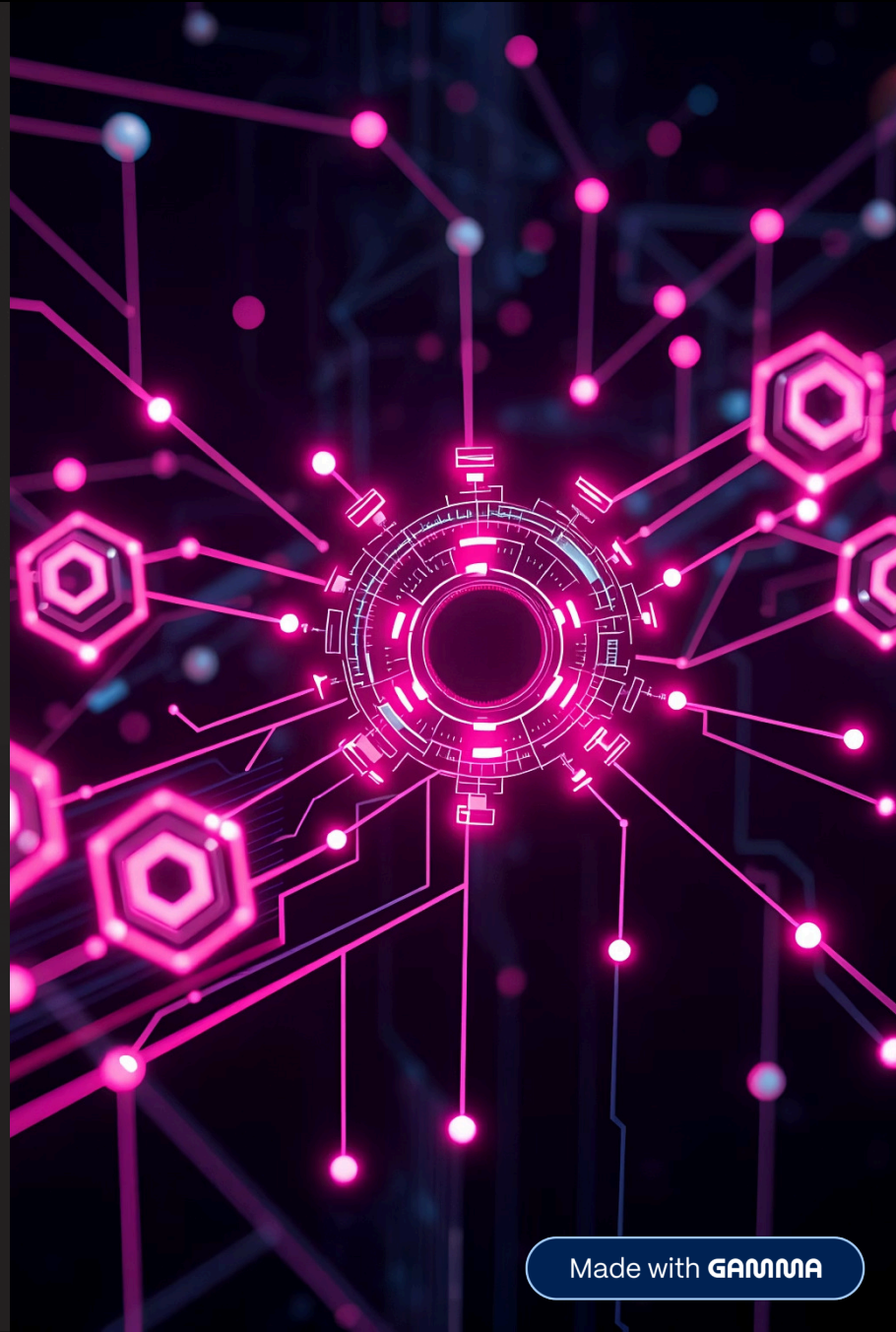
📘 Multi-Agent AI Q&A: Enhanced Traceability with MCP

Pioneering a Multi-Agent RAG Chatbot utilizing Langchain, Streamlit, and ChromaDB for advanced querying.



🔧 Agent-Based Architecture with MCP

1

IngestionAgent

Parses and indexes uploaded files into a vector database for efficient retrieval.

