The Rag System

Using python& Ollama

Mr. Krushna Kapkar
Mr. Rutik Kadam
Mr. Mukul Bhagat
Mr. Athrava waghmare
Mr. Vipin Chaudhari

Introduction

- 1.RAG System: It’s Combines document retrieval and generation for more accurate, context-aware responses.

- 2.Python: Use a retriever to fetch documents and a generator to create answers.

- 3. Ollama: Integrates with models like Llama2 to power retrieval and generation seamlessly.

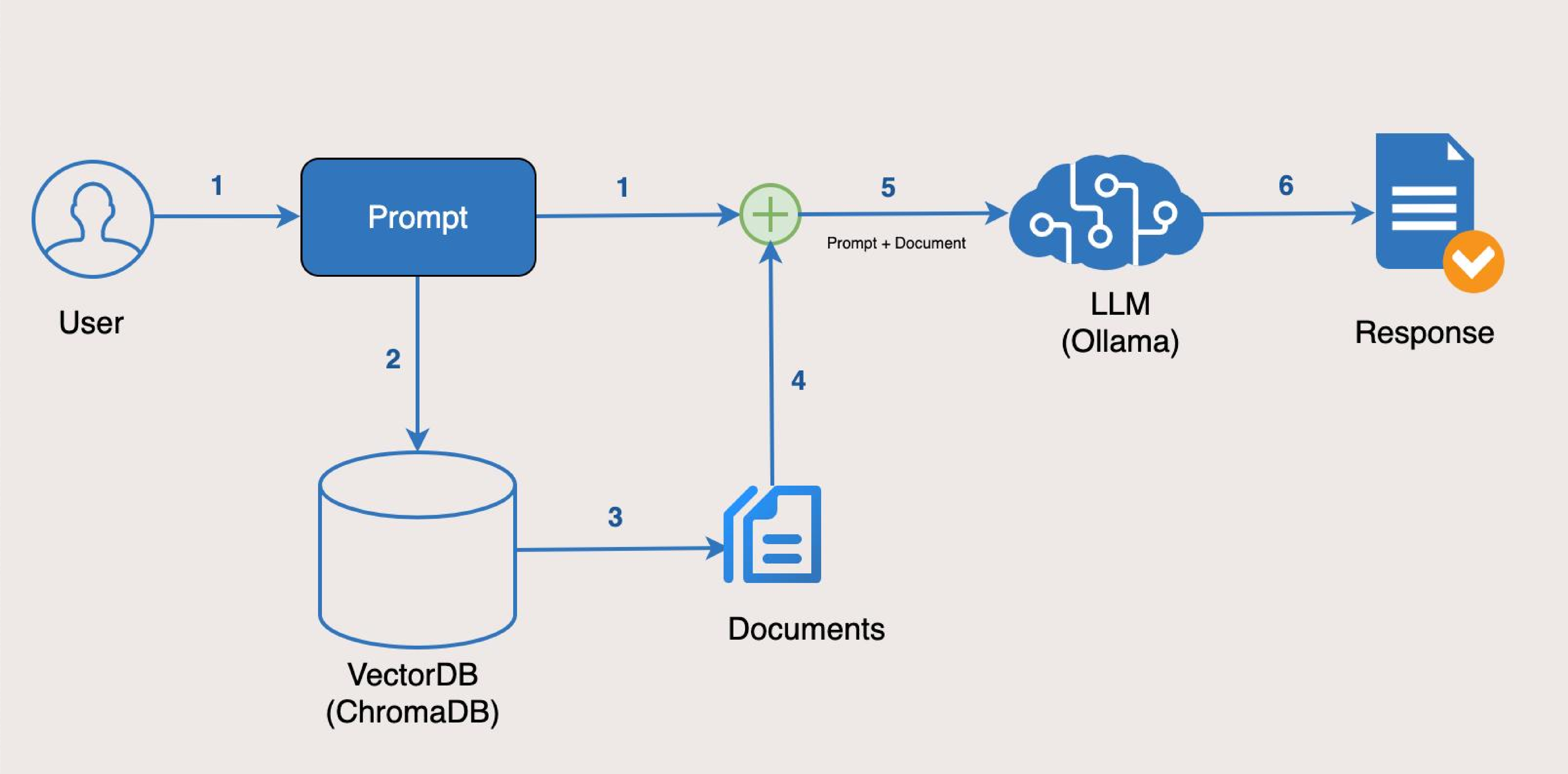
Problem description & goal

- **Problem:** Traditional language models lack access to up-to-date external information, limiting their response accuracy.
- **Goal:** Enhance models by combining retrieval of relevant data with generation for more accurate, context-aware answers.
- **Solution:** Implement a RAG system using Python and Ollama for efficient retrieval and generation.

Dataset Description

- Dataset Description:
- For our RAG system, we utilized a mix of PDFs, JSON files, and text files as the data sources. These files were directly uploaded and used for retrieval, allowing the system to extract relevant information without conversion, ensuring efficient context generation.

