Self-Hosted Multi-Agent RAG System For Contextual Document Processing

ENG ZI JUN

UNIVERSITI TUNKU ABDUL RAHMAN

Self-Hosted Multi-Agent RAG System For Contextual Document Processing

Eng Zi Jun

A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Software Engineering with Honours

Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

September 2025

DECLARATION

Name		Eng Zi Jun
ID No.	:	2104132
Date	:	7/10/2025

APPROVAL FOR SUBMISSION

I certify that this project report entitled "**Self-Hosted Multi-Agent RAG System for Contextual Document Processing**" was prepared by Eng Zi Jun has met the required standard for submission in partial fulfilment of the requirements forthe award of Bachelor of Science (Honours) Software Engineering with Honours at Universiti Tunku Abdul Rahman.

Approved by,

Signature	:	[illegible]
Supervisor	:	**Ir Ts Dr Hum Yan Chai**
Date	:	**7/10/2025**

Signature	:	[illegible]
Co-Supervisor	:	**Ng Keng Hoong**
Date	:	**7/10/2025**

COPYRIGHT STATEMENT

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to Associate Professor **Ir Ts Dr Hum Yan Chai** as my research supervisor and Assistant Professor **Dr Ng Keng Hoong** as my research co-supervisor for their invaluable advice, guidance, and enormous patience throughout the development of this project.

I would also like to extend my sincere thanks to the faculty and departmental members from **Lee Kong Chian Faculty of Engineering & Science** and **Department of Computing**, for creating a pleasant and supportive working environment throughout my studies at Universiti Tunku Abdul Rahman.

ABSTRACT

The increasing use of Artificial Intelligence (AI) in document processing faces persistent challenges such as hallucination, privacy risks, and limited adaptability. This study presents a self-hosted multi-agent Retrieval-Augmented Generation (RAG) system designed to address these limitations by enhancing accuracy and preserving data privacy through a fully local and modular architecture. Built using Marker, Ollama, LangGraph, and Weaviate, the system enables flexible deployment and coordination between agents. Evaluation using the SQuAD dataset measured retrieval and generation performance through metrics such as Recall@3, Mean Reciprocal Rank (MRR), Context Recall, Faithfulness, and Answer Correctness. Two evaluation methods were employed: a calculation-based approach on 100 samples for quantitative assessment, and an LLM-as-Judge approach using GPT-4o on 20 samples for qualitative, human-like evaluation. Results show strong retrieval performance with a Recall@3 of 90%, MRR of 75%, and Context Recall of 100%, demonstrating accurate and consistent grounding. The generation results indicate improved faithfulness and contextual relevance, though challenges remain in scalability and factual precision. Overall, the findings show that the proposed multi-agent RAG system effectively mitigates hallucination and privacy concerns while maintaining adaptability, making it a promising approach for secure and accurate AI-driven document processing.

Keywords: Artificial Intelligence (AI), Retrieval-Augmented Generation (RAG), Large Language Models (LLMs), Self-Hosted AI

Subject Area: Q300-390 Cybernetics

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS / ABBREVIATIONS

RAG	*Retrieval-Augmented Generation*
LLM	*Large language model*
MAS	*Multi-agent system*
BM25	*Best Matching 25*
TF-IDF	*Term Frequency and Inverse Document Frequency*
GDRP	*General Data Protection Regulation*
SPDM	*Semantic Double-Pass Merging*
ANN	*Approximate Nearest Neighbour*
MRR	*Mean Reciprocal Rank*
SQuAD	*Stanford Question Answering Dataset*

CHAPTER 1

INTRODUCTION

1.1 General Introduction

“Data really powers everything that we do.” A quote by Jeff Weiner, the CEO of LinkedIn, summarizes the importance of data in the modern data-driven world. Especially with the acceleration of technology advancement in recent years, the demand for and output of data have never been higher. According to Reinsel et al. (2018), the data generated, cloned, or recorded is expected to increase from 33 Zettabytes (ZB) in 2018 to 175 ZB in 2025. However, the number is estimated to be even bigger due to the popularization of AI-generated content and the digitization of industries in recent years. Thus, this situation increases the complexity and time taken for data handling for manual or even some traditional technology approaches.

