AI Medical Chatbot using RAG

(MediRAG)

PROJECT LAYOUT

Phase 1 –Setup Memory for LLM (Vector Database)

- Load raw PDF(s)
- Create Chunks
- Create Vector Embeddings
- Store embeddings in ChromaDB

Phase 2–Connect Memory with LLM

- Setup LLM (Mistral with HuggingFace)
- Connect LLM with ChromaDB
- Create chain

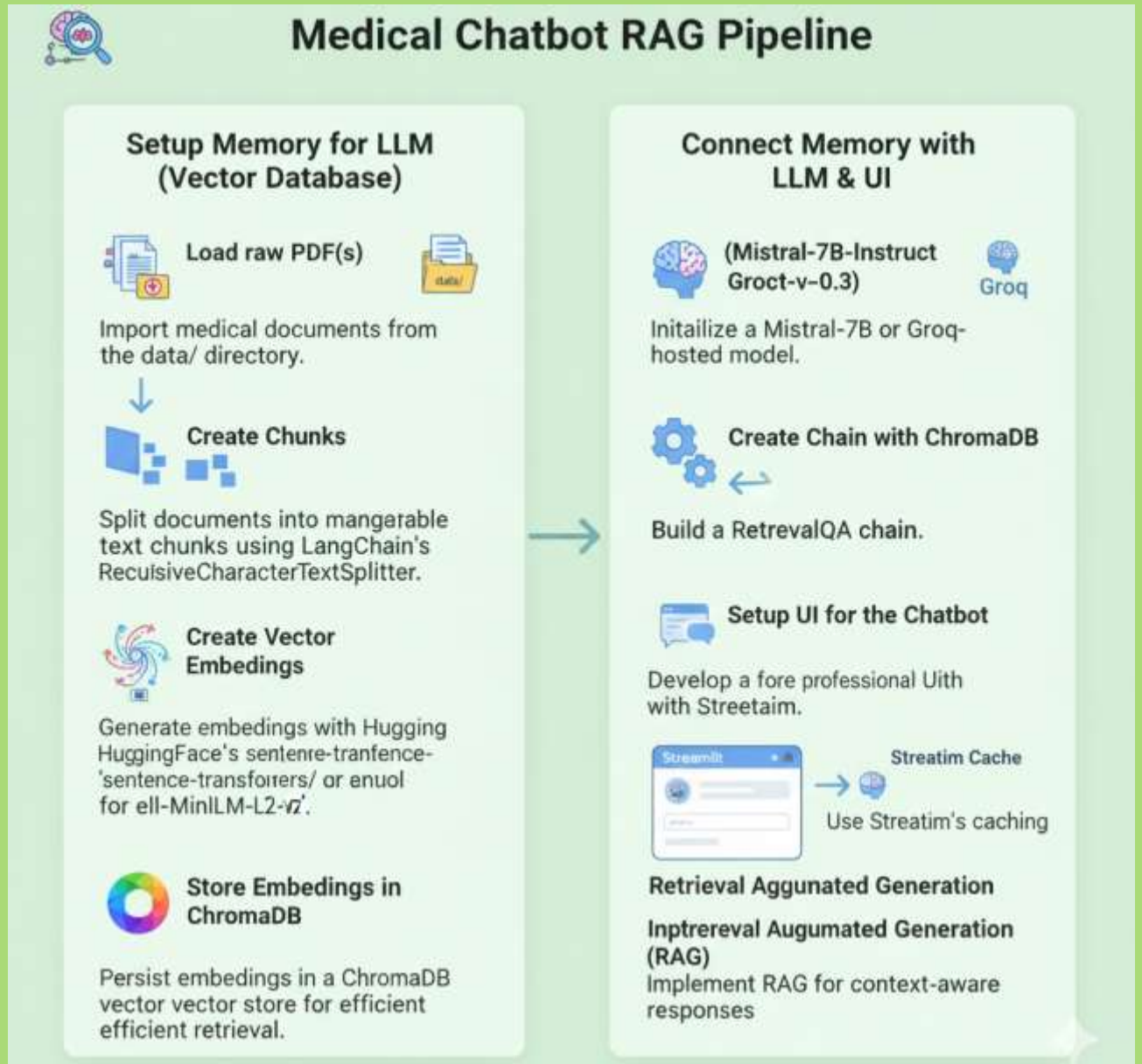
Phase 3–Setup UI for the Chatbot

- Chatbot with Streamlit
- Load Vector store (ChromaDB) in cache
- Retrieval Augmented Generation–RAG

Tools and Technologies

- Langchain (AI Framework for LLM applications)
- HuggingFace (ML/AI Hub)
- Mistral (LLM Model)
- ChromaDB (Vector Database)
- Steamlit (For Chatbot UI)
- Python (Programming Language)
- VS Code (IDE)

Project layout



Improvement Potential/Next Steps

- Add authentication in the UI
- Make use of self-upload document functionality
- Add multiple documents and embed them together
- Add Unit testing of RAG applications

Summary

- **Modern AI Chatbot for Medical Knowledge** MediBot leverages Retrieval-Augmented Generation (RAG) to deliver accurate, context-aware responses from medical PDFs.
- **Modular 3-Phased Development** Structured into memory setup (ChromaDB), LLM integration (LangChain & Groq), and a professional Streamlit UI.
- **Key Technologies Highlighted**

1. Streamlit for interactive UI
2. LangChain & HuggingFace for RAG pipeline
3. ChromaDB for vector storage
4. End-to-End RAG workflow