Lesson 07 Demo 02

ChatGPT-3 Node.js Interface

Objective: To demonstrate the integration of ChatGPT-3 with Node.js, highlighting its ease and versatility for various web and data applications

Tools required: Node.js

Prerequisites: API key

Steps to be followed:

1. Install the OpenAI Node.js library
2. Set up the API key
3. Send API request

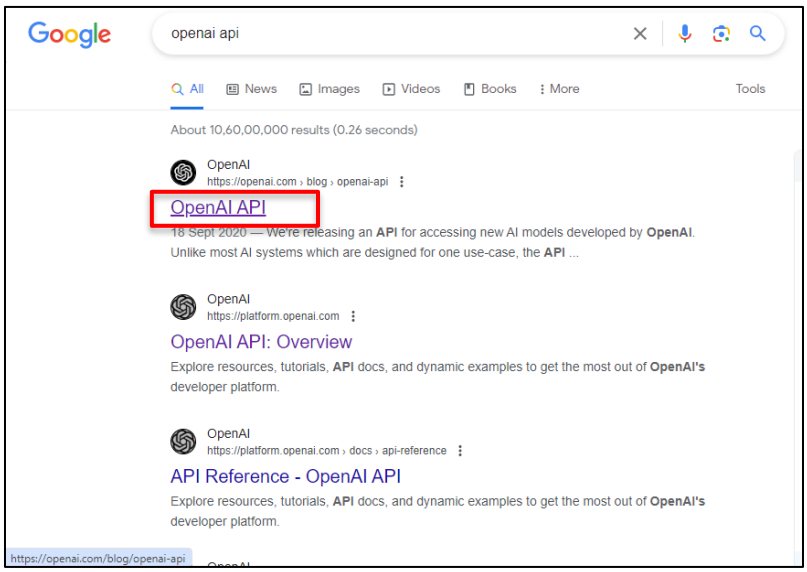
Step 1: Install the OpenAI Node.js library

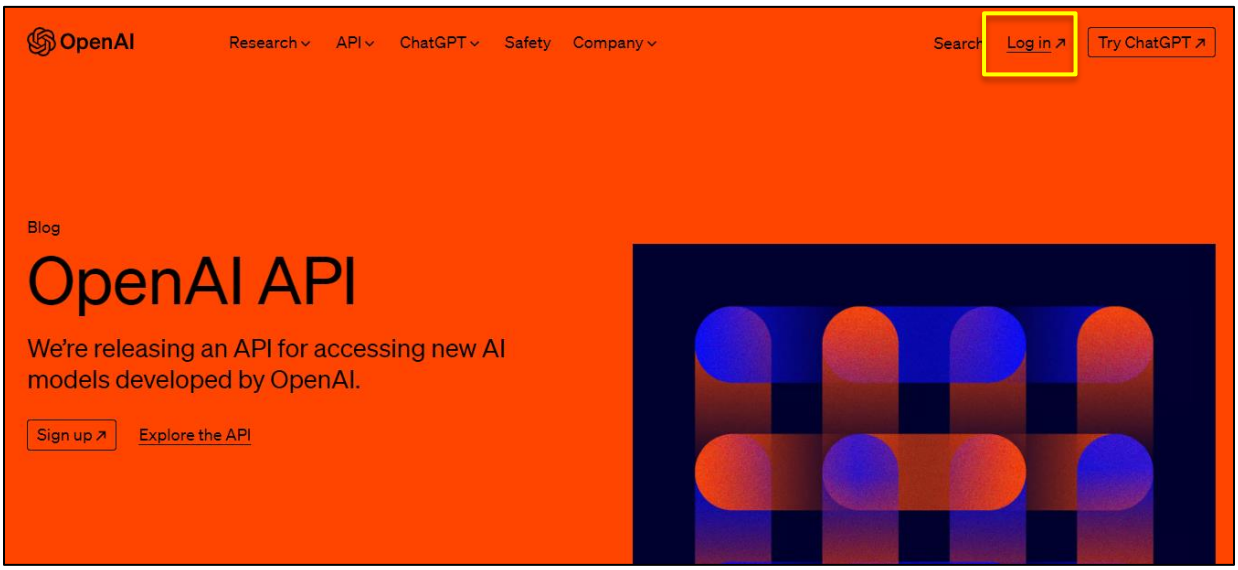
1.1 Launch the lab, then open the terminal and check the node.js version by using the command **node --version**

