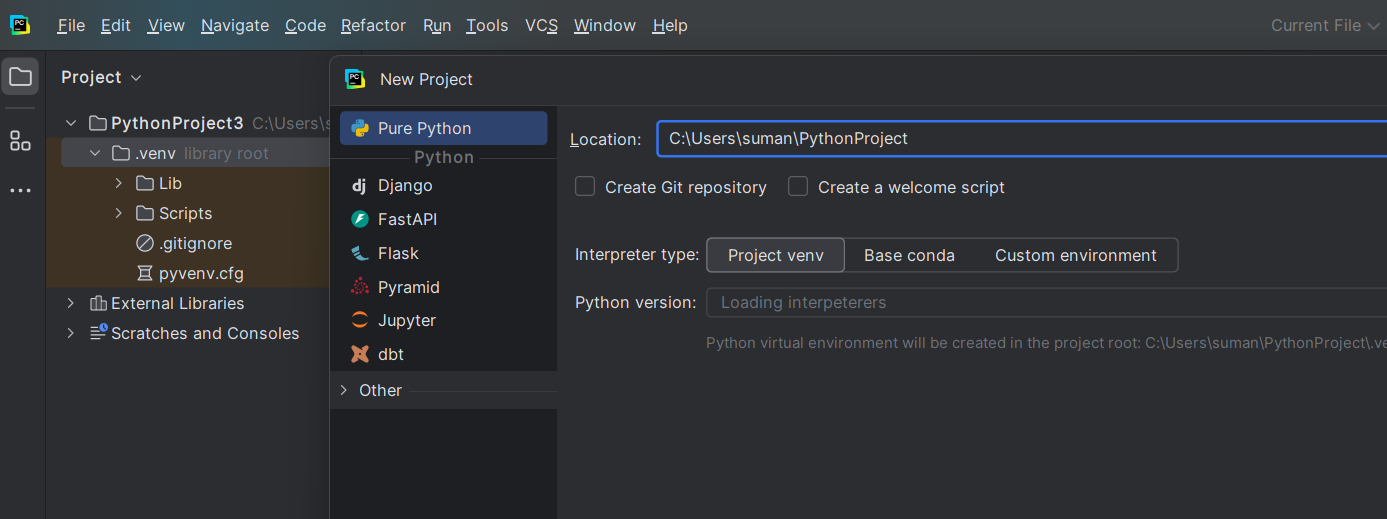
**Integrating LangChain in PyCharm**

1. **Creating a New PyCharm Project with a Virtual Environment**

**a. Create a New Project:**

Go to File → New Project…

Project Location:  
Choose a directory to store your project (C:\Users\<YourName>\PycharmProjects\LangChainProject).



**b. Configure a Virtual Environment:**

There are two methods to create the virtual environment

1. **Open Run Configurations**:

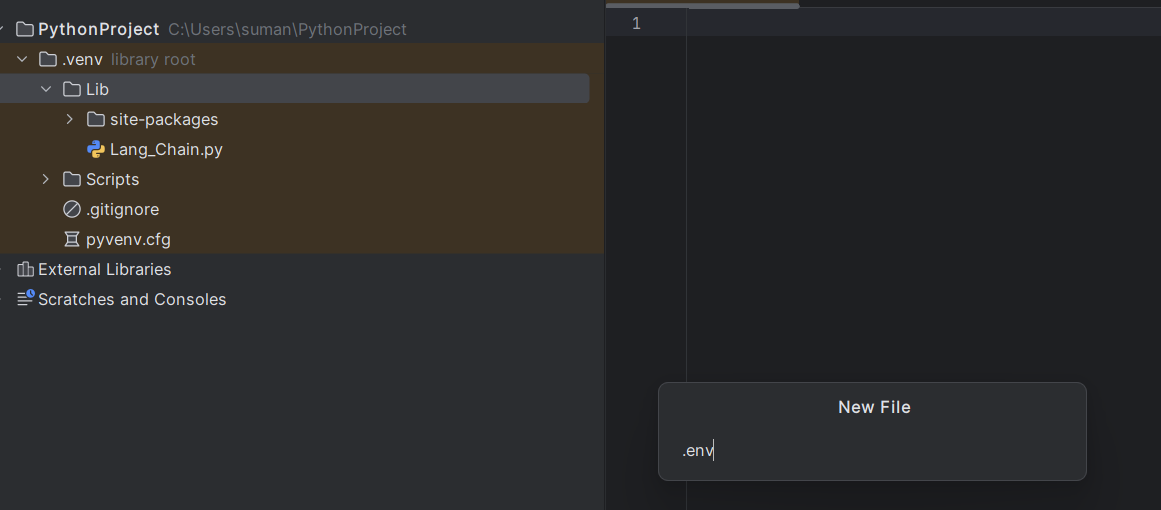
* In PyCharm, go to Run → Edit Configurations….
* Select your run configuration (the one you use to run your script).

Add the Environment Variable:

* In the Environment Variables field, click the “…” button.
* Click the + button to add a new variable.
* Set Name to OPENAI\_API\_KEY and Value to your actual API key (e.g., sk-1234YOURKEY).
* Click OK and then Apply to save your changes.

