University of Sri Jayerardenepura

Progress Evaluation II

Knowledge Graph-based Retrieval-Augmented Generation System for Domain-Specific Information Extraction with Glossary-Aided Responses

Group 15

Meet Our Team ↘

Kavinda Maduranga

WEERASINGHA W.G.K.M.
ICT/20/956

Nipuni Nishadini

De Silva K.N.N.C.
ICT/20/826

Dulan Jayawikrama

Jayawickrama D.S.K.
ICT/20/862

Our Supervicers

↘

Main Supervisor

Dr. Chamara Liyanage
Academic Supervisor,
Department of ICT,
University of Sri Jayewardenepura

External Supervisor

Mr. Hiran Wijesingha
Assistant Director IT,
Sri Lanka Tea Board

Content Outline ↘

Introduction & Research Problem

Organizations with extensive histories struggle to access vital information in their archives, while outdated manual systems risk losing expertise as senior staff retire. This creates a knowledge gap for newer employees who lack historical context for decision-making. An automated system is urgently needed to preserve and make institutional knowledge accessible for future use.

AI Powered System

Development of a system for interacting domain-specific knowledge.

Graph-RAG Model

Retrieval-Augmented Generation with a knowledge graph.

Web Portal

Includes a user-friendly document management portal for easy interaction.

Domain-specific Glossary

Industry-specific terms will be integrated to improve the accuracy of the AI-generated response.

Objectives & Outcomes ↘

Main objective

Develop a graph RAG system integrated to improve organizations' ability to manage, retrieve, and utilize historical documents with the help of an externally managed domain-specific glossary. This system will aid decision-making by providing accurate, contextually relevant information based on of documented institutional actions, decisions, and domain-specific knowledge.

- Web-based document management portal
- LLMs integration for context-aware responses
- Structured knowledge base from historical document
- Domain-specific glossary
- User-friendly interface
- Systems future needs (Scalability and adaptability)

Methodology ↘

1 Data Collection and Preprocessing

- Document Management Portal
- Text Extraction

2 Knowledge Graph Creation

- Knowledge Graph Construction
- Automation

3 Graph based Context Retrieval

- Context retrieval by graph traversal

4 Integration with LLM and Glossary

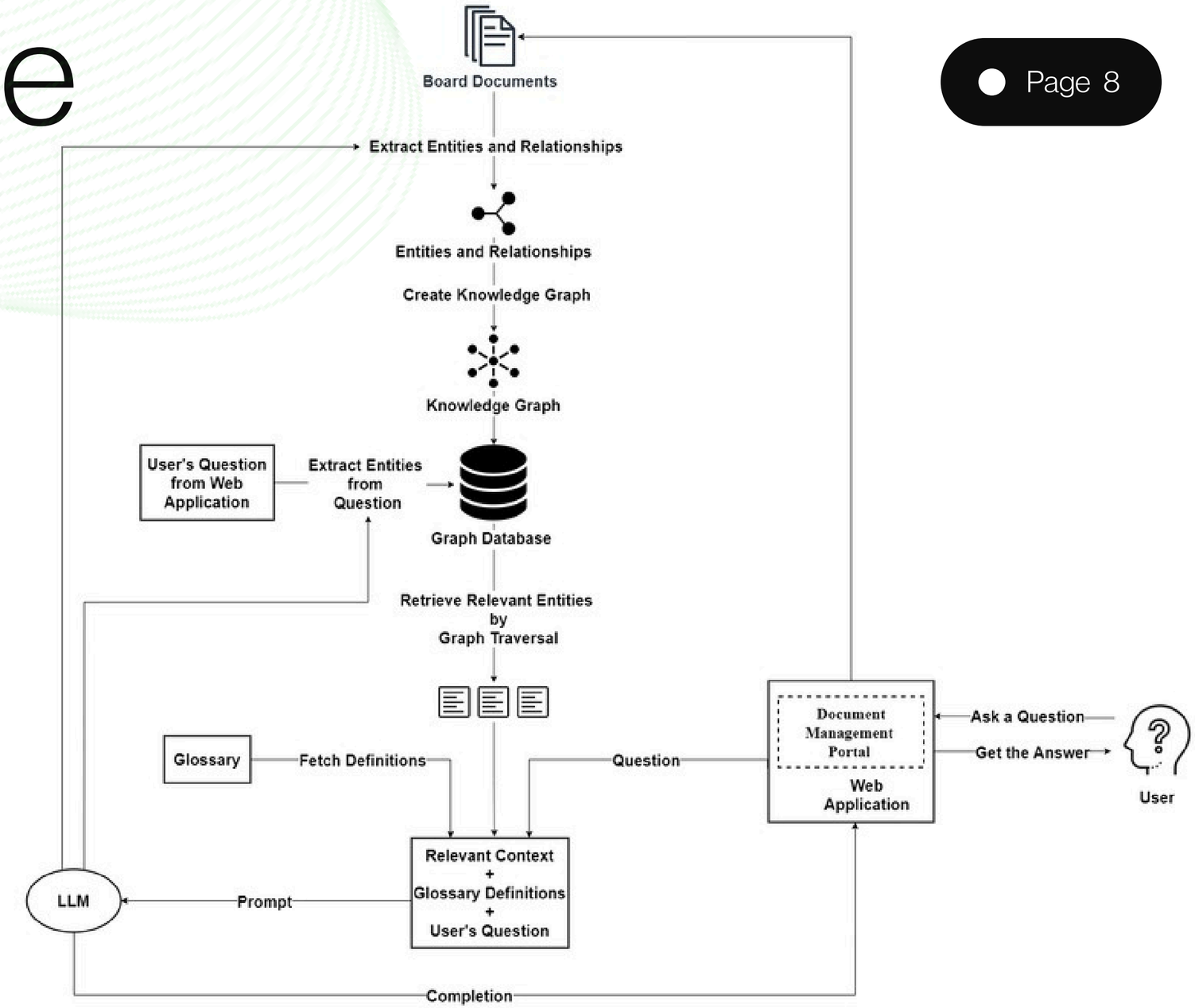
- LLM Integration
- Glossary Integration