With the lightning-fast evolution of AI technologies, useful information can be successfully extracted from unstructured data with a minimum amount of human intervention using sophisticated AI-based pipelines with technologies such as Computer Vision, Natural Language Processing, and Deep Learning. Apart from all the technologies mentioned above, Generative AI is also another popular technology utilized in data analysis and synthesis. According to Chauhan (2024), researchers have been evaluating the potential of Generative AI in scientific content analysis and summarization since the first release of ChatGPT back in 2022. Moreover, roughly 5% of scientific documents already contain AI-generated information, indicating that Generative AI has already became a part of the workflow for different document processing tasks (Cheng et al., 2024). However, the approach is still far from perfect due to some major flaws when executing tasks like document processing. Consequently, an RAG system has been introduced as a potential solution to address the challenges mentioned above.

According to Ramalingam (2023), RAG is a technology that utilizes a retrieval system to fetch information from a database and a generative AI model that uses the retrieved information to provide the answer. This mechanism allows RAG to provide answers with better precision and better timeliness due to the ability of the retrieval system to fetch real-time context from the database (Yu et al., 2024). Thus, RAG is a more fitting option for document analysis and generation than a standalone Generative AI model for fields including medicine, law, and finance, which require answer generation with high precision and high context awareness. Despite all the advantages the traditional single-agent RAG system poses, the single-agent RAG system still faces challenges like scalability, flexibility issues and a lack of orchestration features (Gustavo et al., 2025). With the foundation from the traditional RAG system, this project aims to investigate and develop a project titled: Self-Hosted Multi-Agent RAG System for Contextual Document Processing.

1.2 Problem Statement

As the popularity of AI drastically increases, the intelligence of all the latest AI models is also getting smarter with every iteration. People from different backgrounds have already adapted to using AI in their daily lives or even using it to automate part of their work. However, generative AI still encounters some challenges in document analysis and synthesis. Consequently, the RAG system emerges as one of the potential solutions for issues encountered by Generative AI. However, the normal single-agent RAG system still has room for improvement in some respects. Thus, this section will examine the challenges encountered by generative AI and single-agent RAG systems.

1.2.1 Hallucination

The most significant flaw of generative AI is hallucination. Hallucination in AI often refers to the situation where AI generates unrelated, senseless, and incorrect responses to the user-given query, the most common example being citing a random source that does not exist (Duan et al., 2024). Banerjee et al. (2024) stated that misinterpretation of input query and data fabrication due to the lack of supporting context from the model's training data are a few of the reasons that will cause a generative AI model to produce a hallucinated response. Moreover, LLM relies solely on a large amount of static pre-trained data to generate information. Thus, a knowledge gap exists between the time of release of the LLM model and the latest information. Consequently, LLM might try to generate a response with outdated information, increasing the risk of hallucination when a user queries about recent information or trends (Feng et al., 2024). According to Zhang (2023), this phenomenon can potentially evolve into hallucination snowballing, where AI will commit to the first hallucinated answer of the response and continue to justify the incorrect response with fabricated facts, while the correct response can be given in another session. Thus, if hallucination occurs during document processing, it can significantly hurt the overall quality of the responses, causing the inability to meet the user's requirements.

1.2.2 Privacy

Moreover, privacy is also another issue for the generative AI approach. Enterprises' documents often contain large amounts of sensitive information, such as customer information and their business-related information, that AI provider can secretly collect and use for their own operation. According to one of the most popular Generative AI models, ChatGPT's privacy policy, OpenAI will have access to all device information and communication information when a user accesses their services. Thus, this creates a huge privacy concern when a large amount of sensitive information needs to be analyzed. According to Wu et al. (2023), it is apparent that ChatGPT has insufficient measures to protect users' data in compliance with GDRP, which puts user privacy at risk. Furthermore, centralizing massive amounts of user data in the database of AI providers makes these systems attractive targets for cyberattacks, if a hacker successfully breaches into the database, they could exploit the sensitive information for malicious purposes such as identity theft and selling the data for monetary profit, resulting in serious losses for enterprises.

1.2.4 Modularity & flexibility

Although a single-agent RAG system is effective in solving most of the issues mentioned for document analysis with generative AI, the system still faces some challenges that hinder its capability when facing complex use cases. Lack of flexibility and modularity is one of the major downsides of a single-agent RAG system. According to Gustavo et al. (2025), the architecture of a single-agent RAG system can limit the performance and capability of the RAG system due to the generalization of all workflows in the single-agent approach, causing the inability of the system to adapt to different use workflows and intents. This can also result in less accurate responses by the AI generation component if the relevant context is not present in the retrieval component database or the user workflow is different from the system's intended workflow.

