Chatbot Project Report

1. Project Overview

This project implements a Retrieval-Augmented Generation (RAG) chatbot system that enables users to ask natural language questions about the book on classification on Mental and Behavioral Disorder(AI Training Document) which is provided. The system uses chunked document embedding retrieval and a local large language model (Mistral-7B via Ollama) to generate accurate, grounded responses, with streaming output in a user-friendly web interface.

2. Document Structure and Chunking Logic

Document Structure

- Source: book on classification on Mental and Behavioral Disorder

Chunking Logic

Preprocessing:

- Remove extra whitespace, numbers, line breaks, page breaks, and any markup/HTML.

Sentence-aware chunking:

- Split text into sentences using NLTK for sentence boundaries.
- Group sentences into overlapping "chunks" (~250 words per chunk, 50-word overlap).
- Overlapping windows help preserve context at chunk boundaries.

Rationale:

- Sentence-aware chunking avoids splitting inside sentences, maintaining meaning and coherence.
- Overlap reduces answer failures for queries near the boundaries.

3. Embedding Model and Vector Database

Embedding Model

Model: all-MiniLM-L6-v2 (from Sentence Transformers)

- Fast, lightweight, ~384-dimensional dense vectors.
- Chosen for high performance in retrieval tasks on English legal/contract text.

Process:

- Each chunk encoded as a vector.
- Vectors normalized for cosine similarity.

Vector Database

- **Library:** FAISS (faiss-cpu, Facebook Research)

Why:

- Efficient, in-memory nearest neighbor search.
- Extreme speed for semantic search on small-medium datasets.

Index:

- All chunk vectors stored (IndexFlatIP).
- Allows fast top-k similarity search at query time.

4. Prompt Format and Generation Logic

Prompt Format

- Top-4 most relevant chunks (from FAISS semantic search) are injected in the context window.

Generation Logic

- **Retriever:**
  User query is embedded, top chunks fetched from FAISS.
- **Prompt Construction:**
  Chunks & user query combined as above.
- **LLM Generation:**
  Prompt sent to Mistral-7B-Instruct (run locally via Ollama).
  Model streams its answer, shown in real time in the Streamlit UI.
- **Transparency:**
  The source chunks used for each answer are always displayed to users.

6. Notes on Hallucinations, Model Limitations, and Slow Responses

Hallucinations

- The LLM is instructed to only answer based on provided context.
- If relevant information is not in the top chunks, the model will reply “Sorry, info not found in the document.”
- Occasional hallucination risk if a query is too vague for retrieval or if the LLM over-generalizes—this is reduced by clear prompt instructions.

Model/Deployment Limitations

- The answer quality depends on the chunking; poor chunking can lose context.
- FAISS is in-memory and not suited for very large (multi-million document) deployments, but perfect for contained assignments.
- Ollama models are fast for the hardware, but very large models (beyond 7B) may still lag.

7. Conclusion

- The RAG chatbot enables fact-grounded, conversational Q&A over complex documents.
- Combining chunk-based retrieval (MiniLM + FAISS) with a local streaming LLM (Mistral/Ollama) yields accurate and transparent answers, suitable for both technical and legal users.
- The entire pipeline is modular, reproducible, and accompanied by step-by-step notebook documentation for transparency.