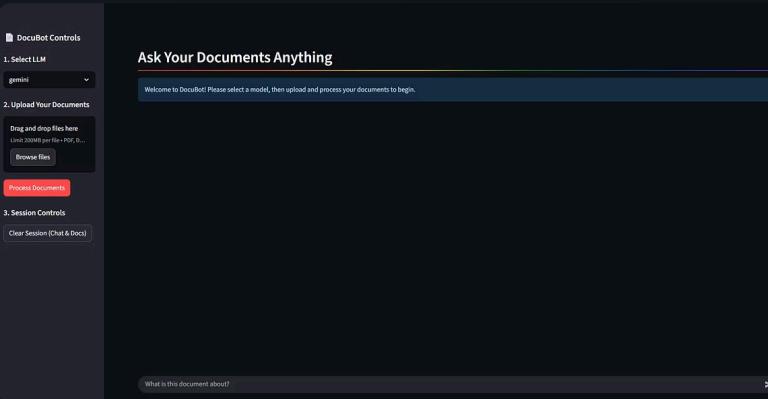


Ask Your Documents Anything



Agentic RAG Chatbot

Multi-Format Document QA using Model Context Protocol (MCP)

Technical Implementation Overview

Project Architecture

The application uses a decoupled architecture with a FastAPI backend (the 'brain') and a Streamlit frontend (the 'face').

User Interface

Streamlit frontend for user interaction and file uploads

Backend API

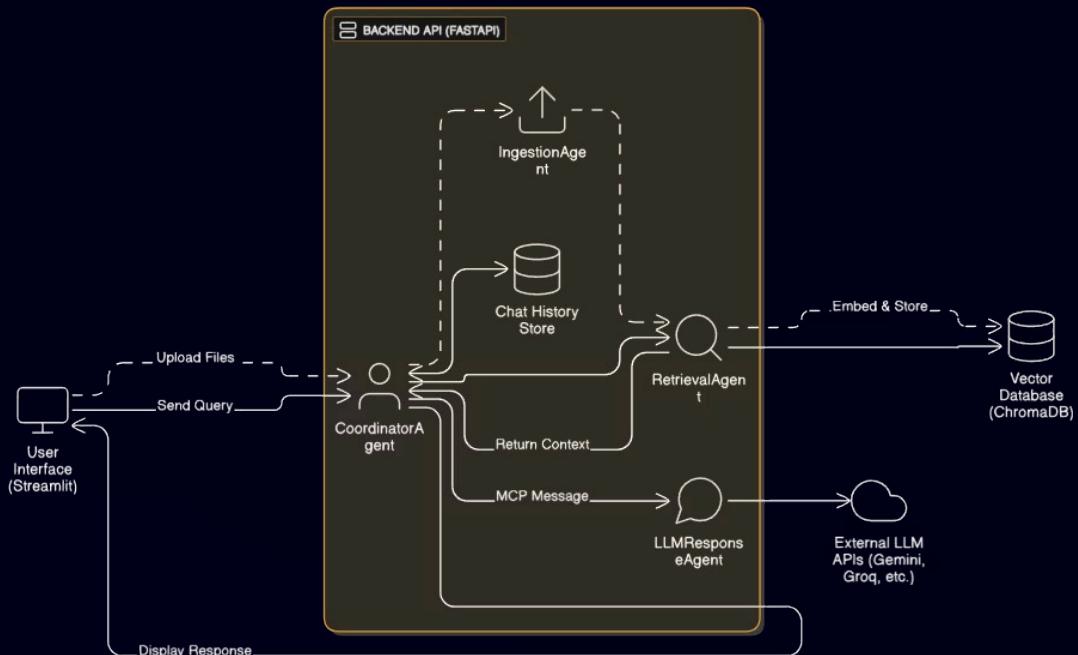
FastAPI CoordinatorAgent with three specialist agents

