

步驟4：打開你的應用程式並開始聊天

1. Now to execute the code, type the below command in terminal:

   現在執行代碼，請在終端中輸入以下命令：

   ```
   npm run dev
   ```

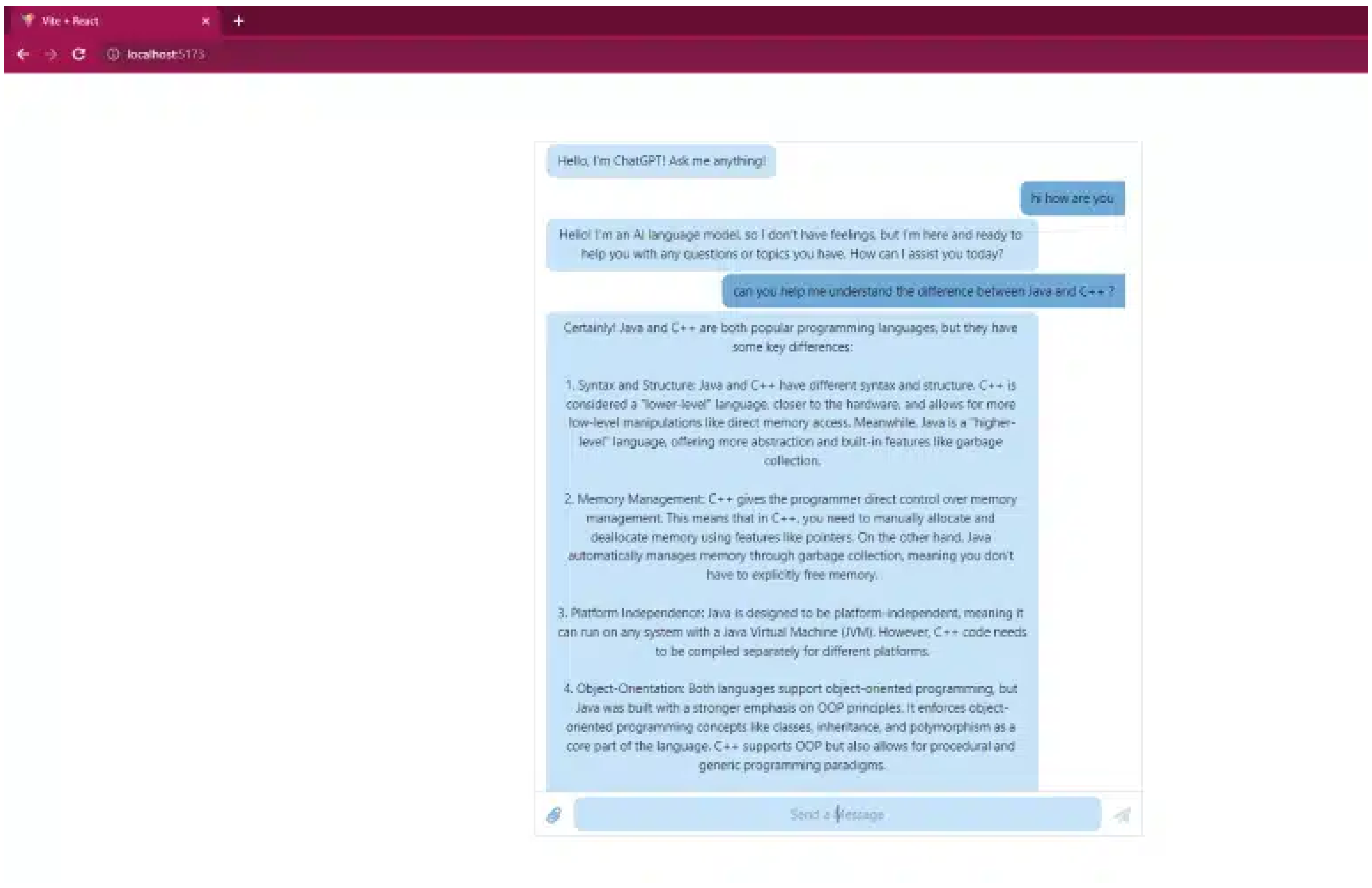


```
PS D:\chat-gpt-app\app> npm run dev

> app@0.0.0 dev
> vite


  ➜  Local:   http://localhost:5173/
  ➜  Network: use --host to expose
  ➜  press h to show help
```

Open app and run ChatGPT
打開應用程式並運行ChatGPT。

2. Now you can just go to the above localhost and see your chat interface, where you can make conversations with the ChatGPT model.