Block Diagram



https://medium.com/@arunpatidar26/rag-chromadb-ollama-python-guide-for-beginners-30857499d0a0

Approach

Upload and Search Documents: You can add different types of files (PDFs, JSON, text) to a smart database. When you ask a question, the system finds the most relevant information from these files.

Combine with Your Question: The system takes both your question and the relevant information from the files to create a detailed and meaningful response.

Generate Smart Answers: The AI (Ollama) uses this combined information to give a clear and accurate response.

Keep Getting Better: Over time, the system improves by learning from past questions and answers, making responses more useful and precise.

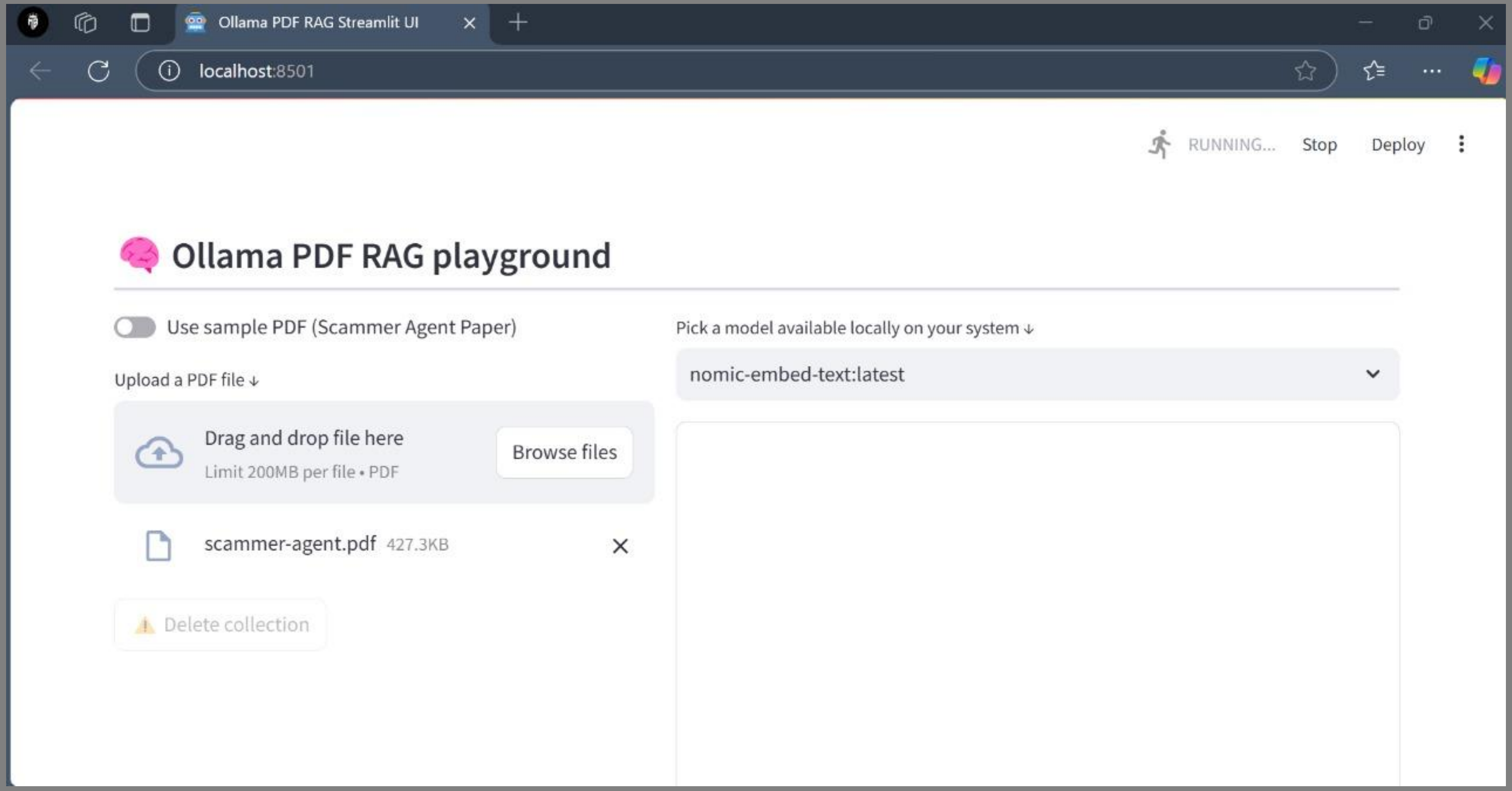
Advantages

1.Improved Decision-Making: The system helps make accurate and quick decisions by analyzing data in real-time.

2.Enhanced Efficiency: Automates processes, saving time and reducing manual errors.

3.Scalability: Can easily handle larger amounts of data and adapt to growing needs without major changes.

