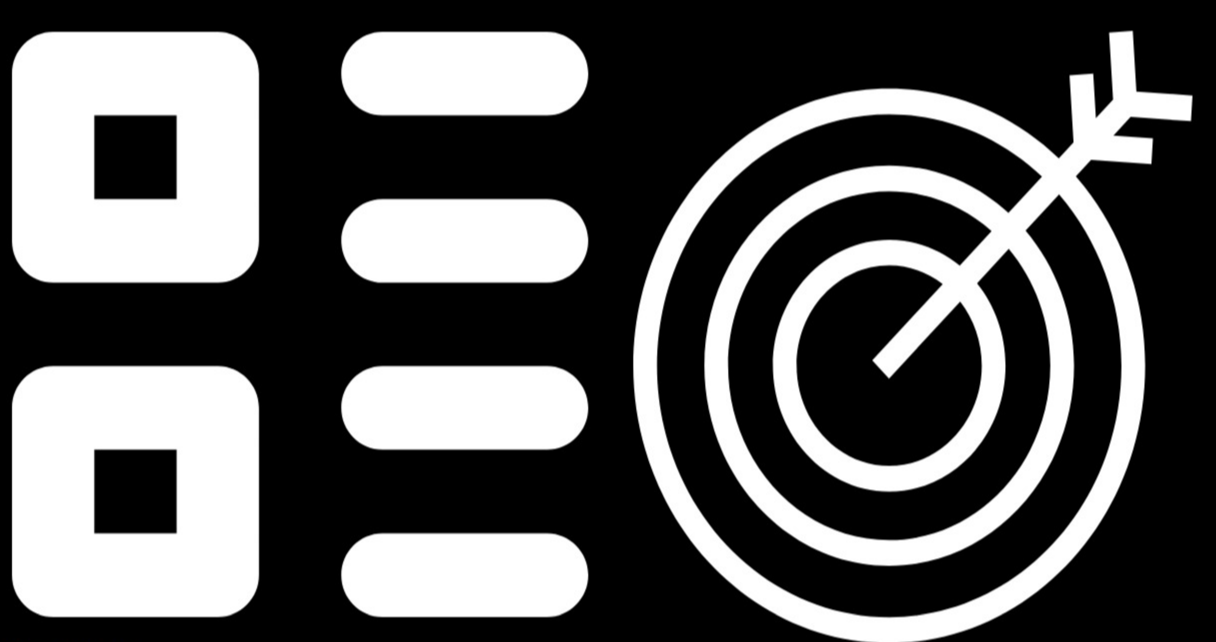
PDF Chatbot with LLAMA2 & RAG

AI-powered assistant for querying PDF content using semantic search and language understanding.



Overview.

This project implements an intelligent chatbot capable of answering user queries based on the content of uploaded PDF documents. It combines semantic retrieval with generative language modeling by using MiniLM2 for dense embeddings and LLAMA2 for natural language responses. The chatbot uses a RAG (Retrieval-Augmented Generation) pipeline, enabling accurate, context-aware answers grounded in the original PDF content.

Objective.

Build a domain-agnostic chatbot for querying PDFs in natural language.
Use lightweight models to ensure fast embedding and inference.
Provide relevant context to the LLM from retrieved document chunks.
Maintain accuracy by grounding responses directly in source content.

Tools.

- **MiniLM2 (all-MiniLM-L6-v2)** – for generating dense vector embeddings of PDF chunks.
- **LLAMA2 (7B)** – as the core language model for answer generation.
- **LangChain / Haystack / Custom RAG** – to connect retriever and generator in a modular pipeline.
- **Chroma** – vector store for fast semantic search.
- **PyMuPDF / pdfminer / pdfplumber** – for extracting clean text from PDFs.
- **Streamlit / Flask / FastAPI** – optional interfaces for interacting with the bot.

Process.

- Extracted raw text from PDF documents and chunked it using overlap strategies.
- Converted each chunk into an embedding using MiniLM2 via SentenceTransformers.
- Stored embeddings in a vector index (e.g., FAISS or Chroma).
- On user query, embedded the question and retrieved top-k relevant chunks.
- Constructed a prompt by combining retrieved context with the question.
- Passed the prompt to LLAMA2 for a grounded, coherent response.

