Track 2: GenAI Data Scientist - Final Assessment

Scenario:

You have been engaged by a midsized healthcare provider to build an intelligent assistant for clinicians. The assistant should answer complex, patient-specific questions by leveraging both generative AI and secure data retrieval from an internal repository, supporting advanced clinical decision-making.

Objectives:

- Demonstrate mastery of prompt engineering, RAG, fine-tuning.
- Integrate LangChain and FAISS for robust retrieval and orchestration.
- Build, test, and reflect on practical system strengths and limitations.

Assessment Tasks and Deliverables:

1. **Advanced Prompt Engineering**

   Task:

- Design dynamic prompts that tailor the assistant's responses according to medical specialty, patient history, and requested evidence type (summaries, citations, guidelines).

Expected Outcome: Submission of at least three well-documented prompt templates, with reasoning for token usage, context packing, and response reliability.

2. Retrieval-Augmented Generation (RAG) with FAISS

Task:

- Set up a FAISS vector database with sample medical literature and clinical guidelines. Implement a LangChain pipeline that: using clinical_docs: Your sample clinical knowledge snippets.
- OpenAI Embeddings: Generates vector representations. Substitute with other embedding models if needed.
- FAISS.from_texts: Embeds and stores your docs, ready for similarity search.
- Query/search: Try out a search to see what info gets retrieved.

Expected Outcome: A python script following above tasks in steps with response.

3. Fine-Tuning an LLM for Domain Adaptation

Task:

- Using a small, synthetic set of clinical Q&A (provided or created), fine-tune a pre-trained LLM (e.g., using HuggingFace Trainer). Focus on clinical terminology, style, and answer granularity.
- **Expected Outcome:** Report describing fine-tuning approach, Explain steps to finetune a clinical Q&A. No working code required – explain in example scripts.

Recommended Tools and Environment:

- Use GitHub Copilot, Azure OpenAI API, LangChain, and a Vector database (FAISS).
- Ensure the solution runs inside GitHub Codespaces within the DEP-Training GitHub organization.

Instructions for Submitting README.md File and GitHub Repo Link:

- Name your folder as **"YourName_GenAITrack2_FinalAssessment"**.
- Create a README.md file in the**"YourName_GenAITrack2_FinalAssessment" folder.**
- Include comprehensive project details to the README.md file.
- Provide clear instructions and a script to run the code.
- Submit your GitHub repository link in this survey.