Internship Task

Build a Local AI Q&A Bot using Ollama + LangChain

Objective

Create a **local Retrieval-Augmented Generation (RAG) chatbot** that answers questions based on uploaded documents (PDFs, TXT, MD, etc.). The bot should run entirely offline using Ollama and a local vector store.

Step-by-Step Breakdown

1. **Model Setup**
    - Install **Ollama** and download a local model (e.g., mistral, llama3, gemma).
    - Verify that the model runs locally and can generate text.
2. **Document Ingestion**
    - Load documents in formats like PDF, TXT, or Markdown.
    - Split documents into manageable chunks (e.g., recursive splitting).
    - Generate embeddings using:
        - ollama/embedding-mistral, or
        - SentenceTransformers (all-MiniLM).
    - Store embeddings in a **vector DB** (ChromaDB, FAISS, etc.).
3. **RAG Setup**
    - On user query:
        - Retrieve top relevant chunks (cosine similarity).
        - Send the chunks + query to Ollama for generation.
    - Return clean, cited answers.
4. **Chat Interface**

- Build a simple UI with **Gradio** or **Streamlit**.
- Must allow:
  - Asking questions
  - Viewing cited context
  - Uploading new files
- (CLI version acceptable if UI is not possible).

5. **Bonus Features (Optional)**
   - Add chat history/memory (LangChain ConversationBufferMemory).
   - Support switching between local models for comparison.
   - Enable **dynamic ingestion** of newly uploaded files.
6. **Testing**
   - Test with factual, follow-up, and edge-case queries.
   - Measure latency and check for hallucinations.
   - Document limitations and improvement ideas.
7. **Deliverables**
   - Github Repo of Codebase with requirements.txt or environment.yml
   - Gradio/Streamlit app OR CLI demo video (Use Loom or similar)
   - Sample documents + 5–10 test queries.
   - README with setup + usage instructions.

💡 In the Loom Screen Recording try to include:

- Live demo: upload 1 doc → ask 2 queries → show cited chunks on each answer.
- Show **ingestion**: chunking params, embeddings call, vector DB size.
- Show **latency** (on-screen timer) for each step.
- Show **failure case** and how they mitigated it (prompt, k, chunk size, re-rank, etc.).
- Architecture slide (1 minute): data flow diagram.
- Brief code tour (key files only, <1 min).

Evaluation Criteria

Criteria	Weight
Understanding of pipeline & architecture	25%
Data processing & embeddings quality	20%
Integration with Ollama	20%
Testing & performance (accuracy, latency)	15%
Interface, UX, & documentation	10%
Bonus features (multi-model, memory, tools)	10

For clarifications email : **niyas@bookcygnus.com**