   現在你可以訪問上述的本地主機，看到你的聊天界面，你可以與ChatGPT模型進行對話。



Hello, I'm ChatGPT! Ask me anything!

hi how are you

Hello! I'm an AI language model, so I don't have feelings, but I'm here and ready to help you with any questions or topics you have. How can I assist you today?

can you help me understand the difference between Java and C++ ?

Certainly! Java and C++ are both popular programming languages, but they have some key differences:

1. Syntax and Structure: Java and C++ have different syntax and structure. C++ is considered a "lower-level" language, closer to the hardware, and allows for more low-level manipulations like direct memory access. Meanwhile, Java is a "higher-level" language, offering more abstraction and built-in features like garbage collection.

2. Memory Management: C++ gives the programmer direct control over memory management. This means that in C++, you need to manually allocate and deallocate memory using features like pointers. On the other hand, Java automatically manages memory through garbage collection, meaning you don't have to explicitly free memory.

3. Platform Independence: Java is designed to be platform-independent, meaning it can run on any system with a Java Virtual Machine (JVM). However, C++ code needs to be compiled separately for different platforms.

4. Object-Orientation: Both languages support object-oriented programming, but Java was built with a stronger emphasis on OOP principles. It enforces object-oriented programming concepts like classes, inheritance, and polymorphism as a core part of the language. C++ supports OOP but also allows for procedural and generic programming paradigms.

Localhost ChatGPT interface
本地主機 ChatGPT 介面

Step-by-step guide explaining the React code

逐步指南解釋 React 代碼

1. **Import Statements:** Import the necessary hooks to handle the states and `@chatscope/chat-ui-kit-styles` and `@chatscope/chat-ui-kit-react` third-party libraries for styling and UI components of the chat interface.

   導入語句： 導入處理狀態的必要鉤子，以及 `@chatscope/chat-ui-kit-styles` 和 `@chatscope/chat-ui-kit-react` 第三方庫，用於聊天界面的樣式和UI組件。

2. **State Initialization:** The component initializes state using the `useState` hook:

   狀態初始化： 這個組件使用 `useState` 鉤子來初始化狀態：

- `messages`:

  An array of messages representing the chat conversation. It starts with an initial message from "ChatGPT."

  表示聊天對話的訊息陣列。它以一條來自 "ChatGPT" 的初始訊息開始。

- `isTyping`:

  A boolean state to indicate whether the AI ("ChatGPT") is typing a response.

  一個布林值狀態，表示AI（"ChatGPT"）是否正在輸入回應。

3. **handleSendRequest Function:**
   - This function is called when the user sends a message.

     當用戶發送訊息時調用此函數。

   - It adds the user's message to the `"messages"` state as an "`outgoing`" message.

     它將用戶的訊息添加到 "messages" 狀態作為 "outgoing" 訊息。

   - Sets `isTyping` to `true` , indicating that the AI is processing the response.

     設置 `isTyping` 為 `true`，表示AI正在處理回應。

   - Calls the `processMessageToChatGPT` function to send the conversation history to the AI model and await its response.

     調用 `processMessageToChatGPT` 函數將對話歷史發送到AI模型並等待其回應。

   - Upon receiving the response, it adds the AI's message to the `"messages"` state as an `"incoming"` message and sets `isTyping` to `false` .

     在收到回應後，它將AI的訊息添加到 "messages" 狀態作為 "incoming" 訊息，並將 `isTyping` 設置為 `false`。

4. **processMessageToChatGPT Function:**
   - This function prepares the chat messages to be sent to the ChatGPT API.

     此函數準備要發送到ChatGPT API的聊天訊息。

   - It maps each message in the `chatMessages` array to an object with a "role" (either "assistant" or "user") and "content" (the message text).

     將 `chatMessages` 陣列中的每條訊息映射為一個具有 "role"（"assistant" 或 "user"）和 "content"（訊息文本）的物件。

   - Creates a request body with the model `(gpt-3.5-turbo)` and the messages, including a system message ("I'm a Student using ChatGPT for learning"). This message is used to set context and instruct the model on how to behave during the conversation.

