Conversational RAG with LangChain, Ollama, and Web Search in Gradio

Project Description:

This project implements a **Conversational RAG (Retrieval-Augmented Generation)** chatbot using **LangChain** powered by a **local LLM through Ollama**, capable of answering technical queries using:

- A **vector database of LangChain documentation or local files**,
- **Internet search via SerpAPI**, and
- Optional **Python code execution** for dynamic reasoning.

A **Gradio interface** enables real-time, user-friendly interaction.

Objectives:

- Load a **local LLM using Ollama** (e.g., llama3, mistral, etc.).
- Index a **local set of documents** (e.g., LangChain markdown files or tutorials).
- Integrate **SerpAPI** as a fallback for web queries.
- Provide a **Gradio-based chat interface** with history.
- Use **LangChain Agents (CHAT_CONVERSATIONAL_REACT_DESCRIPTION)** to route queries intelligently between tools (vector store, search, Python).

Components:

1. **Local LLM**:
    - Hosted using Ollama.
    - Accessed via ChatOllama in LangChain.
2. **Document RAG**:
    - Convert LangChain documentation to text/markdown.
    - Split into chunks and store in a vector store (e.g., FAISS).
3. **Search Tool**:
    - Use SerpAPIWrapper from LangChain tools for real-time web search.

4. **Python Tool** *(optional)*:
    - PythonREPLTool from langchain_experimental.
5. **Gradio UI**:
    - Chat box with memory of past messages.
    - Optionally show source documents for RAG results.

💬 Example Prompts:

- “How do I build a custom agent in LangChain?”
- “What are the key features of llama3?”
- “What’s 2 to the power of 20 in Python?”