Vector Database

ChromaDB stores document embeddings for semantic search

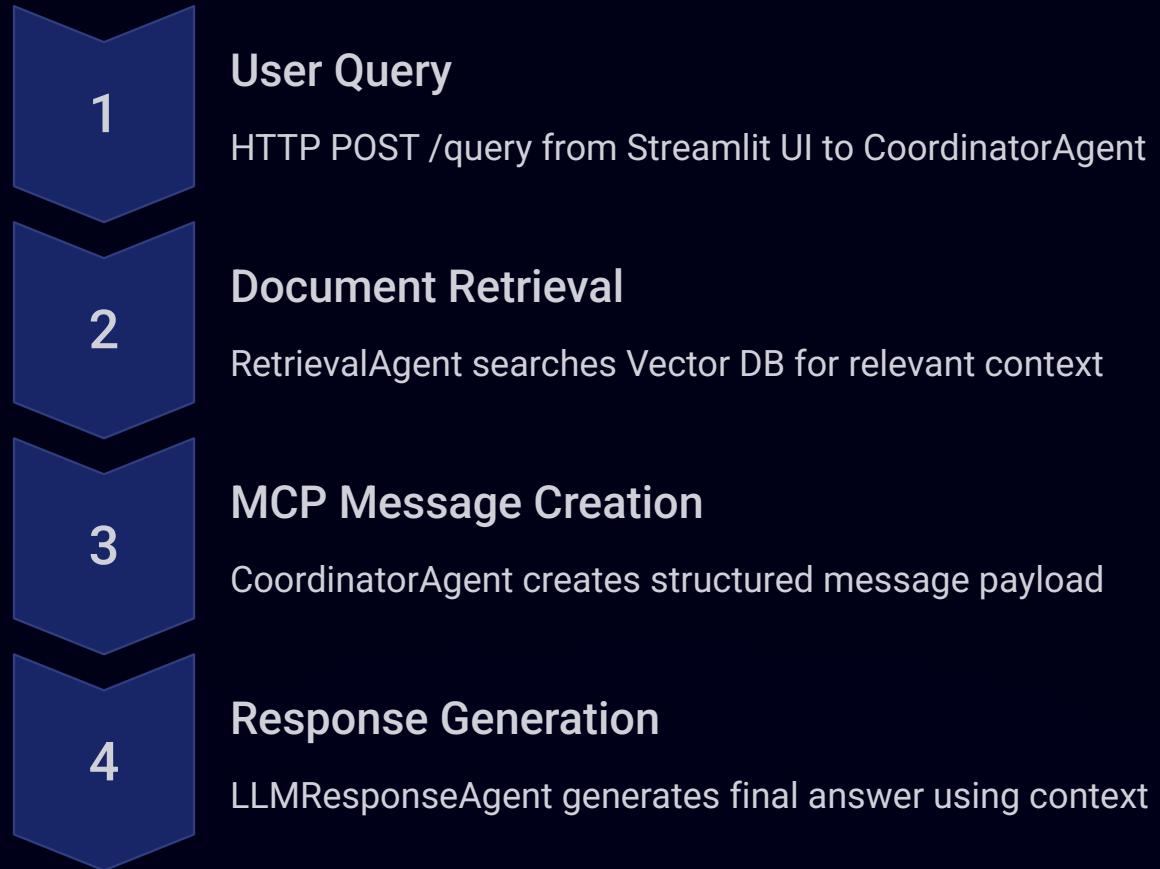
External LLM APIs

Pluggable interface for Gemini, Groq, and Hugging Face

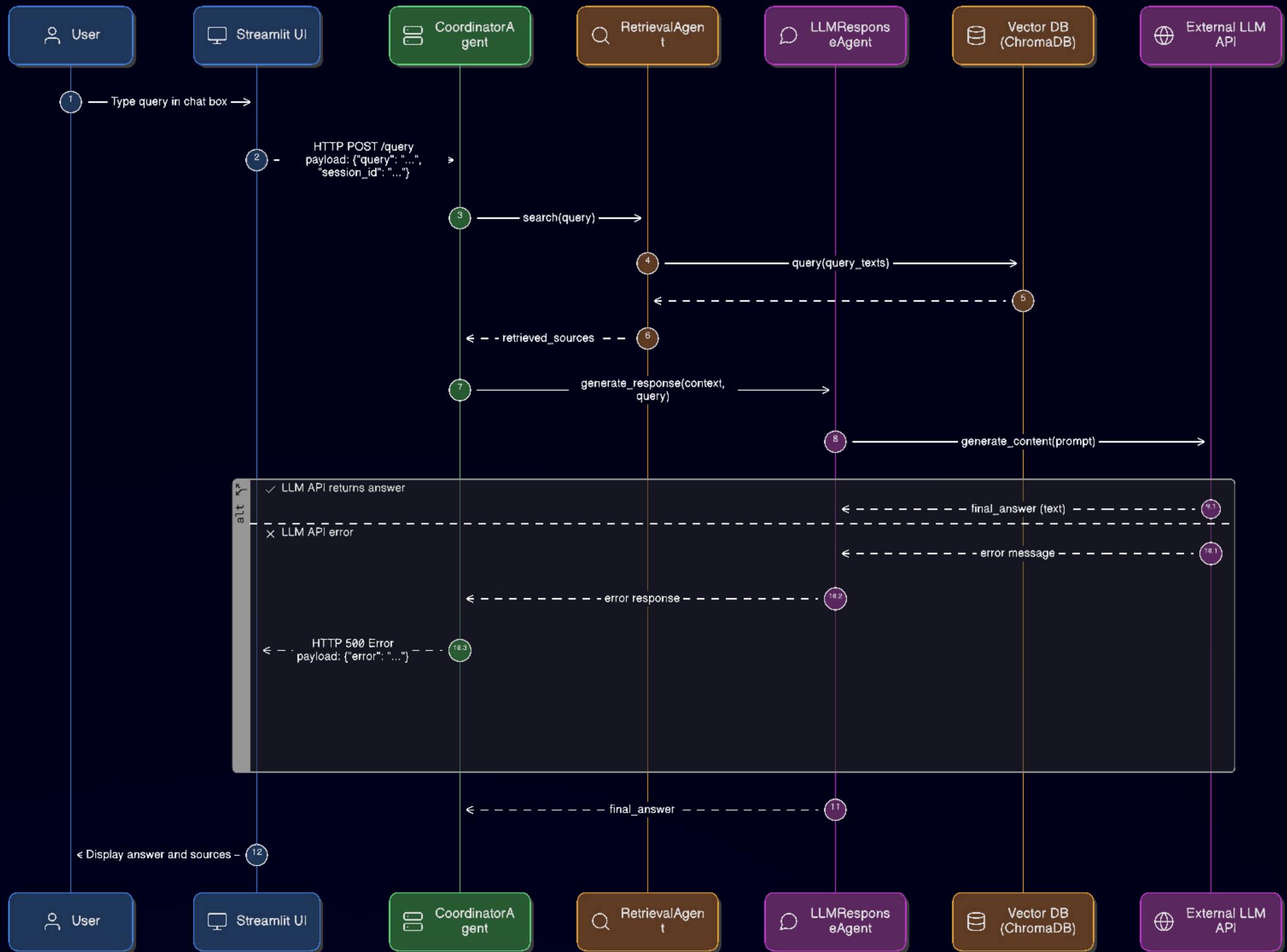


System Flow & Message Passing

The lifecycle of a user query involves coordinated communication between agents using MCP messages.



```
{  
  "sender": "CoordinatorAgent",  
  "receiver": "LLMResponseAgent",  
  "type": "CONTEXT_RESPONSE",  
  "payload": {  
    "retrieved_context": "[...]",  
    "query": "..."  
  }  
}
```



Technology Stack

Tech Stack

- Backend: FastAPI, Uvicorn
- Frontend: Streamlit
- Vector DB: ChromaDB (in-memory)
- Embeddings: SentenceTransformers
- LLMs: Google Gemini, Groq, Hugging Face
- Core Libraries: Pydantic, LangChain
- Package Manager: uv



Live application interface with document processing and chat functionality

Application Demo

The screenshot displays a user interface for a document processing application, likely named DocuBot, shown across three panels.

DocuBot Controls Panel: This panel is on the left and contains the following sections:

- 1. Select LLM:** A dropdown menu currently set to "gemini".
- 2. Upload Your Documents:** Includes a "Drag and drop files here" area with a 200MB limit, a "Browse files" button, and a "Process Documents" button.
- 3. Session Controls:** Buttons for "Clear Session (Chat & Docs)" and "What is this document about?".

Drag and drop files here Panel: This panel is in the center and shows a file named "safi.txt" (2.9KB) uploaded via drag-and-drop. It features a "Process Documents" button and a confirmation message: "✓ Documents Processed!".