1.3 Aim and Objectives

The objectives of this project are:

- To develop a Retrieval-Augmented Generation (RAG) System that can extract relevant information from a document uploaded by the user and selected internet sources ensuring the response is based on grounded information.

- To develop a modular RAG System with an Agentic architecture to ensure the system can dynamically adjust workflow to different user intent and system component settings.

- To implement the system for local deployment, ensuring data privacy by eliminating reliance on external cloud services and enabling complete operation on a standalone machine.

-To integrate with the external document processing MCP server to allow the system to automatically execute document processing operations with LLMs.

1.4 Proposed solution

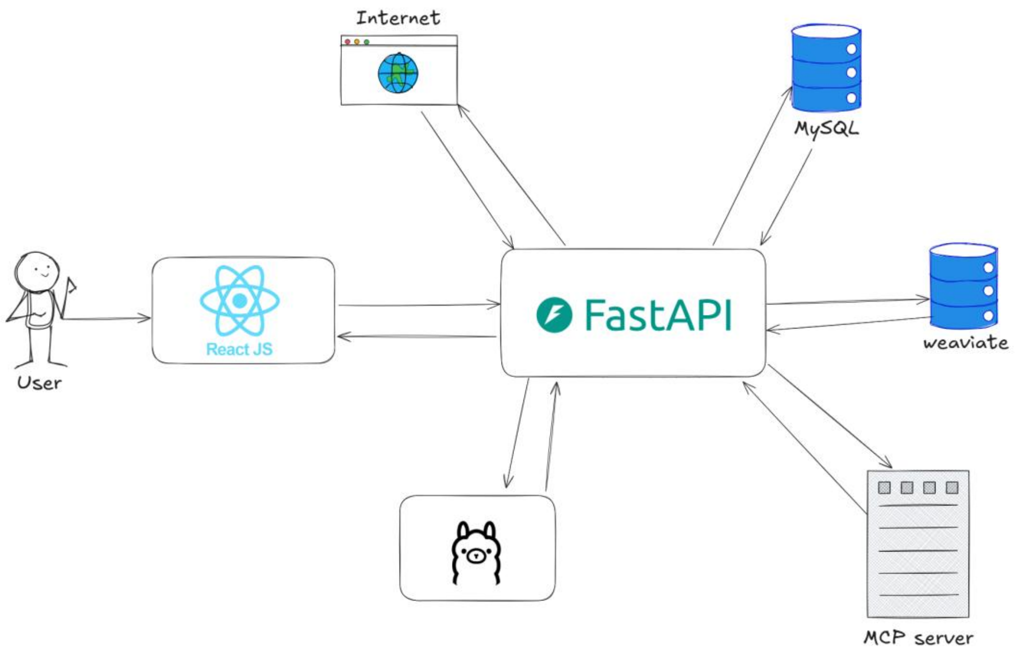


Figure 1.1: Simple outline of architecture

To tackle the problem identified, A local web-based RAG application will be developed following the architecture diagram provided. The users can perform document Q&A and note generation all within a local environment without having to share their data with an external service provider. Other than privacy, the RAG system will be built to tackle the hallucination problem by ensuring all response generations are based on grounded information provided by the user or the internet. Lastly, the system will also follow an agentic design to tackle the modularity and flexibility issue faced by the traditional LLM solution.

Firstly, the user will interact with the system via a React frontend interface that consists of multiple pages, each with their distinct functions and components. For each user interaction, the React frontend will send the HTTP requests to the FastAPI backend to process user requests. The FastAPI backend communicates with a local vector database, Weaviate, to store the text chunks from the document in vector representation and retrieve similar items in vector embeddings, obtaining relevant chunks of documents in response to the user's queries. Additionally, the system backend also interacts with a MySQL database to assist in keyword searches, manage user files, and maintain chat history for active sessions that will be used for other system operations.

For the generation capability, the FastAPI backend will interact with an Ollama that is responsible for hosting LLM locally, which will be used to generate responses without sending data to external services to maintain data privacy. When the query requires response generation, FastAPI will send the query to Ollama’s LLM model, and the response generated will be sent back to the FastAPI backend. When external information is required, the FastAPI backend utilizes a Web Search module, which forwards questions to external search engines to receive up-to-date information and send it back into the system to reduce the hallucination rate of the responses.

Lastly, the FastAPI backend will connect to various MCP servers for different applications via an MCP request. To produce notes, FastAPI calls the relevant tools from the MCP server, which can generate notes in different application formats. This flexible design makes it easy for the system to support note generation for the application to the user's liking, aiming to increase the modularity of the system.