5 Answer Generation and Delivery

- LLM Processing
- Frontend Delivery

Architecture Design

↘



Technologies ↘

neo4j

Graph database management system designed to store and query connected data efficiently.

LangChain

A framework for building applications that integrate LLMs with external data sources.

mongoDB.

NoSQL database that stores data flexibly in JSON-like documents, ideal for modern, scalable applications.

Azure

A cloud platform for building, deploying, and managing applications globally. It offers scalable and secure services for various needs

React

Popular JavaScript library for building interactive user interfaces, especially for web applications.

Flask
web development, one drop at a time

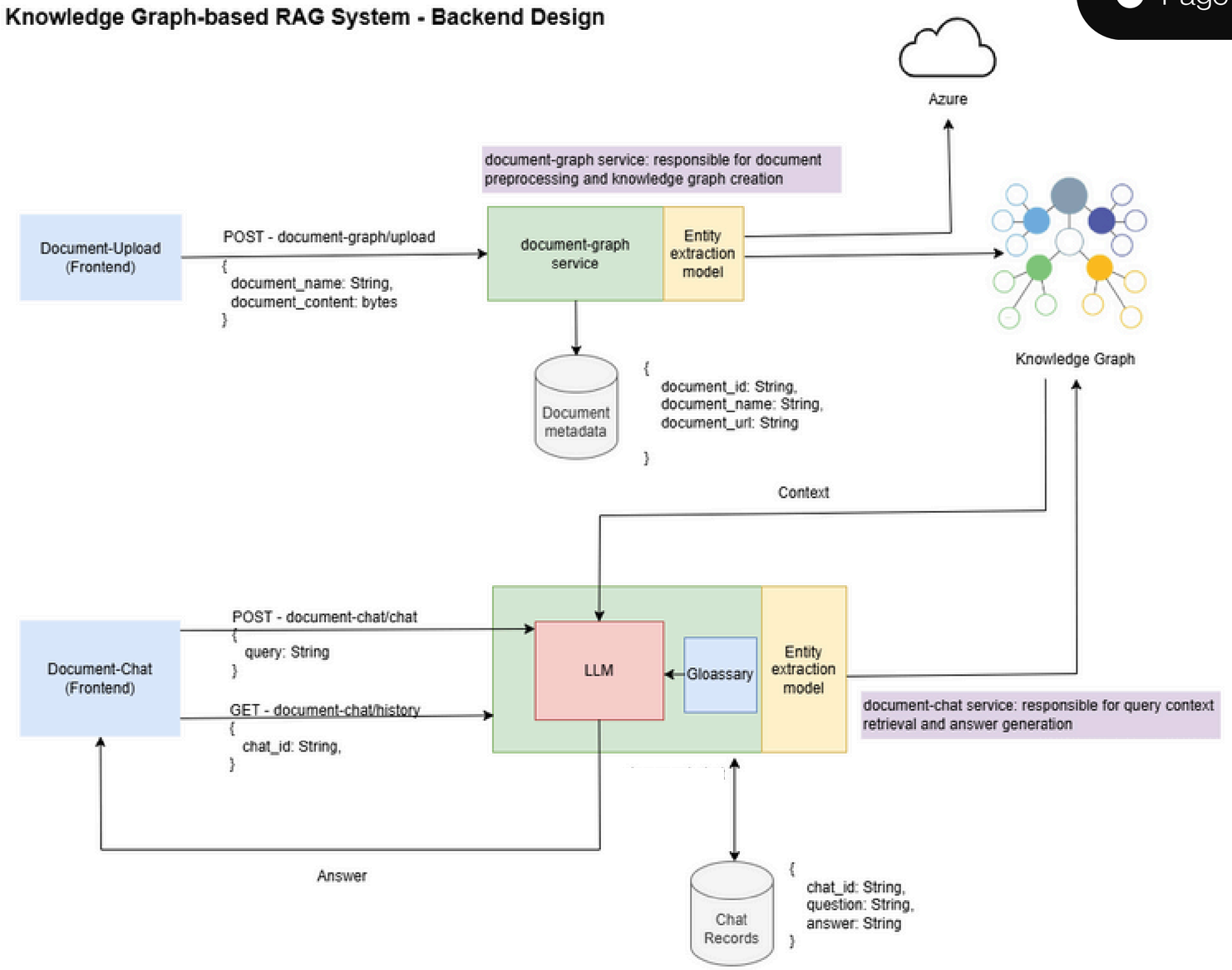
Modern, high-performance web framework for building APIs with Python, based on standard Python type hints.