Results (continued)



Ollama PDF RAG playground

Use sample PDF (Scammer Agent Paper)

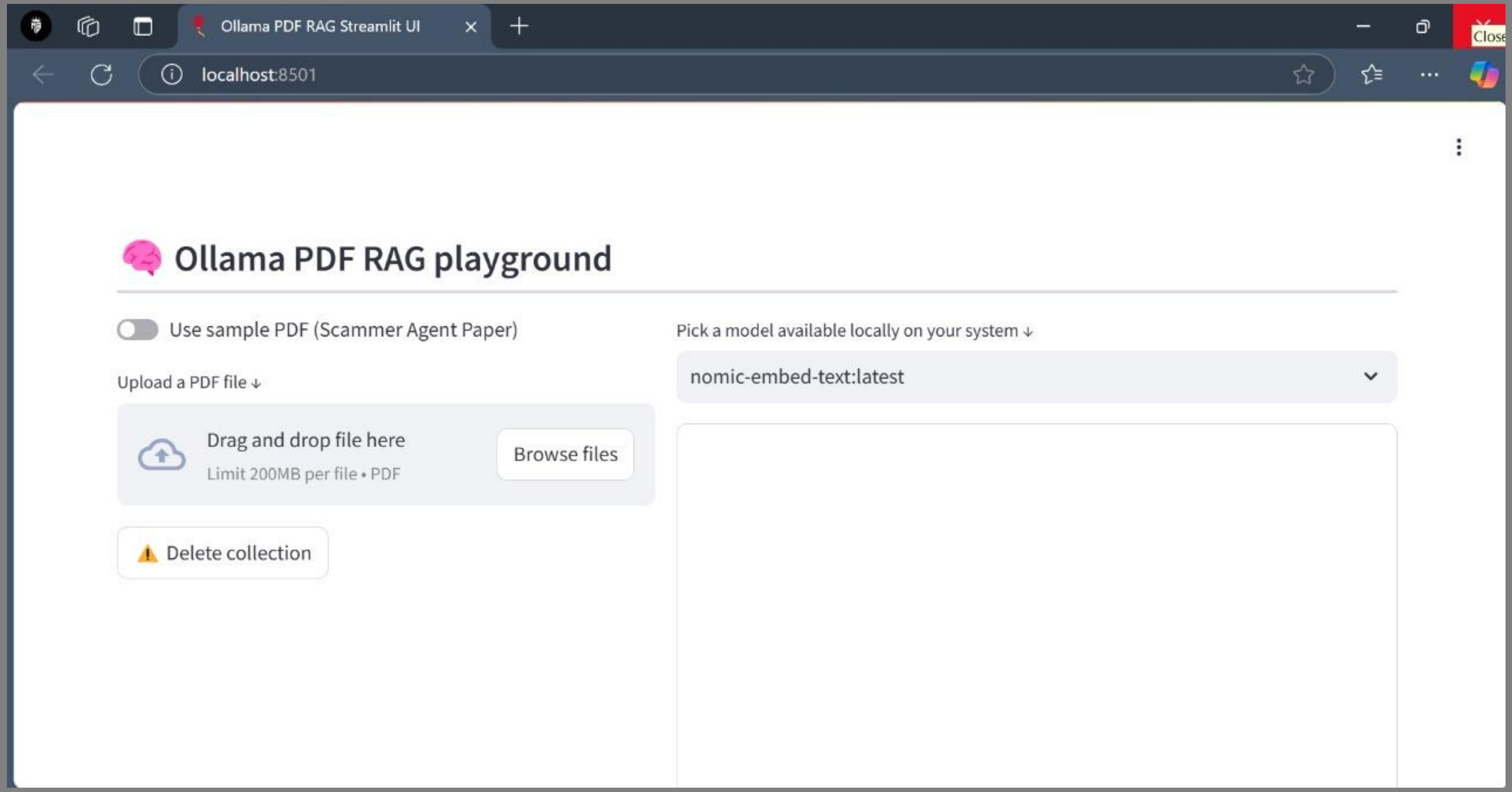
Pick a model available locally on your system ↓

nomic-embed-text:latest

Upload a PDF file ↓

Drag and drop file here
Limit 200MB per file • PDF

scammer-agent.pdf 427.3KB



Ollama PDF RAG playground

Use sample PDF (Scammer Agent Paper)

Pick a model available locally on your system ↓

nomic-embed-text:latest

Upload a PDF file ↓

Drag and drop file here
Limit 200MB per file • PDF

Obtained results

Results

The accuracy of our RAG system is approximately 80%, based on the relevance of the retrieved documents and the quality of the generated responses. This performance was evaluated by measuring retrieval precision and the coherence of answers generated by the Ollama model.

Applications

1. Healthcare: Analyzes patient data for quicker diagnosis and treatment decisions.
2. Finance: Tracks market trends for real-time investment insights.
3. Supply Chain: Optimizes inventory and predicts demand to reduce costs.
4. Customer Support: Automates query handling for faster responses.
5. Smart Cities: Manages traffic and energy systems for better urban planning.

Conclusion

The RAG System, developed using Python and Ollama, effectively integrates real-time data processing and decision-making. By leveraging advanced algorithms, it streamlines operations and enhances accuracy in predictive analytics. This project demonstrates the power of combining modern tools for building efficient and scalable solutions, paving the way for future advancements in intelligent systems.

References

https://youtu.be/Oe-7dGDyzPM?si=FojsXhx0ussFwTes

Repository

Chatgbt

THANK YOU