1.5 Scope and Limitations of the Study

The final deliverable of this project is a self-hosted multi-agent system based on traditional RAG technology for document processing that can utilize different agents in the system to perform specialized tasks. The final product of this project will be in the form of a local web application with a front-end interface that allows users to upload their documents to a local database, interact with the system via querying, and change the system's operational settings. Moreover, the system allows users to acquire more accurate information by asking follow-up questions.

This system aims to deliver a modular, secure, and smart document analysis pipeline driven by a multi-agent framework where every agent provides a different range of functionalities within the complete workflow of document intake, orchestrating, processing, retrieval, and generation. The module that will be covered in this project:

1. **Document Upload and Database Module**
   The Document Upload is implemented in this system to provide a way for users to upload their files in PDF format via the user interface. The local database module will handle uploaded document storage for private and efficient access for other modules.

2. **Orchestrator**
   The orchestrator acts as the central controller for the system, which is responsible for dynamically adjusting the workflow based on the user's intent to distribute tasks to the best-suited agent, enhancing the system's flexibility.

3. **Preprocessing Agent**
   The Preprocessing agent in the system will detect and extract the text and table from the stored file. The extracted text and table will then be separated into different chunks to increase the relevancy and efficiency of the retrieval process. After the initial chunking process, a tiny LLM model will be implemented in this agent to refine the content in the chunks to add context to the chunks and remove

unnecessary information, which will produce contextual chunks that will be stored in databases for retrieval needs. Lastly, this agent will also utilize an embedding model to turn the contextual chunks into vector representation for similarity search.

4. **Retrieval Agent**

   The retrieval agent is responsible for retrieving pieces of information that are relevant to the user's query. The retrieval agent will consist of different retrieval methods that are suitable for different retrieval scenarios. This module is also responsible for communication between different web search modules to retrieve relevant information from the internet based on the user's query. This retrieval method can ensure the retrieved information is based on factual information from user-uploaded documents.

5. **Generation Agent**

   The generation agent is responsible for the generation of the final response with the help of a local LLM model to reduce the chance of hallucination. This agent will elaborate on the retrieved information from the obtained retrieval agent and generate a natural language response based on the user’s query.

6. **Document processing Agent**

   The document processing agent is responsible for determining an automated workflow that is suitable for the user's query. The document processing agent will manage communication between different tools from the connected MCP server.

7. **Authentication module**

   A basic login and signup module will be implemented in this system to ensure no outsider can access the system, which can protect the privacy of the user’s information and documents stored in the local database.

However, there are a few limitations to the project. Firstly, the system will allow users to deal only with English-language document processing, and the specialized database search will be limited to legal, financial, and medical domains. Moreover, the synthesis of figures in the document will not be covered. In addition, the precision of generated responses may be influenced by the availability of unique or limited data sets and the limited computing power. System performance and result accuracy might further be influenced by hardware and computational limits. Therefore, real-time processing will not be the main priority of this project. Lastly, the automation agent will only cover auto-note generation.