     使用模型 `(gpt-3.5-turbo)` 和訊息創建請求主體，包括一條系統訊息（"我是使用ChatGPT進行學習的學生"）。此訊息用於設置上下文並指導模型在對話期間的行為。

   - Makes a POST request to the ChatGPT API using `fetch` and returns the API response as JSON.

     使用 `fetch` 發送POST請求到ChatGPT API，並將API回應作為JSON回傳。

5. **Rendered Components:**

   渲染的組件：

   - The `App` component returns JSX, representing the chat interface.

     `App` 組件返回JSX，表示聊天界面。

   - The interface consists of a `MainContainer` containing a `ChatContainer` .

     介面包括一個包含 `ChatContainer` 的 `MainContainer`。

   - Inside the `ChatContainer` , there's a `MessageList` , which renders the chat messages.

     在 `ChatContainer` 內部，有一個 `MessageList`，用於渲染聊天訊息。

   - Each message is represented by the `Message` component with its corresponding data.

     每條訊息由 `Message` 組件表示，帶有相應的數據。

   - The `MessageInput` component allows you to send messages and calls the `handleSendRequest` function on send.

     `MessageInput` 組件允許您發送訊息並在發送時調用 `handleSendRequest` 函數。

6. **Styling and Layout:**

- The outer `div` with `className="App"` sets up the app's base styling.

  具有 `className="App"` 的外部 `div` 設置應用程式的基本樣式。

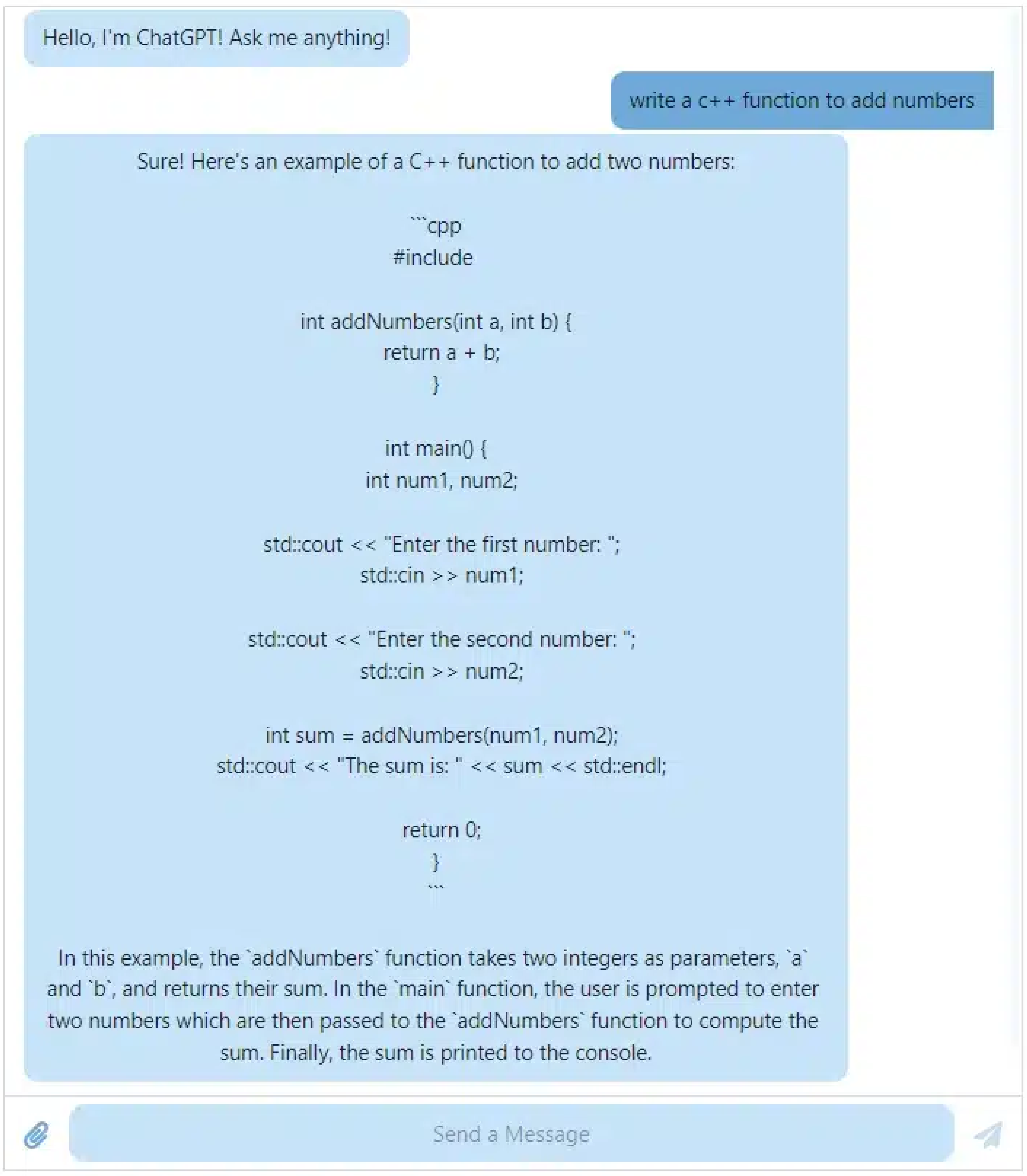
- The chat interface is contained within a `div` with custom inline styling for height and width.