Llama 3.1

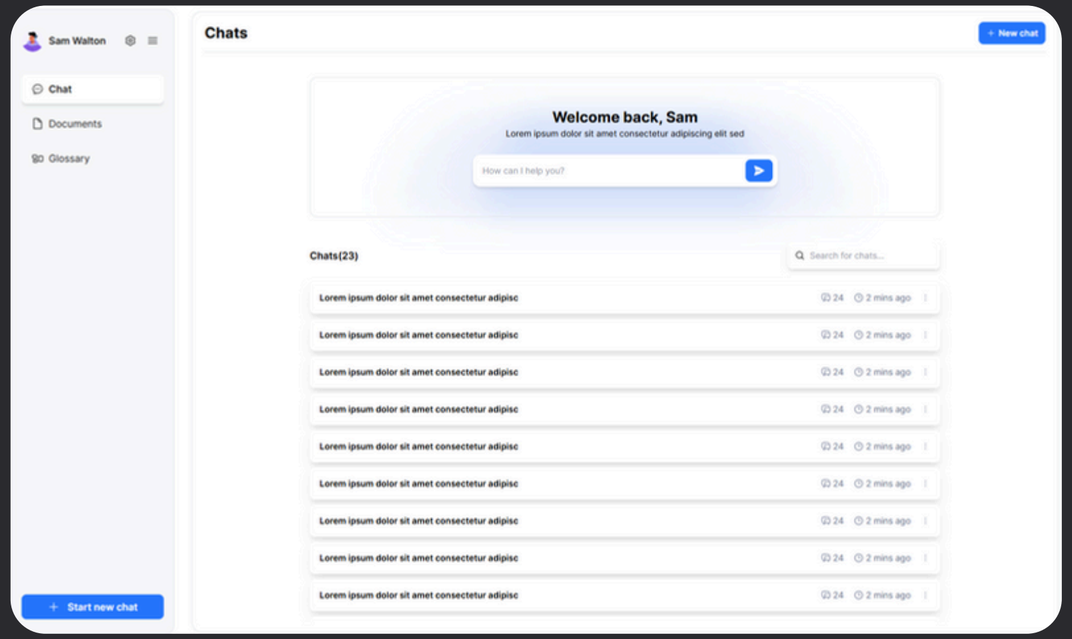
Gemma 3

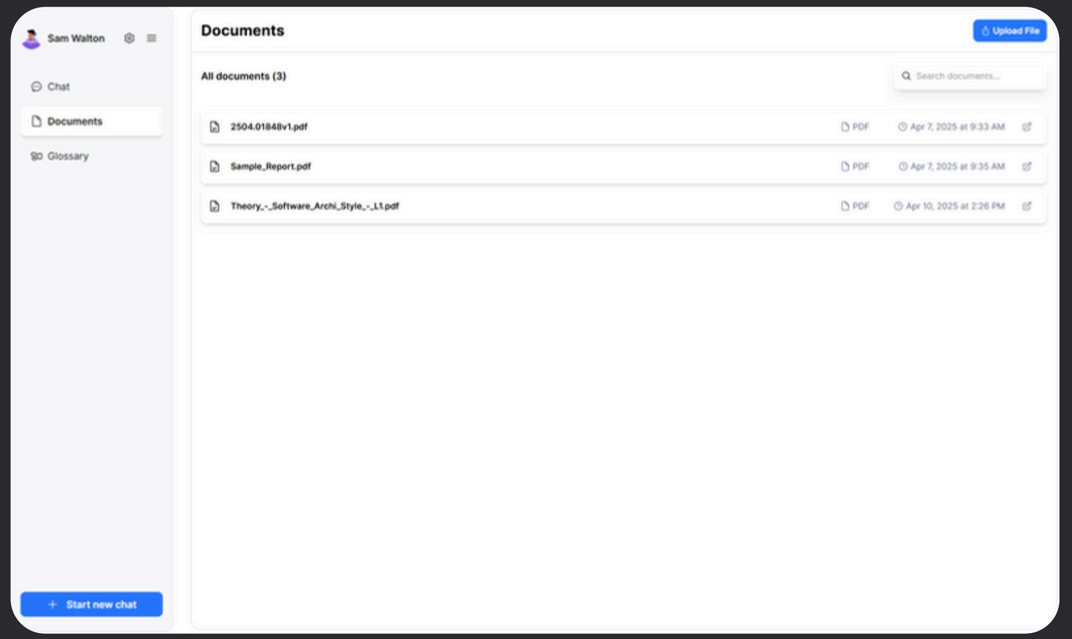
Large language model developed by Meta for various natural language processing tasks.