CHAPTER 2

LITERATURE REVIEW

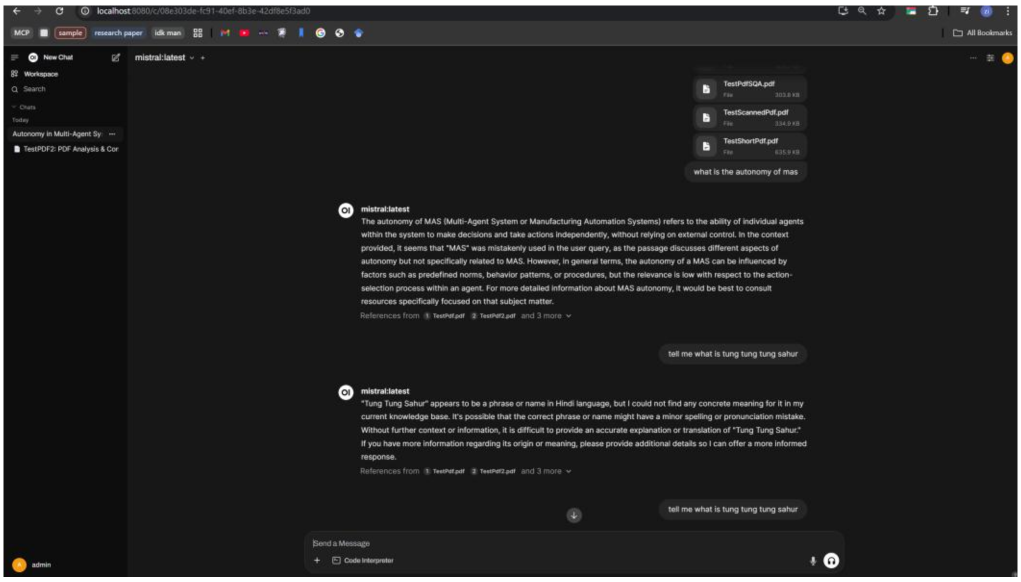
2.1 Introduction

In this chapter, the current architecture design and technologies related to the development of this project are studied to understand more about the current solutions and get ideas on how to enhance them. Based on the analysis of past research work and current applications, the review aims to explore the topics that provide useful concepts that might help in the development of the project.

2.2 Similar existing applications

This section will review some of the popular open-source RAG projects: Open WebUI, Kotaemon, and RAGFlow to identify their system functionalities, strengths, and drawbacks. The comparison will help in the identification of essential design factors and features that could be included in the creation of this project.

2.2.1 Open WebUI



what is the autonomy of mas

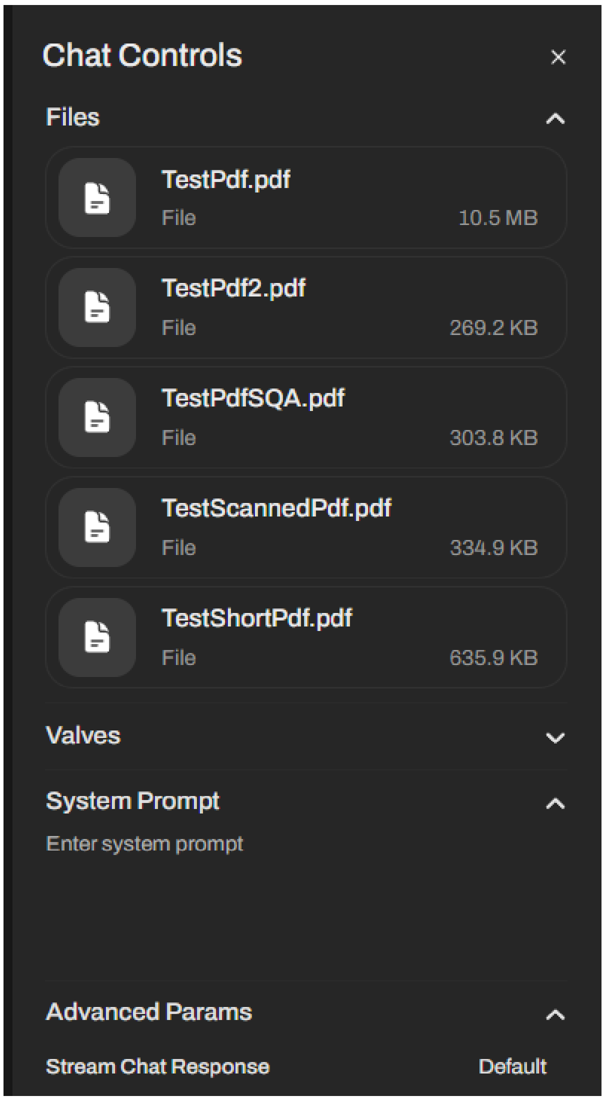
mistral:latest
The autonomy of MAS (Multi-Agent System or Manufacturing Automation Systems) refers to the ability of individual agents within the system to make decisions and take actions independently, without relying on external control. In the context provided, it seems that "MAS" was mistakenly used in the user query, as the passage discusses different aspects of autonomy but not specifically related to MAS. However, in general terms, the autonomy of a MAS can be influenced by factors such as predefined norms, behavior patterns, or procedures, but the relevance is low with respect to the action-selection process within an agent. For more detailed information about MAS autonomy, it would be best to consult resources specifically focused on that subject matter.
References from 1 TestPdf.pdf 2 TestPdf2.pdf and 3 more

tell me what is tung tung tung sahur

mistral:latest
"Tung Tung Sahur" appears to be a phrase or name in Hindi language, but I could not find any concrete meaning for it in my current knowledge base. It's possible that the correct phrase or name might have a minor spelling or pronunciation mistake. Without further context or information, it is difficult to provide an accurate explanation or translation of "Tung Tung Sahur." If you have more information regarding its origin or meaning, please provide additional details so I can offer a more informed response.
References from 1 TestPdf.pdf 2 TestPdf2.pdf and 3 more