  聊天界面包含在一個具有自定義內聯樣式（用於高度和寬度）的 `div` 內。

Overall, the code set up a React app with a chat interface showing messages and allowing users to interact with the AI.

總的來說，這段代碼設置了一個React應用程式，具有一個顯示訊息並允許用戶與AI交互的聊天界面。

Example: Prompt ChatGPT for C++ code to add two numbers



Hello, I'm ChatGPT! Ask me anything!

write a c++ function to add numbers

Sure! Here's an example of a C++ function to add two numbers:

~~~
```cpp
#include

int addNumbers(int a, int b) {
return a + b;
}

int main() {
int num1, num2;

std::cout << "Enter the first number: ";
std::cin >> num1;

std::cout << "Enter the second number: ";
std::cin >> num2;

int sum = addNumbers(num1, num2);
std::cout << "The sum is: " << sum << std::endl;

return 0;
}
```
~~~

In this example, the `addNumbers` function takes two integers as parameters, `a` and `b`, and returns their sum. In the `main` function, the user is prompted to enter two numbers which are then passed to the `addNumbers` function to compute the sum. Finally, the sum is printed to the console.

ChatGPT prompt for C++ code to add two numbers

Pro tip: the best way to handle errors from the ChatGPT API

專業提示：處理ChatGPT API的錯誤的最佳方式

It's a best practice to monitor exceptions that occur when interacting with any external API. For example, the API might be temporarily unavailable, or the expected parameters or response format may have changed and you might need to update your code, and your code should be the thing to tell you about this. Here's how to do it with error monitoring platform Rollbar:

在與任何外部API交互時，監控發生的異常是一種最佳實踐。例如，API可能暫時不可用，或者預期的參數或響應格式可能已更改，您可能需要更新代碼，而您的代碼應該告訴您這一點。以下是使用錯誤監控平台 Rollbar 的方法：

First install Rollbar using the node package manager, npm:

首先使用node package manager，npm，安裝Rollbar：

```
npm install --save rollbar
```

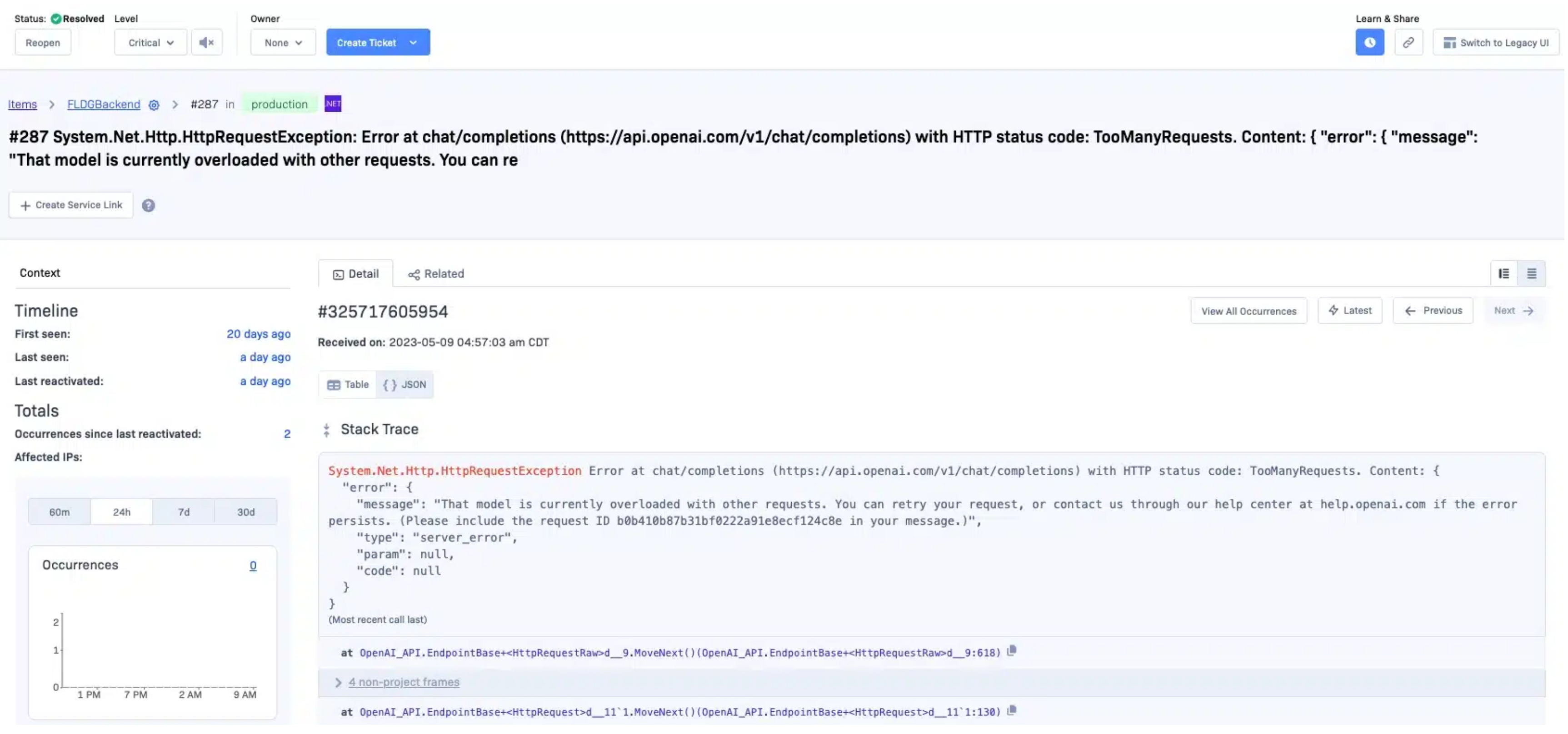
Next, setup the Rollbar configuration (example using Express):

接下來，設置Rollbar配置（以下是使用Express的示例）：

```
var express = require('express');
var Rollbar = require('rollbar');
var rollbar = new Rollbar('POST_SERVER_ITEM_ACCESS_TOKEN');

var app = express();

app.get('/', function(req, res) {
  // ...
});

// Use the rollbar error handler to send exceptions to your rollbar account
app.use(rollbar.errorHandler());

app.listen(6943);
```

Example of an API error:

(Caught by Rollbar)



#287 System.Net.Http.HttpRequestException: Error at chat/completions (https://api.openai.com/v1/chat/completions) with HTTP status code: TooManyRequests. Content: { "error": { "message": "That model is currently overloaded with other requests. You can re