2

RetrievalAgent

Fetches the most relevant contextual information from ChromaDB based on user queries.

3

LLMResponseAgent

Generates the final, coherent answers by synthesizing information from the LLM and retrieved context.

4

MCP (Message Passing System)

Tracks the step-by-step flow and interactions between all agents for transparent tracing and debugging.

🔄 System Flow Diagram

1

User Uploads Document

Supports various formats (PDF, DOCX, PPTX, CSV) for flexible data ingestion.

2

IngestionAgent Processes

Parses content, extracts text, and stores vector embeddings in ChromaDB.

3

Query to RetrievalAgent

Identifies and retrieves the most relevant document chunks based on semantic similarity.

4

LLMResponseAgent Generates

Crafts a comprehensive and accurate reply using the LLM and retrieved data.

5

MCP Logs & Displays

Provides a real-time, step-by-step trace of agent activities for full transparency.

🧰 Tech Stack Used

Frontend & Backend

- Streamlit for interactive user interface.
- Python as the core programming language.
- LangChain for orchestrating agent interactions.
- Google Generative AI for powerful language models.

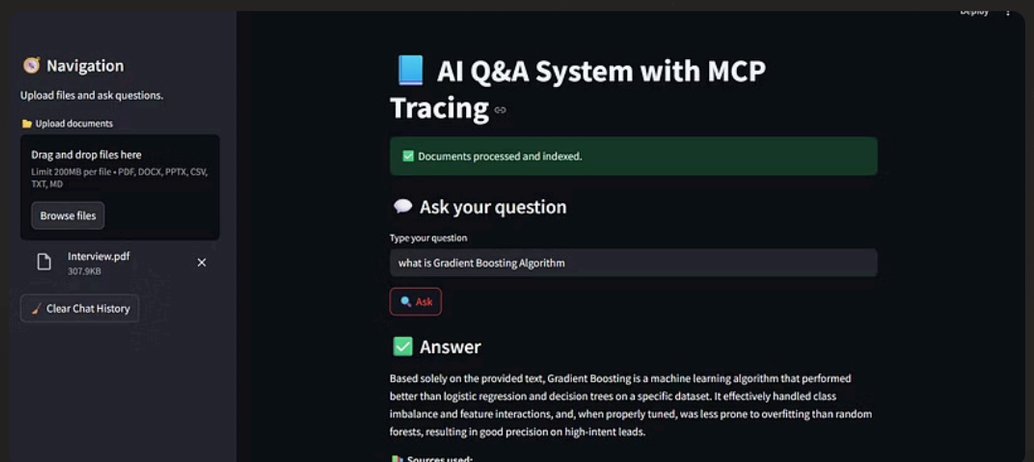
Data & Parsing

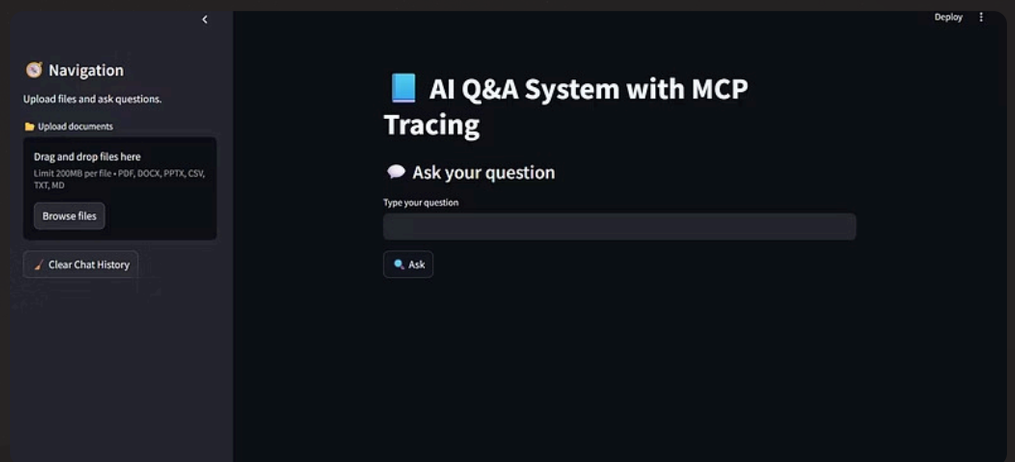
- ChromaDB as the vector store for efficient semantic search.
- PyMuPDF for PDF parsing.
- python-docx for Word document processing.
- python-pptx for PowerPoint file handling.
- pandas for tabular data (CSV) ingestion.