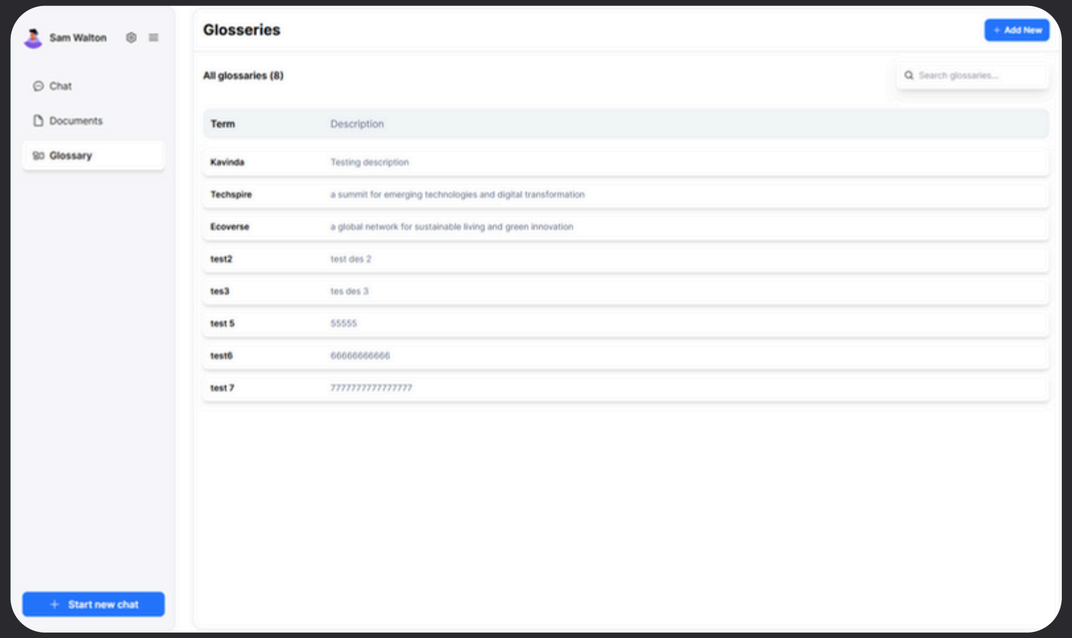
Backend Design↘

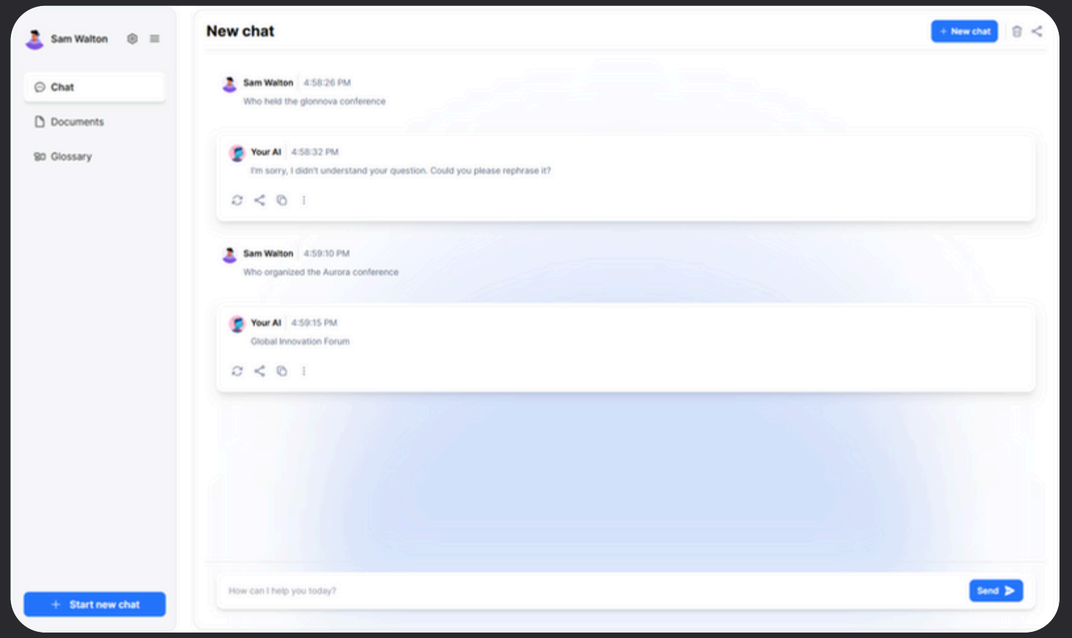


Frontend Design ↘





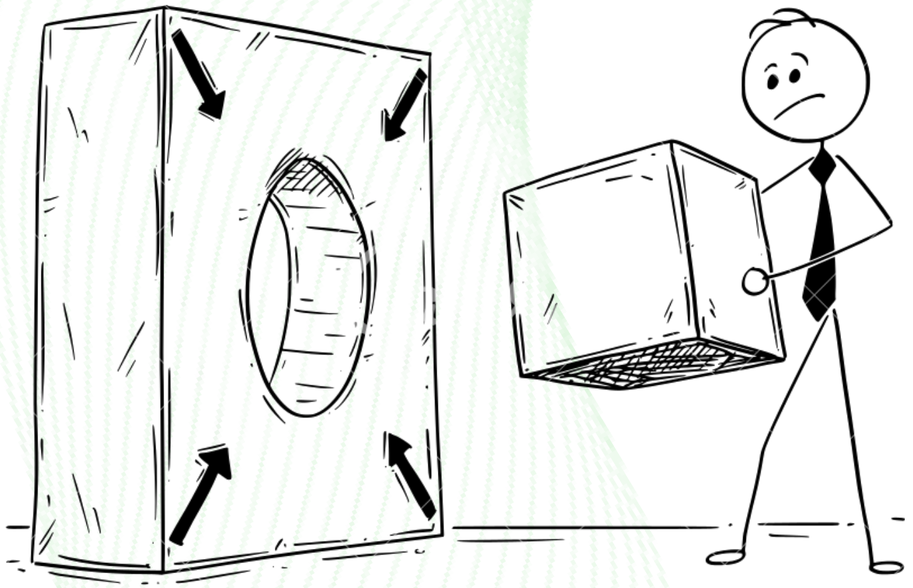




What we have done↘

1. Frontend development - 80% completed
2. Backend development - 75% completed
3. All the required technologies integrated
4. Knowledge Graph created with multiple Docs
5. Glossary integrated

Challenges we faced ↘



1. LLM is not yet effectively utilizing the glossary definitions as expected.
2. Extraction of entities and relationships is time-consuming.
3. Find comphrehensive PDF documents dataset.
4. Extracting document content from complex structures (eg: Tables)

Remaining Things ↘

1. Revert Knowledge graph creation
2. Complete Backend
3. Implement chat history feature
4. Integrate chat history for answer generation
5. LLM response optimization
6. Knowledge Graph optimization
7. Implement OCR functionality
8. Frontend feature enhancements

Have Questions?

Thank You!