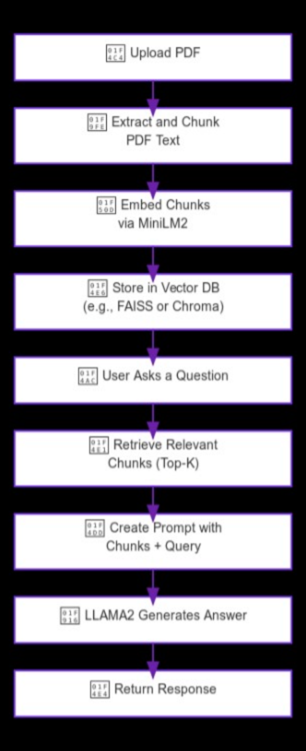
Methodologies.

- **Retrieval-Augmented Generation (RAG):** Uses semantic similarity to fetch relevant context chunks from the document, which are then used to formulate a more accurate and source-grounded response.
- **Embedding + Vector Search:** Uses MiniLM2 for creating low-dimensional vector representations of text, allowing fast and memory-efficient retrieval with FAISS or Chroma.
- **Prompt Engineering:** Combined user query with top-k context snippets in a predefined prompt template for optimal generation quality.
- **Lightweight Deployment:** Leveraged quantized or low-memory variants of LLAMA2 and used efficient libraries for PDF parsing and search.

Challenges

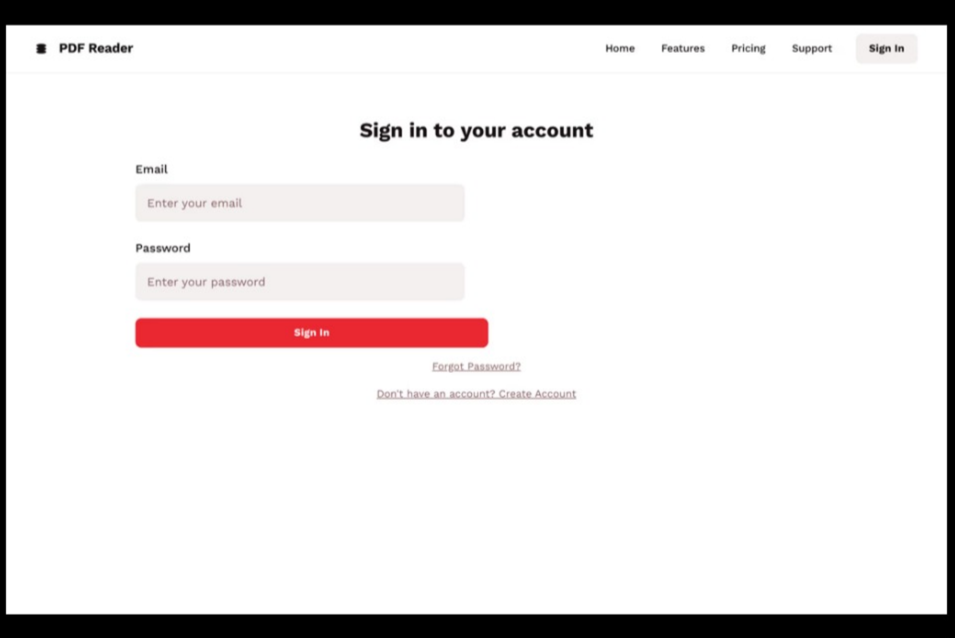
- **Chunk Granularity Trade-off**: Smaller chunks led to incomplete context retrieval, while larger ones diluted relevance or exceeded token limits.
- **PDF Parsing Inconsistencies**: Poorly formatted or scanned PDFs often caused line breaks, headers, and footnotes to interfere with clean chunking.
- **Embedding Speed vs. Accuracy:** Using MiniLM2 ensured speed but sometimes sacrificed nuanced semantic matching, especially in technical domains.
- **Context Injection Limitations**: LLAMA2 occasionally hallucinated when the retrieved context lacked sufficient detail or clarity.
- **Model Loading & Memory Constraints**: Running LLAMA2 locally required careful resource management or quantized versions to prevent OOM errors.