1.2 Now you can install OpenAI node.js library by using the below command:
npm install --save openai

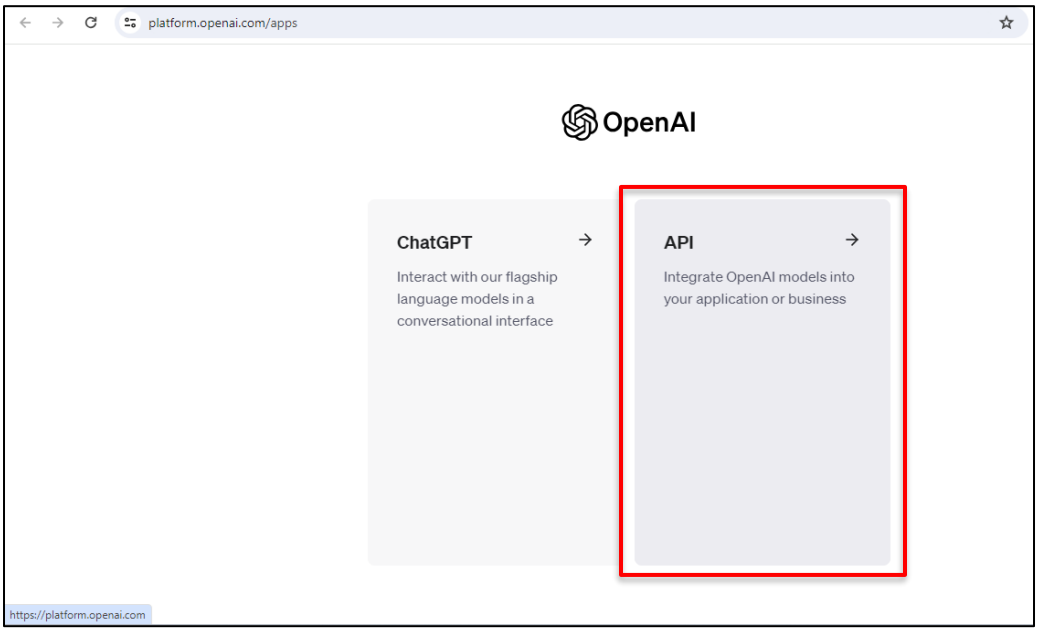
Step 2: Set up API key

2.1 Open your web browser, search for the OpenAI API, and click on the first link. Then, log into your OpenAI account as shown below:

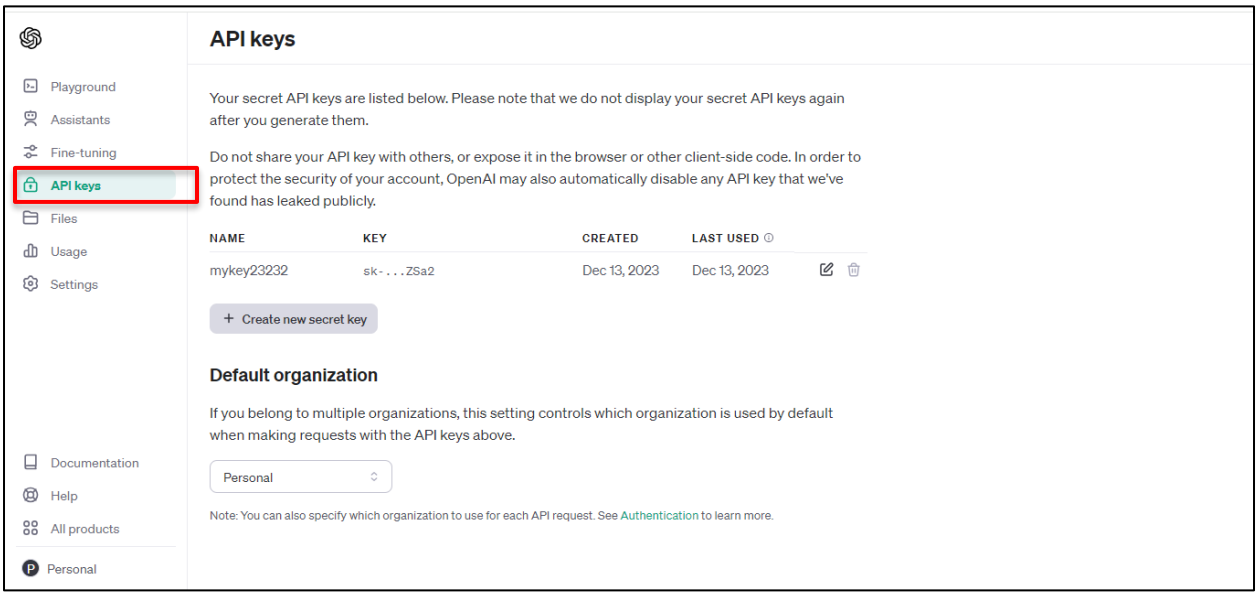




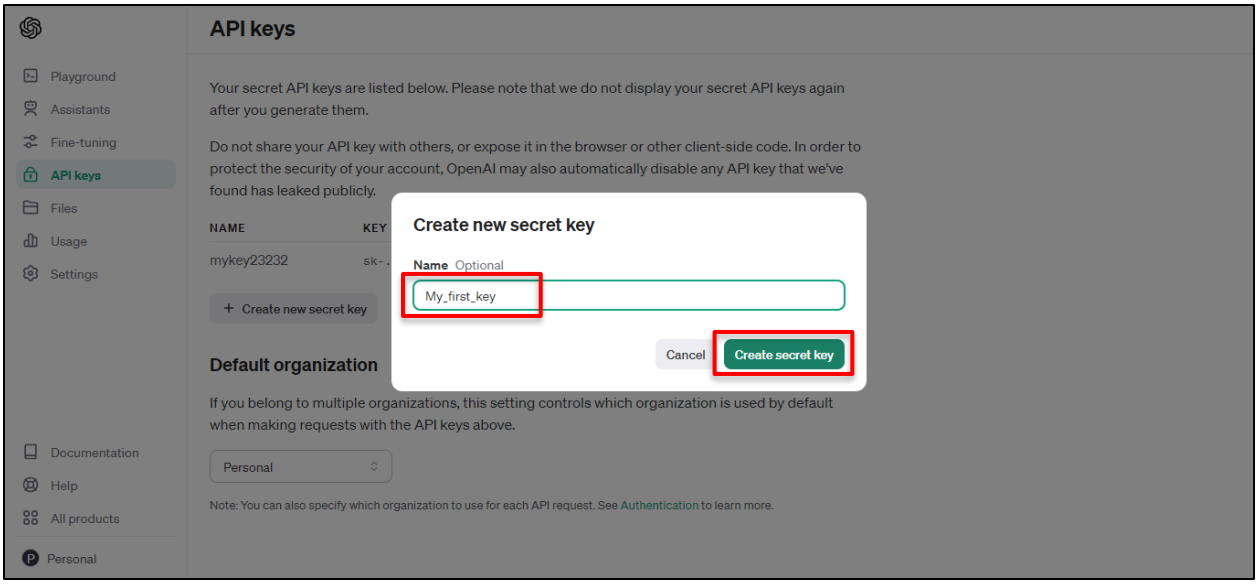
Now, click on **API** option as shown below:



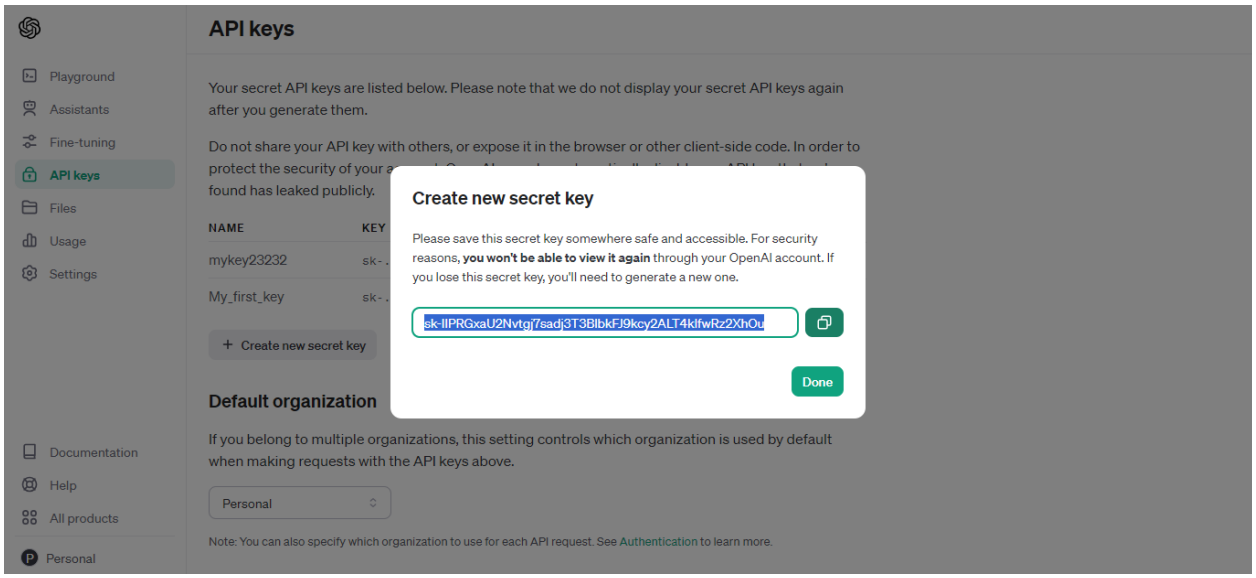
2.2 Now, click on **API Keys** option as shown:



2.3 From the API Keys window, click on **Create new secret key** and give an arbitrary name to the key, then click on **Create secret key**



Copy the secret key generated and paste it in a notepad for future reference



Step 3: Send API request

3.1 Go back to your terminal and create a **.js** file named as openai-test.js by using the command **touch openai-test.js**

```
File  Edit  View  Search  Terminal  Help
prakharguptasim@ip-172-31-42-97:~$ touch openai-test.js
prakharguptasim@ip-172-31-42-97:~$
```

3.2 Open the file by using the command **vi openai-test.js**

```
prakharguptasim@ip-172-31-42-97:~$ touch openai-test.js
prakharguptasim@ip-172-31-42-97:~$ vi openai-test.js
```

3.3 Now, paste the below code inside the file, and then save the file

```
const OpenAI = require("openai");

const openai = new OpenAI({
apiKey: "YOUR API KEY"
});

const openFun=async()=>{
const chatCompletion = await openai.chat.completions.create({
model: "gpt-3.5-turbo",
messages: [{"role": "user", "content": "YOUR PROMPT TEXT",}],
max_tokens:100
});
console.log(chatCompletion.choices[0].message.content);
}
```

Note: In the above code, replace **YOUR API KEY** with the key generated in step 2.3 and **YOUR PROMPT TEXT** with any arbitrary prompt you want. In this demo, we have given **what is MERN Stack.**

```
File  Edit  View  Search  Terminal  Help
const OpenAI = require("openai");


const openai = new OpenAI({
  apiKey: "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"
});


const openFun=async()=>{
const chatCompletion = await openai.chat.completions.create({
    model: "gpt-3.5-turbo",
    messages: [{"role": "user", "content": "what is MERN Stack",}],
    max_tokens:1000
  });
  console.log(chatCompletion.choices[0].message.content);
}
-- INSERT --                                                    19,63          Top
```

3.4 Now, run the **.js** file by using the command **node openai.test.js** to get a response

```
File  Edit  View  Search  Terminal  Help
prakharguptasim@ip-172-31-42-97:~$ node openai-test.js
MERN Stack is a popular web development stack that consists of four technologies: MongoDB, Express.js, React.js, and Node.js.

Here is a breakdown of each component:

1. MongoDB: MongoDB is a NoSQL database that allows for the storage and retrieval of data in a flexible, JSON-like format. It is a popular choice for
 MERN Stack because it provides scalability and flexibility to handle large amounts of data.

2. Express.js: Express.js is a fast and minimalist web application framework for Node.js. It provides a set of robust tools and features to build web
 applications and APIs. Express.js helps to handle HTTP requests, routing, and middleware.

3. React.js: React.js is a JavaScript library for building user interfaces. It allows developers to create reusable UI components and efficiently upd
ate only the necessary parts of the user interface as the data changes. React.js is known for its declarative and component-based approach.

4. Node.js: Node.js is a runtime environment that allows developers to build server-side applications using JavaScript. It provides a non-blocking, e
vent-driven architecture that is ideal for building scalable and high-performance web applications. Node.js enables running JavaScript code on the se
rver-side and allows for seamless integration with other parts of the MERN Stack.

By combining MongoDB, Express.js, React.js, and Node.js, developers can create full-stack web applications using a single language (JavaScript) throu
ghout the entire development process. This eliminates the need for context switching and allows for efficient development and maintenance of the appl
ication.
prakharguptasim@ip-172-31-42-97:~$
```

By following the above steps, you have successfully integrated ChatGPT- 3 with Node.js.