**2. Use a .env File (Alternative Method)**

Install python-dotenv: Open your terminal in PyCharm and run

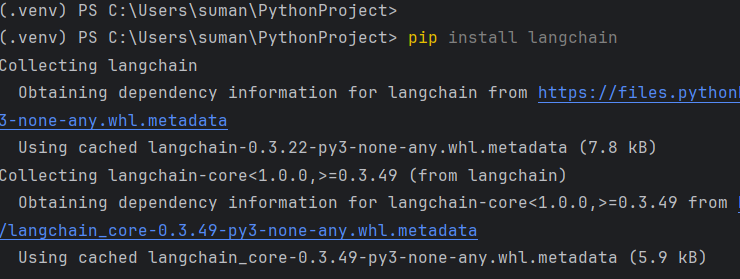
Create a .env File 

Add your API key like this: OPENAI\_API\_KEY=sk-1234YOURKEY

**replace sk-xxxxxxxxxxxxxxxxxxxx with your actual API key.**

**2. Installing Required Packages**

pip install langchain

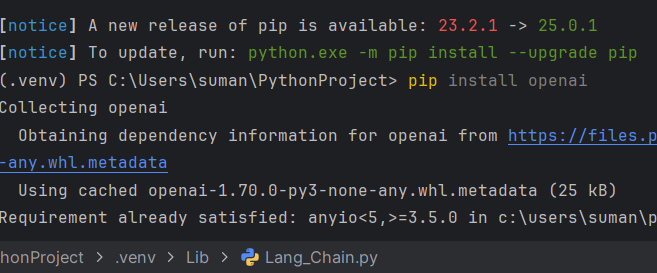


**Install Additional Dependencies:**

For this integration example, we will use:

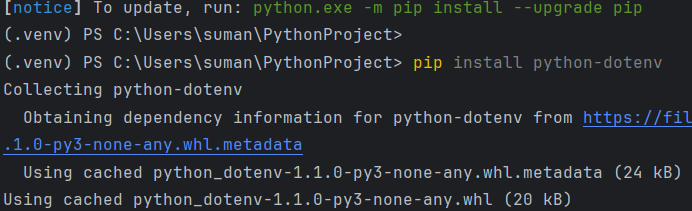
openai – for connecting to OpenAI APIs.

pip install openai



python-dotenv – for managing environment variables.

pip install python-dotenv



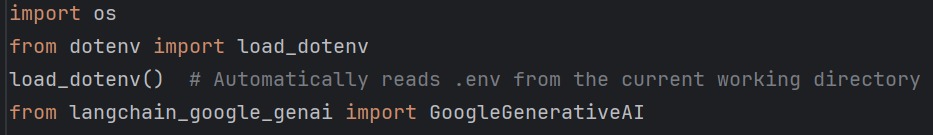
We have used langchain\_google\_genai – for integrating with Google Generative AI (if needed).

pip install openai python-dotenv langchain\_google\_genai

A screen shot of a computer

AI-generated content may be incorrect.

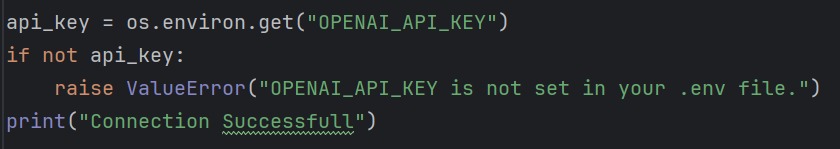
**3. Integrate Langchain in Your Code**



The os module is imported to interact with the operating system (e.g., to access environment variables).

The load\_dotenv function from the dotenv module is used to load environment variables from a .env file. This keeps sensitive information (like API keys) out of your source code.

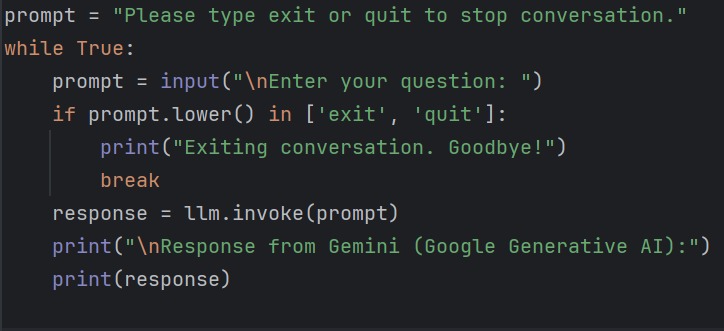
This imports the GoogleGenerativeAI class from the langchain\_google\_genai package, which is used to interact with Google’s Generative AI (in this case, the Gemini model).



The code retrieves the API key stored in an environment variable named OPENAI\_API\_KEY.

If the key isn’t found, the program raises a ValueError to stop execution.

If the key exists, it prints a message confirming a successful connection.



**Initial Message:**

A prompt message informs the user that they can exit the conversation by typing "exit" or "quit".

**Infinite Loop (while True):**

The code enters a continuous loop where:

**User Input:**

It asks the user to input a question.

**Exit Condition:**

If the input (converted to lowercase) is either "exit" or "quit", the loop is terminated with a farewell message.

**AI Model Invocation:**

For any other input, it calls llm.invoke(prompt), sending the user's question to the Gemini Generative AI model.

**Output:**

The response from the AI is printed to the console.

