Take-Home Assignment

In this take-home assignment, you will design and implement a **toy Retrieval-Augmented Generation (RAG)** chatbot that can serve internal company documents (policies, meeting summaries, etc.). You will **curate or synthesize** a small corpus of “internal” documents, then **architect** a RAG pipeline, considering vectorization methods, retriever–reader design, prompt engineering, and enterprise-grade security/privacy. Finally, you will outline deployment options (e.g., on-prem vs. cloud, access controls) and justify technology choices (e.g., LangChain, vector DB, model selection). The goal is to assess both hands-on implementation skills and the candidate’s ability to reason about data governance, scalability, and maintainability in a corporate setting.

1. Context & Objectives

- **Business need**: Employees need a conversational assistant that answers questions about internal documents—policies, procedures, meeting notes—using only authorized content.
- **Core objective**: Build a minimal RAG-based prototype demonstrating retrieval + generation, and produce a design document explaining architecture, data-sensitivity safeguards, and deployment strategy.

2. Assignment Tasks

2.1. Document Curation (Toy Corpus)

1. **Source selection**: Curate 5–10 open-source or synthetic “internal” docs: e.g., company handbook pages, mock meeting minutes, policy briefs.
2. **Preprocessing**: Split into passages/chunks, apply basic cleaning (e.g., remove boilerplate).

2.2. RAG Pipeline Implementation

1. **Embedding & Retrieval**
    - **RAG Methodology**: Compare various RAG methodologies and justify your choice.
    - Choose an embedding model (e.g., Sentence-Transformers) and vector database (e.g., FAISS, Pinecone).
    - Index passages and implement a similarity search retriever.
2. **Generation**
    - Integrate a language model (e.g., OpenAI GPT-3.5/Turbo or an open LLM) to condition on retrieved contexts.
    - Design prompts to yield concise, source-grounded answers.

3. **Evaluation**
    - Demonstrate with 3–5 sample queries (e.g., “What’s the vacation policy?” “Summarize yesterday’s engineering sync”).

3. Deliverables

1. **Code repository** containing:
    - Data preprocessing scripts
    - RAG implementation (retriever + generation)
    - Example queries and outputs
2. **Design document** (PDF/Markdown) covering architecture, and deployment.
3. **README** with setup instructions and explanatory notes.