Mindgrasp PRD: Perfect PDF Context

Deadline: Friday, March 28th, 2025

📌 Project Overview

Current LLM-based systems often struggle to maintain high-fidelity context when dealing with multiple and varied knowledge sources—especially PDFs with inconsistent formatting (e.g., OCR scans, tables, multi-column layouts). The **Perfect PDF Context Chat Bot** solves this by enabling accurate, context-backed responses to questions based on any number or type of PDF documents.

Core Problem

- LLMs hallucinate when they don't understand what's truly important across documents.
- They struggle with:
  - OCR-scanned documents.
  - PDFs with non-linear layouts (multi-column, tables, footnotes, etc.).
  - Maintaining source attribution and relevancy.

Objectives

1. Build a **chat interface** that can answer user questions based strictly on uploaded PDFs.
2. Ensure responses are:
   - **Accurate**
   - **Traceable** (always cite exact source context)
   - **Concise**, only using relevant snippets.
3. Handle:
   - OCR and non-OCR PDFs.
   - Layout inconsistencies (columns, footnotes, headers).
   - Multiple PDFs across different topics.

Key Features

1. Multi-PDF Upload

- Drag-and-drop or select multiple PDFs.

- Preprocessing pipeline kicks off on upload.

2. Advanced Parsing + Chunking

- OCR support for scanned documents.
- Intelligent layout parsing:
    - Detects columns, headings, tables, footnotes.
    - Maintain reading order and logical flow.
- Break documents into context-aware “smart chunks”.

3. Semantic Chunk Indexing

- Use embedding models (e.g., OpenAI, Cohere, or local models like Instructor) to generate high-fidelity vector embeddings for each chunk.
- Store with metadata (document name, page number, position).

4. Context-Aware Retrieval

- Given a user query, retrieve only the **most relevant and minimal set** of chunks.
- Prevent context overload—only include what’s truly needed.

5. Grounded Response Generation

- LLM (e.g., GPT-4 or Claude) generates answers **strictly using** retrieved chunks.
- Each part of the answer is linked to its exact source (e.g., “PDF1, page 4”).

6. Citations and Transparency

- Inline citations (with hover or tooltip UI) to highlight where data came from.
- Option to click and view the original PDF segment side-by-side.

7. Chat History + Document Context

- Memory of current chat session and linked documents.
- Ability to refer to previous Q&A (optional v1.1+).

🧪 Stretch Goals

- Support audio/voice Q&A.
- Summarization of entire documents before querying.
- Learning goal alignment (quiz generation, study guides).
- Allow export of conversation + citations.

📐 Architecture Overview

1. **Frontend** (React):
   - PDF upload interface
   - Chat window with source references
   - Context viewer pane (for exact snippet references)
2. **Backend** (Flask or FastAPI):
   - PDF ingestion pipeline
   - OCR + layout analysis
   - Chunking + embeddings
   - Vector DB (e.g., Weaviate, Pinecone, or FAISS)
   - Query + response generation
   - Logging + metrics
3. **Storage:**
   - PDFs (Object Storage)
   - Vector Embeddings (Vector DB)
   - Metadata DB (e.g., Postgres)

✅ MVP Success Criteria

- Upload 1–10 diverse PDFs (including OCR).
- Ask a question that pulls relevant answers.
- Each part of the answer is backed by exact, visible source reference.
- Time-to-answer under 10 seconds.
- Minimal hallucinations outside the PDF context.
- Thoughtful UI/UX that is appealing and intuitive