PDF-QA-Chatbot: Building a Retrieval-Augmented Chatbot for Private PDF Documents

Demo App

A live demo of the chatbot is available here:
https://pdf-app-chatbot-4ovbnutopxatdm8vevdtpk.streamlit.app/

Project Motivation

The goal of this project was to create a chatbot that can answer questions using the content of private PDF documents. This is a highly practical use case for companies, researchers, and individuals who need to extract value from internal documentation, CVs, business plans, or any proprietary PDF resource. Unlike generic chatbots, which only know public data, this assistant can "read" and answer based on your own files.

What Was Built

- A **Streamlit web app** where users can upload PDFs, index them, and ask questions.
- A **modular backend** that:
  - Extracts text from PDFs (using PyMuPDF)
  - Splits and embeds the text (using SentenceTransformers)
  - Indexes embeddings for fast retrieval (using FAISS)
  - Uses OpenAI's GPT (or a local LLM) to generate answers, given the relevant context
- A **reset function** to clear previous documents and cache, ensuring fresh indexing.

Technical Approach

- **Text Extraction:** Each uploaded PDF is parsed page by page. The text is concatenated and split into manageable chunks.
- **Embeddings:** Each chunk is converted into a dense vector using a pre-trained SentenceTransformer (all-MiniLM-L6-v2).
- **Vector Search:** All vectors are stored in a FAISS index. When a user asks a question, the query is embedded and the most similar document chunks are retrieved.
- **Answer Generation:** The top-k relevant chunks are concatenated and sent as context to an LLM (default: OpenAI GPT-3.5-turbo), which generates a natural language answer.
- **Interface:** The app is built with Streamlit for easy use and rapid prototyping.

What Actually Works

- The app runs locally and on cloud platforms (Streamlit Cloud, Hugging Face Spaces), provided the OpenAI API key is set as a secret.
- PDF upload, indexing, and question-answering work as intended for most text-based PDFs.
- The reset button reliably clears the document and embedding cache, preventing old documents from contaminating new queries.
- The system catches and displays errors (e.g., OpenAI API issues, context length limits).

What Doesn't Work (Yet) / Limitations

- **Scanned PDFs or images:** The system does not perform OCR, so it cannot extract text from image-based PDFs.
- **Long context:** If too many or too large chunks are included, the OpenAI API will reject the prompt for exceeding token limits. This was mitigated by limiting the number and size of retrieved chunks.
- **Evaluation:** No formal quantitative evaluation (e.g., F1 score, exact match) was performed. Testing was done manually with a handful of PDFs and questions.
- **No conversation memory:** The bot answers each question independently, without dialogue history.
- **No fine-tuning:** All models used are off-the-shelf; no domain adaptation or supervised training was performed.

Key Lessons and Observations

- **RAG is powerful:** Even with simple chunking and retrieval, the chatbot can answer specific questions that a generic LLM cannot.
- **Context management is crucial:** Prompt size must be carefully managed to avoid API errors and maximize answer quality.
- **User experience matters:** Features like the reset button and clear error messages greatly improve usability, especially when iterating on document sets.
- **Deployment is easy but not trivial:** Streamlit Cloud makes sharing the app simple, but handling secrets and file system quirks (like cache directories) is essential.

Possible Improvements

- Automated evaluation: Build a small benchmark of Q&A pairs for systematic testing.
- OCR integration: Add Tesseract or similar for scanned PDFs.
- Smarter chunking: Use sentence/paragraph boundaries, or dynamic chunk sizes based on token count.
- User authentication: For more sensitive use cases.
- Conversation memory: Allow follow-up questions and dialogue context.
- Support for more LLMs: Integrate with local models for cost/privacy.

Conclusion

This project demonstrates that it is feasible to build a practical, user-friendly RAG-based chatbot for private PDFs with open-source tools and a commercial LLM API. The approach is robust for text-based documents and provides immediate value for anyone needing to query their own files in natural language. With further work on evaluation, chunking, and OCR, this system could be production-ready for business or research use.