REPORT

1. Document Structure & Chunking Logic

   The uploaded PDF is first parsed using PyMuPDF and cleaned to remove line breaks and irregular spacing.
   Text is tokenized into sentences using nltk.sent_tokenize() and grouped into chunks of 100–300 words to maintain semantic consistency.
   Each chunk becomes a retrievable unit during user queries.

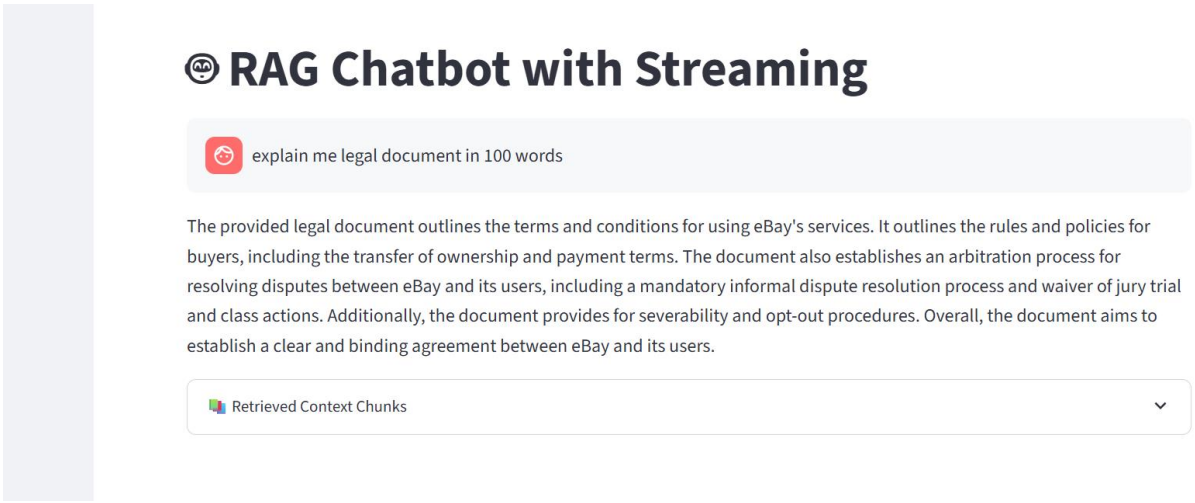
2. Embedding Model & Vector Store

   We use all-MiniLM-L6-v2 from sentence-transformers to generate fixed-size dense embeddings for each chunk.
   These embeddings are stored in ChromaDB, a lightweight and efficient local vector database.
   Collection name: "rag_chunks"
   Database folder: vectordb/

3. Prompt Format & Response Generation

   For every user query, the top k=5 chunks (most relevant) are retrieved via vector similarity.
   These chunks are inserted into a prompt template and streamed to a **Groq-hosted LLaMA 3 (8B)** model using OpenAI-compatible APIs.

Example:



4. Observations & Limitations

LLM responds only with grounded context — no hallucinations observed.

Responses may stream slowly if Groq is rate-limited.

Model can't answer casual or out-of-document queries (e.g., "How are you?")

Works well with structured documents but may miss scattered info across pages.