tell me what is tung tung tung sahur

Figure 2.1: Open WebUI main chat interface



Chat Controls

Files

TestPdf.pdf — File — 10.5 MB

TestPdf2.pdf — File — 269.2 KB

TestPdfSQA.pdf — File — 303.8 KB

TestScannedPdf.pdf — File — 334.9 KB

TestShortPdf.pdf — File — 635.9 KB

Valves

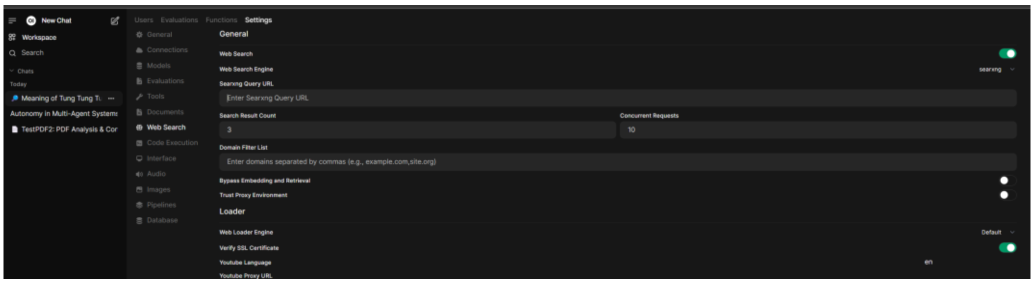
System Prompt

Enter system prompt

Advanced Params

Stream Chat Response: Default

Figure 2.2: Open WebUI file management for one chat session



General

Web Search: ●

Web Search Engine: searxng

Searxng Query URL: Enter Searxng Query URL

Search Result Count: 3

Concurrent Requests: 10

Domain Filter List: Enter domains separated by commas (e.g., example.com,site.org)

Bypass Embedding and Retrieval

Trust Proxy Environment

Loader

Web Loader Engine: Default

Verify SSL Certificate: ●

Youtube Language: en

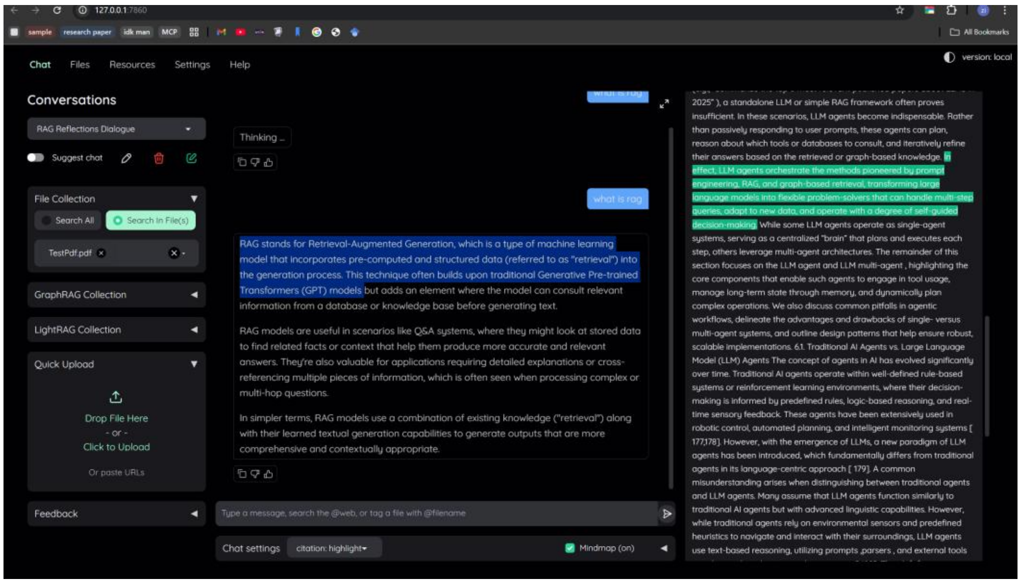
Youtube Proxy URL

Figure 2.3: Open WebUI setting interface

Open WebUI is a popular open-source RAG tool that offers a simple, intuitive platform that enables users to perform basic RAG tasks in an entirely offline environment. Open WebUI allows users to fetch relevant information from documents with different file formats, such as PDF, markdown, and LaTeX, making the system more flexible for different user requirements. Additionally, Open WebUI also provides web search functionality, which allows the system to fetch up-to-date information from the internet, enhancing the system's ability to provide a grounded response to the user. Open WebUI also provides user access control that ensures the security of the system. Another notable feature of Open WebUI is its complete compatibility with Ollama models, allowing users to change models or tailor components effortlessly, making it a flexible option for individuals seeking to operate local LLMs in a custom configuration.

