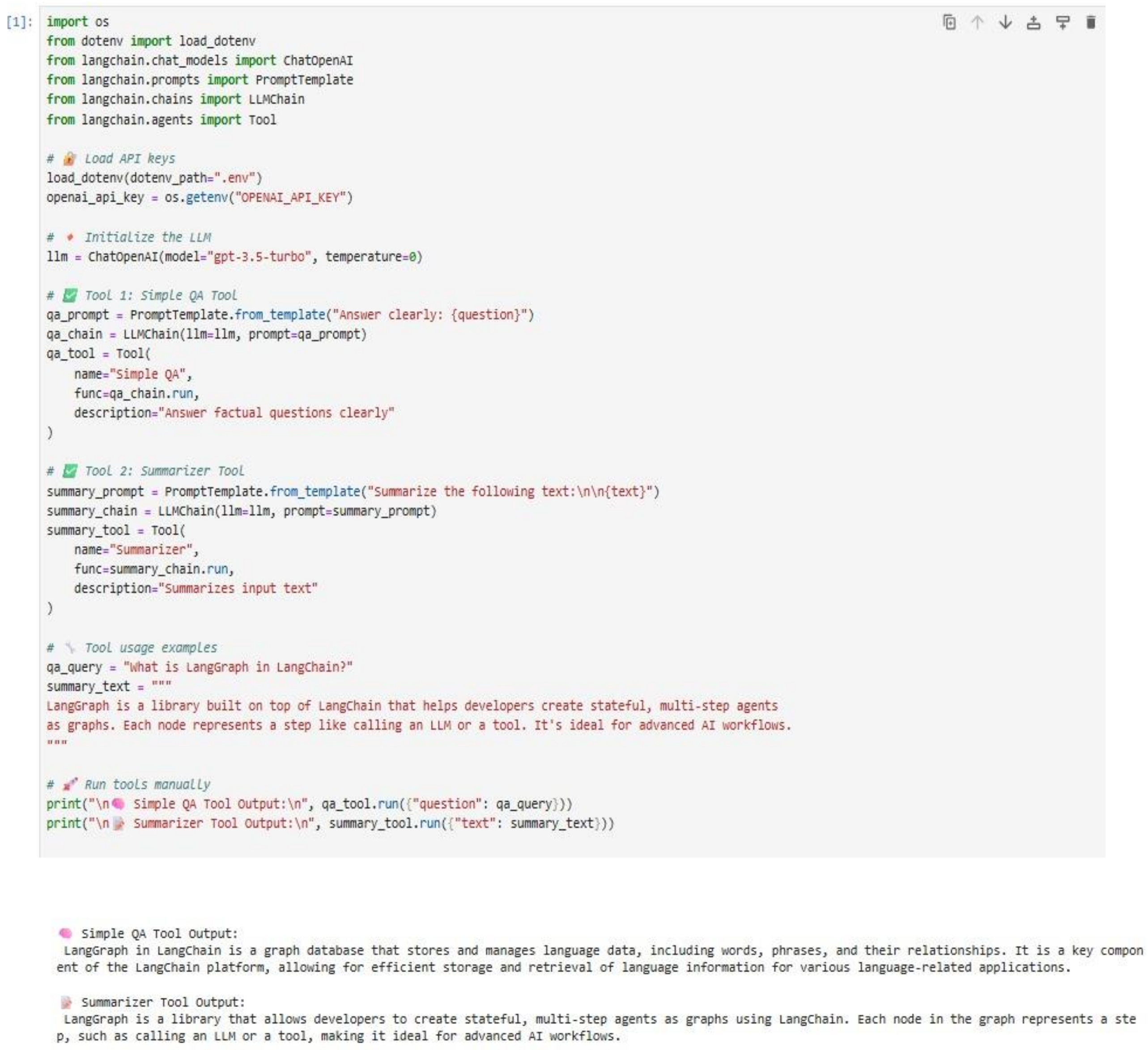

```
[1]: import os
     from dotenv import load_dotenv
     from langchain.chat_models import ChatOpenAI
     from langchain.prompts import PromptTemplate
     from langchain.chains import LLMChain
     from langchain.agents import Tool

     # 🔐 Load API keys
     load_dotenv(dotenv_path=".env")
     openai_api_key = os.getenv("OPENAI_API_KEY")

     # 🔸 Initialize the LLM
     llm = ChatOpenAI(model="gpt-3.5-turbo", temperature=0)

     # ✅ Tool 1: Simple QA Tool
     qa_prompt = PromptTemplate.from_template("Answer clearly: {question}")
     qa_chain = LLMChain(llm=llm, prompt=qa_prompt)
     qa_tool = Tool(
         name="Simple QA",
         func=qa_chain.run,
         description="Answer factual questions clearly"
     )

     # ✅ Tool 2: Summarizer Tool
     summary_prompt = PromptTemplate.from_template("Summarize the following text:\n\n{text}")
     summary_chain = LLMChain(llm=llm, prompt=summary_prompt)
     summary_tool = Tool(
         name="Summarizer",
         func=summary_chain.run,
         description="Summarizes input text"
     )

     # 🔧 Tool usage examples
     qa_query = "What is LangGraph in LangChain?"
     summary_text = """
     LangGraph is a library built on top of LangChain that helps developers create stateful, multi-step agents
     as graphs. Each node represents a step like calling an LLM or a tool. It's ideal for advanced AI workflows.
     """

     # 🚀 Run tools manually
     print("\n🧠 Simple QA Tool Output:\n", qa_tool.run({"question": qa_query}))
     print("\n📝 Summarizer Tool Output:\n", summary_tool.run({"text": summary_text}))
```

```
🧠 Simple QA Tool Output:
 LangGraph in LangChain is a graph database that stores and manages language data, including words, phrases, and their relationships. It is a key compon
ent of the LangChain platform, allowing for efficient storage and retrieval of language information for various language-related applications.

📝 Summarizer Tool Output:
 LangGraph is a library that allows developers to create stateful, multi-step agents as graphs using LangChain. Each node in the graph represents a ste
p, such as calling an LLM or a tool, making it ideal for advanced AI workflows.
```