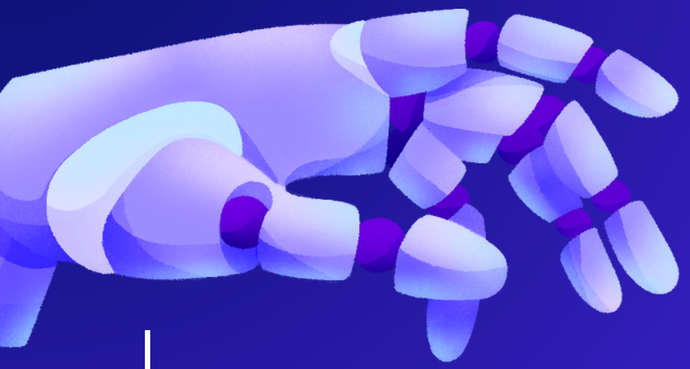
PDF CHATBOT WITH AI/ML

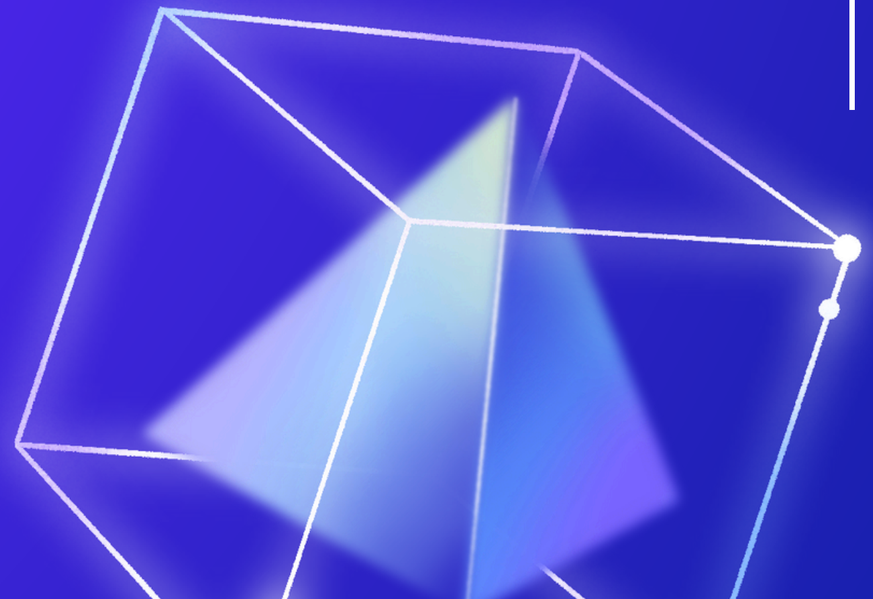
PROJECT

By Nourin Noushad



OBJECTIVE

- Develop a chatbot to answer user queries from PDF documents.
- Allow PDF upload and processing into a knowledge base.
- Enable real-time Q&A via a conversational interface.
- Deploy with Docker for easy reproducibility.



ARCHITECTURE OVERVIEW

- Frontend (Streamlit) ↔ Backend (FastAPI).
- Document upload and text extraction.
- ChromaDB/Embeddings for semantic search.
- LLM (Groq/OpenAI) for generating responses.

TECH STACK

- Python 3.11
- FastAPI (backend API)
- Streamlit (frontend UI)
- ChromaDB (vector database)
- Sentence Transformers (embeddings)
- Groq / OpenAI LLM (chat responses)
- Docker & Docker Compose

NLP APPROACH

01 Extract text from uploaded PDFs.

02 Split text into chunks for indexing

03 Generate embeddings using Sentence Transformers.

04 Store vectors in ChromaDB

05 Retrieve relevant chunks based on query.

06 LLM generates final contextual answer.

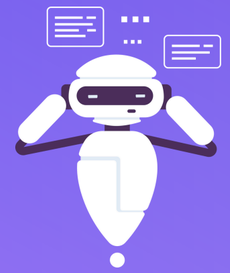
USER INTERFACE

OBJECTIVE 01

Upload PDF section.

OBJECTIVE 02

Chat window with conversational responses.



OBJECTIVE 03

Summaries, keywords, and table of contents.

DOCKER SETUP

- Dockerfile.api → Backend container.
- Dockerfile.frontend → Frontend container.
- docker-compose.yml → Runs both services.
- Environment variables in .env file.
- Single command: docker compose up --build

CHALLENGES FACED

- Large PDF token limits in LLMs (handled with chunking).
- Ensuring fast query response time.
- Docker dependency resolution and networking.
- UI enhancements for professionalism.

DEMO STEPS

1. START BACKEND (FASTAPI)
2. START FRONTEND (STREAMLIT)
3. UPLOAD PDF
4. ASK QUESTIONS IN CHAT
5. SHOW SUMMARIES, KEYWORDS, AND TABLE OF CONTENTS
6. DOCKER DEPLOYMENT RUN

ADVANTAGES

01 Fast Q&A: Instantly get answers from PDFs.

02 Multi-Feature: Summaries, keywords, table of contents.

03 Context-Aware: Answers based only on document content.

04 User-Friendly: Simple web interface for all users.

05 Portable: Dockerized for easy deployment.

06 Scalable & Extensible: Handles large documents; easy to upgrade LLM or embeddings.

CONCLUSION

- Implemented a functional PDF-based chatbot.
- Used NLP + embeddings + LLM for contextual answers.
- Simple, professional UI with Streamlit.
- Containerized with Docker for easy deployment.
- Ready for extension and production scaling

THANK YOU!