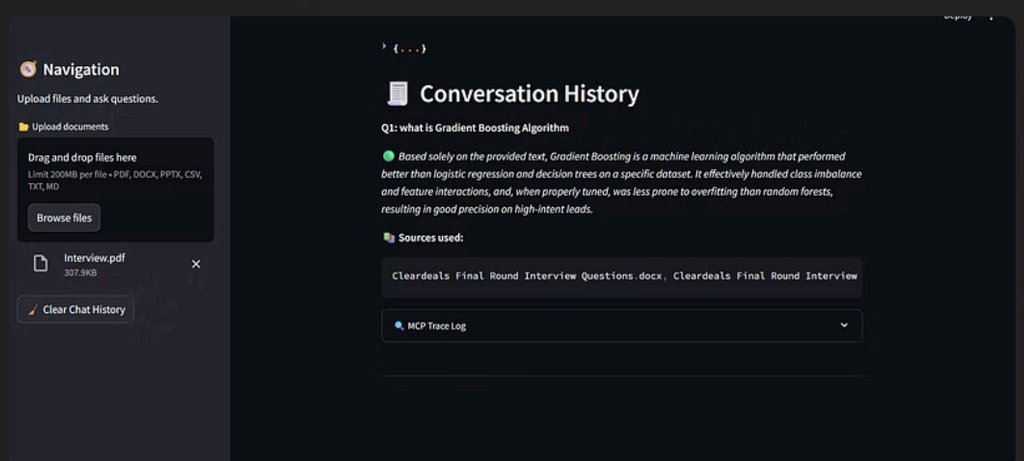
Tracing

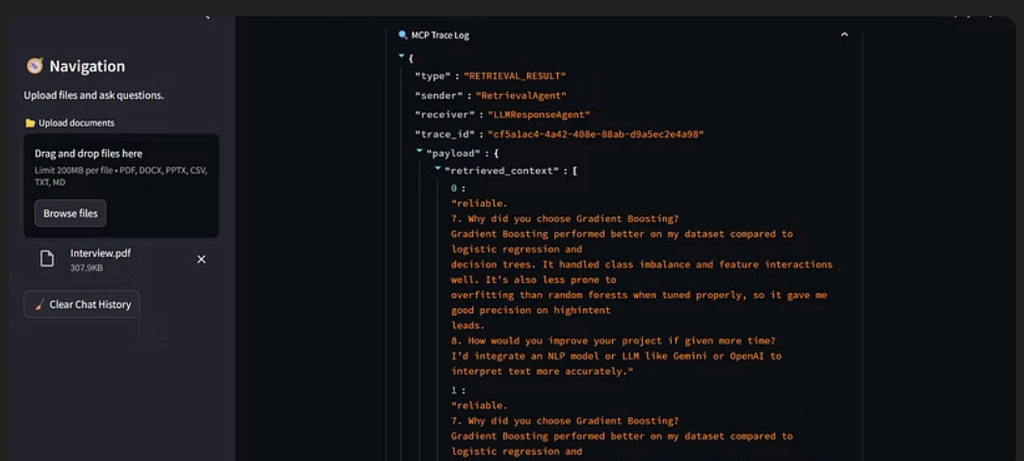
- Custom MCP (Message Control Plane) for detailed debugging and workflow visibility.

This card showcases a screenshot of the working application that was developed as part of the project. The screenshot provides a visual representation of the user interface and the key features implemented in the application.









🚧 Challenges & 🔭 Future Scope

Current Challenges

- Ensuring consistent parsing across diverse document formats (PDF, DOCX, PPTX, CSV).
- Optimizing agent communication and coordination within Langchain framework.
- Developing a dynamic and user-friendly display for real-time MCP logs in the Streamlit UI.

Future Enhancements

- Integrating conversational memory to maintain context across turns and implement lannggraph for dynamic communication between Agents.
- Implementing robust authentication and authorization mechanisms.
- Exploring voice integration for a more natural user interaction experience.