Because of the focus on ease of use and simplicity, it lacks extra functionality that can potentially be helpful to the user. For example, Open WebUI uses a chat session-based file retrieval method, meaning users need to reimport the same file if they want to start a new chat session. Open WebUI also does not support multi-agent workflow, which can limit the capability of the system.

2.2.2 Kotaemon



what is rag

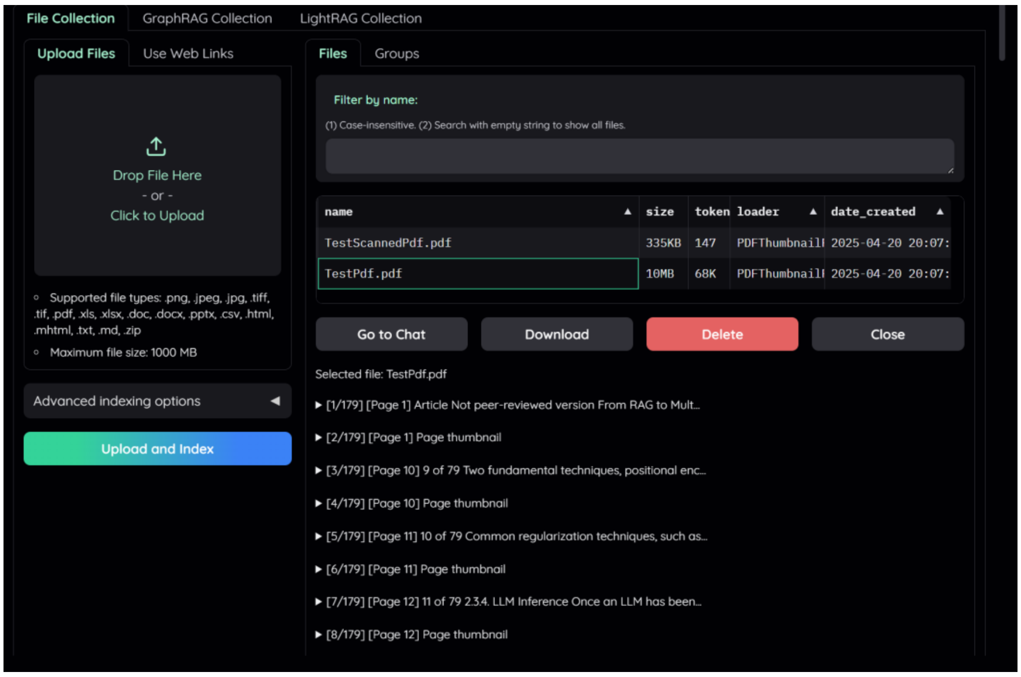
RAG stands for Retrieval-Augmented Generation, which is a type of machine learning model that incorporates pre-computed and structured data (referred to as "retrieval") into the generation process. This technique often builds upon traditional Generative Pre-trained Transformers (GPT) models but adds an element where the model can consult relevant information from a database or knowledge base before generating text.

RAG models are useful in scenarios like Q&A systems, where they might look at stored data to find related facts or context that help them produce more accurate and relevant answers. They're also valuable for applications requiring detailed explanations or cross-referencing multiple pieces of information, which is often seen when processing complex or multi-hop questions.

In simpler terms, RAG models use a combination of existing knowledge ("retrieval") along with their learned textual generation capabilities to generate outputs that are more comprehensive and contextually appropriate.

