AI Engineer Technical Assignment Report: RAG Chatbot

Candidate: Divyansh Chourasia
Role: AI Engineer (LLMs & Generative AI)
Organization: Amlgo Labs
Submission Date: 8 August 2025

---

1. System Architecture Overview

This project implements a **Retrieval-Augmented Generation (RAG)** chatbot, designed to answer user queries based on a given domain-specific document (AI Training Policy). The architecture uses a local embedding model and a local LLM for fully offline inference. The pipeline has the following key components:

- **Chunking module**: Parses and segments the `.docx` document
- **Embedding + Vector Store**: Embeds chunks and stores them in ChromaDB
- **Retriever**: Finds relevant chunks based on semantic similarity
- **LLM Response Generator**: Uses `flan-t5-base` to generate answers based on retrieved context

---

2. Document Preprocessing & Chunking Logic

- The uploaded `.docx` file is parsed using the `python-docx` library.
- Sentences are extracted using `nltk.sent_tokenize`.
- Chunks of ~200 characters are created with overlap to preserve context boundaries.
- Each chunk is saved with a unique ID into `chunks/chunks.json`.

This approach ensures that the chunk size fits both token and semantic boundaries, reducing loss of information.

---

3. Embedding Model and Vector Database

- **Embedding Model**: `all-MiniLM-L6-v2` from `sentence-transformers` for fast, lightweight sentence embeddings
- **Vector DB**: `ChromaDB` (using DuckDB+Parquet backend)

Each chunk is encoded into a 384-dimension vector and stored in a persistent Chroma collection named `rag_chunks`. During query time, the top 3 most similar chunks are retrieved using cosine similarity.

---

4. Prompt Design & Generation Logic

The following prompt template is used to guide the LLM:

```
Answer the question based on the context.
Context: <top-k chunks>
Question: <user query>
```

This format ensures the LLM is instructed to rely only on the retrieved document segments, avoiding hallucination and grounding responses in actual content.

---

5. Example Queries and Observations

(A) Success Case

- **Q:** What is the purpose of the AI training program?
- **A:** The purpose is to educate employees on AI compliance, best practices, and policy understanding.

(B) Short Response

- **Q:** Who will take the final call on policy breaches?
- **A:** The management.

(C) Off-topic / Failure

- **Q:** What is the capital of India?
- **A:** [No context retrieved] or hallucinated answer

---

6. Limitations & Future Improvements

- **Short Answers**: Caused by limited context length and base model capacity. Can be improved by using a more expressive model like `mistral-7b`.
- **Hallucination Risk**: Still exists if retrieved chunks are weak or partially relevant.
- **Latency**: Acceptable for small models, but would benefit from GPU acceleration.
- **Streaming Support**: Disabled in final version for stability; can be added for better UX.

---

7. Local Hosting Instructions

```
python chunk_docx.py
python src/embed_and_store.py
streamlit run app.py
```

All models and embeddings are local. No API calls or external services are used, making it suitable for secure enterprise deployment.

End of Report