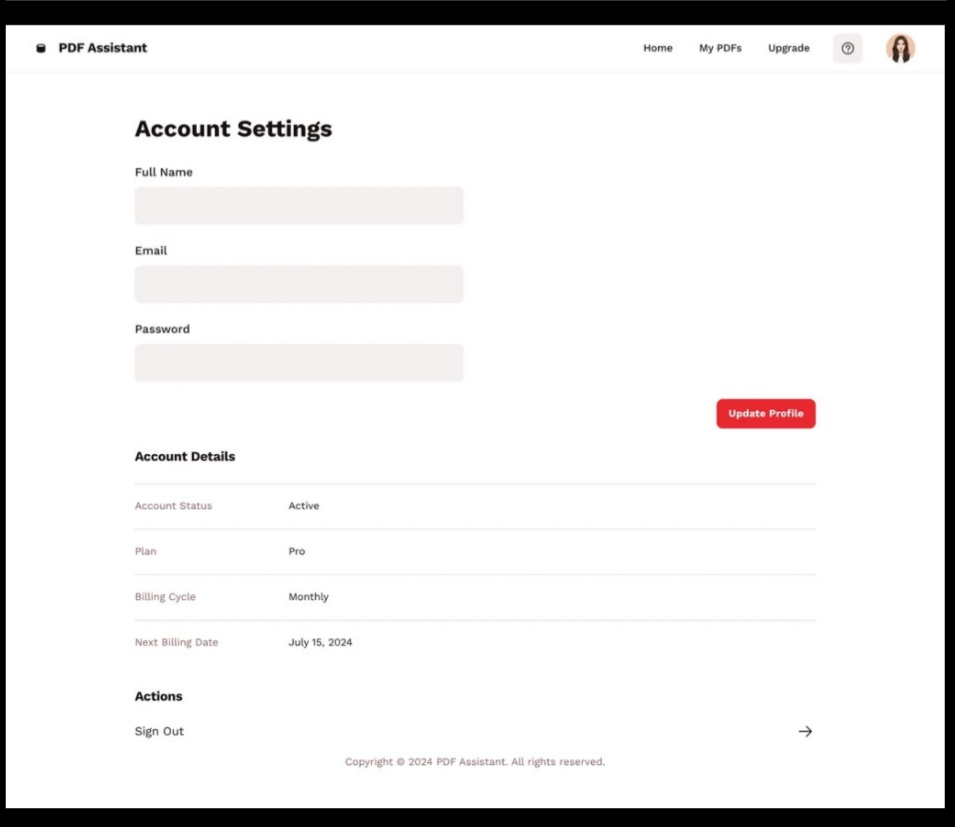
Flowchart

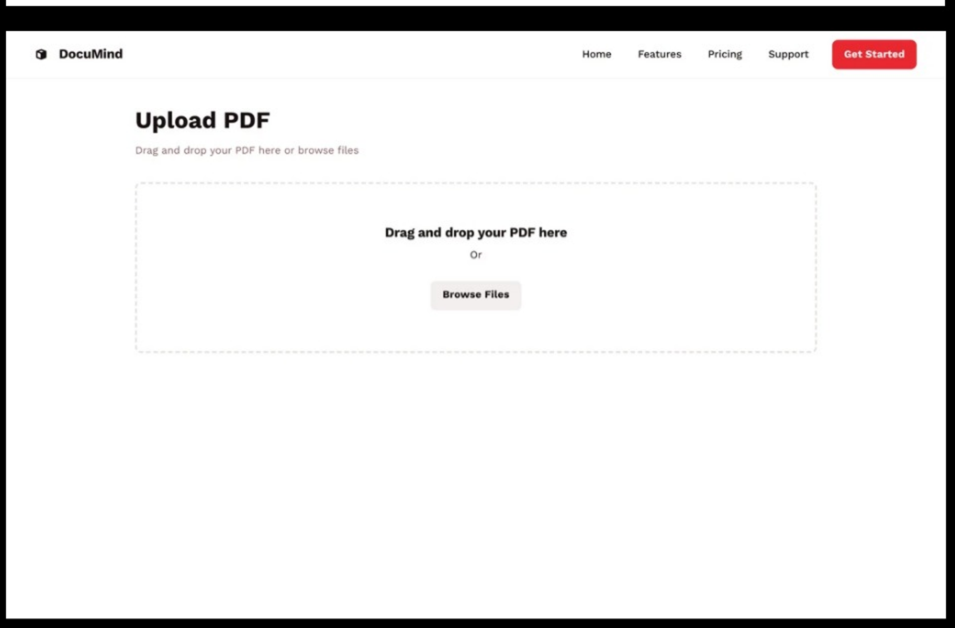


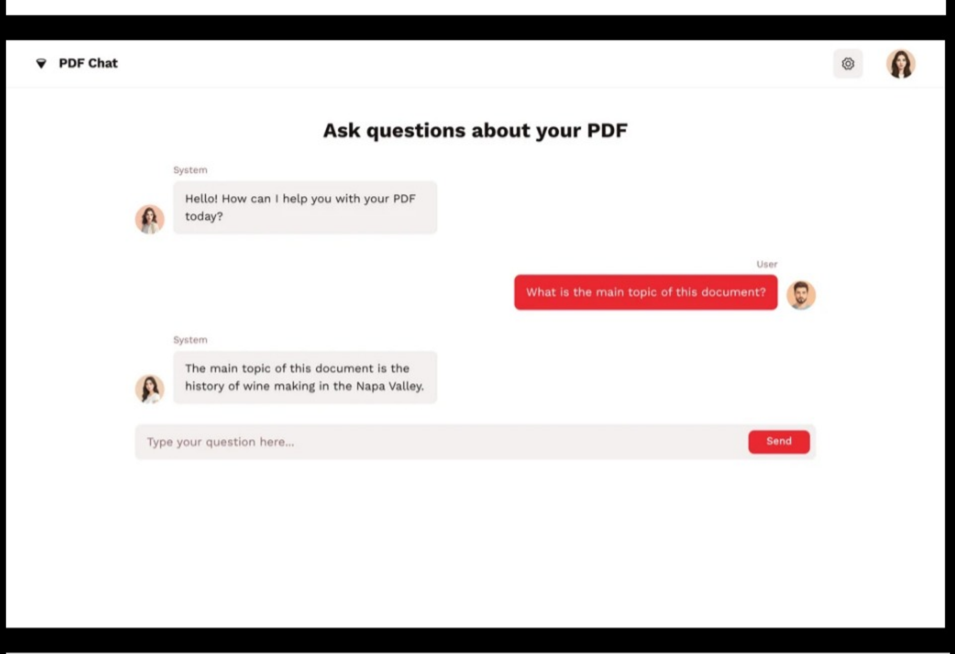
Interface & Usage Preview

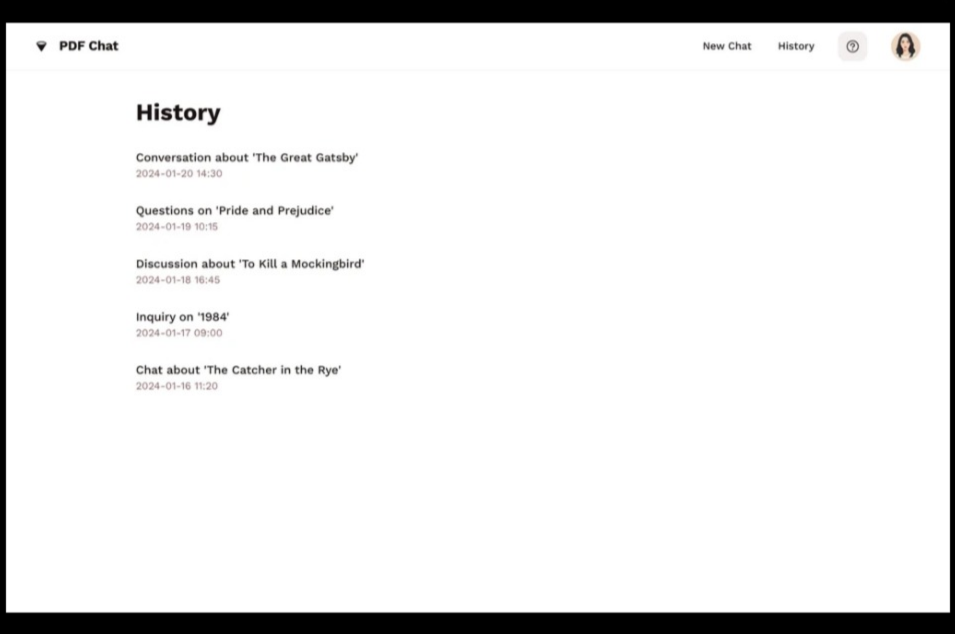
A glimpse of the chatbot in action with PDF inputs and responses











Results, Learnings, and Impact

The chatbot demonstrated strong performance on diverse PDF types—research papers, manuals, and policy documents. MiniLM2 ensured fast embedding generation and low inference cost, while LLAMA2 provided accurate, human-like answers grounded in context. Key insights include the importance of context chunking, prompt clarity, and the need for robust PDF text extraction pipelines.