Ask Your Documents Anything Panel: This panel is on the right and displays the results of an AI query. The question was "what are the skills of the person in the file". The response includes:

- 1. Languages:**
 - Python
 - Java
 - C
 - HTML/CSS
 - JavaScript
- 2. Developer Tools:**
 - Git
 - Github
 - VS Code
- 3. Frameworks:**
 - React
 - Node.js
 - Express
 - Next.js
 - Tailwind
- 4. Artificial Intelligence:**
 - ML
 - NLP

- o PostgreSQL
 - o MySQL
6. Other skills:
- o Generative AI
 - o Deploying and managing generative AI applications on Oracle Cloud Infrastructure (OCI)
 - o Database management: CRUD APIs, query writing, optimization, and troubleshooting
 - o Problem-solving and algorithms
 - o Software development: DSA, data structures, and systems design

▼ View Sources



Error from backend: {"detail": "The daily API quota for the selected model has been exceeded. Please try again tomorrow or switch models."}

Key Challenges & Solutions

Retrieval Failure

Challenge: Queries failed to find relevant chunks due to semantic differences.

Solution: Increased top_k retrieved documents for broader context.

API Rate Limiting

Challenge: Gemini API quota exhaustion caused backend crashes.

Solution: Implemented robust error handling with user-friendly messages.

Prompt Engineering

Challenge: LLM confusion between task instructions and document content.

Solution: Clear persona definition and strong context delimiters.

Future Improvements



Persistent Storage

Upgrade to disk-based ChromaDB and Redis cache for production-ready stateful application.



Advanced Retrieval

Implement Multi-Query Retrieval with LLM-generated question variations for resilient search.



Asynchronous Processing

Move large document ingestion to background task queue for non-blocking user experience.

