AI Engineer- Assignment

Objective

Develop a simple AI pipeline using LangChain, LangGraph, and LangSmith to demonstrate an understanding of embeddings, vector databases, RAG, and clean coding practices.

Assignment Requirements

The candidate must:

- Use **LangGraph** to implement an agentic pipeline with **two functionalities**:
  - Fetch **real-time weather data** using the OpenWeatherMap API.
  - Answer questions from a **PDF document** using RAG (Retrieval-Augmented Generation).
- Implement a **LangGraph node** that decides whether to call the weather API or fetch relevant information from the PDF.
- Process the fetched data using a **Large Language Model (LLM)** via LangChain.
- Generate **embeddings** for the processed data and store them in a **vector database** (Qdrant).
- Implement a **RAG-based query mechanism** to retrieve and summarize stored information.
- Evaluate the LLM's response using **LangSmith**.
- Write **test cases** for API handling, LLM processing, and retrieval logic while maintaining clean, modular code.
- Create a **UI interface using Streamlit** to demonstrate the working of the application using a simple chat interface.

Deliverables

- Python code in a **GitHub repository**.
- **README.md** with setup instructions and implementation details.
- **LangSmith logs/screenshots** showcasing LLM response evaluation.
- **Test results** from unit tests.
- **Streamlit UI demo**.
- A **Loom video** explaining your LangSmith results and your code.

Evaluation Criteria

- Correct integration of LangGraph and LangChain.
- Proper decision-making within LangGraph nodes.
- Working vector database storage and retrieval.
- Effective LangSmith evaluation.

- Clean, well-structured, and well-tested code.
- Functional and user-friendly Streamlit UI to demonstrate the application.