#325717605954

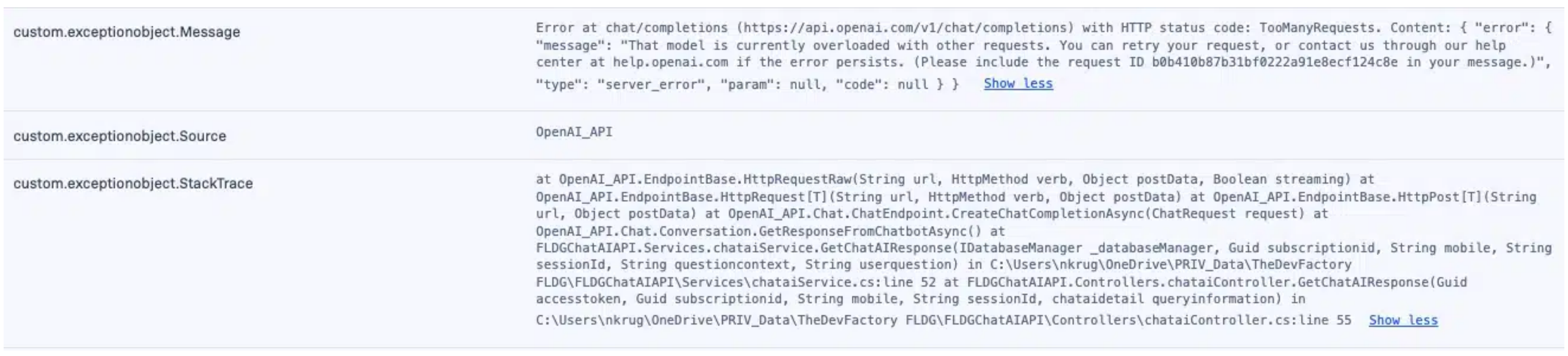
Received on: 2023-05-09 04:57:03 am CDT

Timeline	
First seen:	20 days ago
Last seen:	a day ago
Last reactivated:	a day ago

Totals	
Occurrences since last reactivated:	2
Affected IPs:	

```
System.Net.Http.HttpRequestException Error at chat/completions (https://api.openai.com/v1/chat/completions) with HTTP status code: TooManyRequests. Content: {
  "error": {
    "message": "That model is currently overloaded with other requests. You can retry your request, or contact us through our help center at help.openai.com if the error persists. (Please include the request ID b0b410b87b31bf0222a91e8ecf124c8e in your message.)",
    "type": "server_error",
    "param": null,
    "code": null
  }
}
(Most recent call last)
  at OpenAI_API.EndpointBase+<HttpRequestRaw>d__9.MoveNext()(OpenAI_API.EndpointBase+<HttpRequestRaw>d__9:618)
 > 4 non-project frames
  at OpenAI_API.EndpointBase+<HttpRequest>d__11`1.MoveNext()(OpenAI_API.EndpointBase+<HttpRequest>d__11`1:130)
```

Example of API error caught by Rollbar



custom.exceptionobject.Message	`Error at chat/completions (https://api.openai.com/v1/chat/completions) with HTTP status code: TooManyRequests. Content: { "error": { "message": "That model is currently overloaded with other requests. You can retry your request, or contact us through our help center at help.openai.com if the error persists. (Please include the request ID b0b410b87b31bf0222a91e8ecf124c8e in your message.)", "type": "server_error", "param": null, "code": null } }` Show less
custom.exceptionobject.Source	`OpenAI_API`
custom.exceptionobject.StackTrace	`at OpenAI_API.EndpointBase.HttpRequestRaw(String url, HttpMethod verb, Object postData, Boolean streaming) at OpenAI_API.EndpointBase.HttpRequest[T](String url, HttpMethod verb, Object postData) at OpenAI_API.EndpointBase.HttpPost[T](String url, Object postData) at OpenAI_API.Chat.ChatEndpoint.CreateChatCompletionAsync(ChatRequest request) at OpenAI_API.Chat.Conversation.GetResponseFromChatbotAsync() at FLDGChatAIAPI.Services.chataiService.GetChatAIResponse(IDatabaseManager _databaseManager, Guid subscriptionid, String mobile, String sessionId, String questioncontext, String userquestion) in C:\Users\nkrug\OneDrive\PRIV_Data\TheDevFactory FLDG\FLDGChatAIAPI\Services\chataiService.cs:line 52 at FLDGChatAIAPI.Controllers.chataiController.GetChatAIResponse(Guid accesstoken, Guid subscriptionid, String mobile, String sessionId, chataidetail queryinformation) in C:\Users\nkrug\OneDrive\PRIV_Data\TheDevFactory FLDG\FLDGChatAIAPI\Controllers\chataiController.cs:line 55` Show less

Example error message of API error caught by Rollbar

To get your Rollbar access token, sign up for free and follow the instructions for React.

We can't wait to see what you build with ChatGPT. Happy coding!