2025"), a standalone LLM or simple RAG framework often proves insufficient. In these scenarios, LLM agents become indispensable. Rather than passively responding to user prompts, these agents can plan, reason about which tools or databases to consult, and iteratively refine their answers based on the retrieved or graph-based knowledge. In effect, LLM agents orchestrate the methods pioneered by prompt engineering, RAG, and graph-based retrieval, transforming large language models into flexible problem-solvers that can handle multi-step queries, adapt to new data, and operate with a degree of self-guided decision-making. While some LLM agents operate as single-agent systems, serving as a centralized "brain" that plans and executes each step, others leverage multi-agent architectures. The remainder of this section focuses on the LLM agent and LLM multi-agent , highlighting the core components that enable such agents to engage in tool usage, manage long-term state through memory, and dynamically plan complex operations. We also discuss common pitfalls in agentic workflows, delineate the advantages and drawbacks of single- versus multi-agent systems, and outline design patterns that help ensure robust, scalable implementations. 6.1. Traditional AI Agents vs. Large Language Model (LLM) Agents The concept of agents in AI has evolved significantly over time. Traditional AI agents operate within well-defined rule-based systems or reinforcement learning environments, where their decision-making is informed by predefined rules, logic-based reasoning, and real-time sensory feedback. These agents have been extensively used in robotic control, automated planning, and intelligent monitoring systems [177,178]. However, with the emergence of LLMs, a new paradigm of LLM agents has been introduced, which fundamentally differs from traditional agents in its language-centric approach [179]. A common misunderstanding arises when distinguishing between traditional agents and LLM agents. Many assume that LLM agents function similarly to traditional AI agents but with advanced linguistic capabilities. However, while traditional agents rely on environmental sensors and predefined heuristics to navigate and interact with their surroundings, LLM agents use text-based reasoning, utilizing prompts ,parsers , and external tools

Figure 2.4: Kotaemon main chat interface



- Supported file types: .png, .jpeg, .jpg, .tiff, .tif, .pdf, .xls, .xlsx, .doc, .docx, .pptx, .csv, .html, .mhtml, .txt, .md, .zip
- Maximum file size: 1000 MB

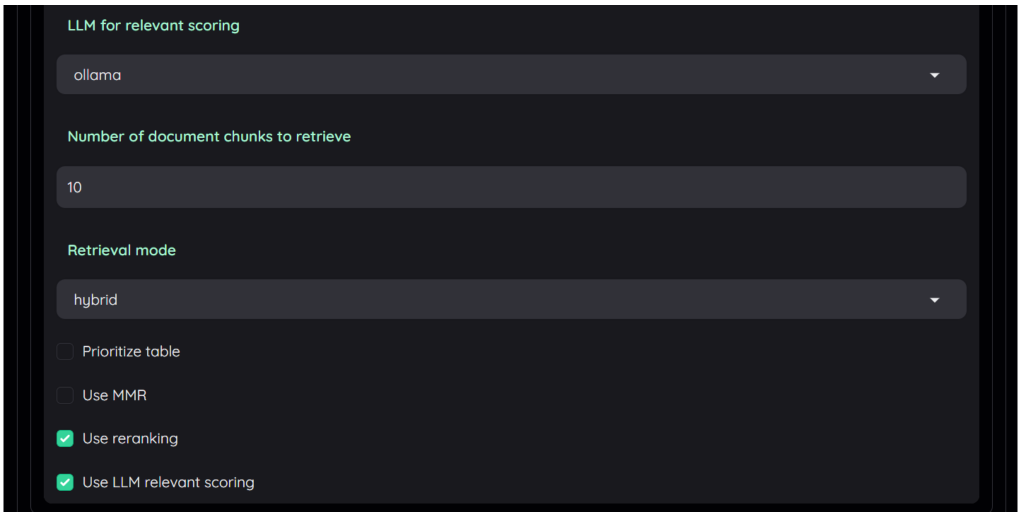
Filter by name:
(1) Case-insensitive. (2) Search with empty string to show all files.

name	size	token	loader	date_created
TestScannedPdf.pdf	335KB	147	PDFThumbnail	2025-04-20 20:07:
TestPdf.pdf	10MB	68K	PDFThumbnail	2025-04-20 20:07:

Selected file: TestPdf.pdf

▶ [1/179] [Page 1] Article Not peer-reviewed version From RAG to Mult...
▶ [2/179] [Page 1] Page thumbnail
▶ [3/179] [Page 10] 9 of 79 Two fundamental techniques, positional enc...
▶ [4/179] [Page 10] Page thumbnail
▶ [5/179] [Page 11] 10 of 79 Common regularization techniques, such as...
▶ [6/179] [Page 11] Page thumbnail
▶ [7/179] [Page 12] 11 of 79 2.3.4. LLM Inference Once an LLM has been...
▶ [8/179] [Page 12] Page thumbnail

Figure 2.5: Kotaemon file management interface


LLM for relevant scoring: ollama

Number of document chunks to retrieve: 10

Retrieval mode: hybrid

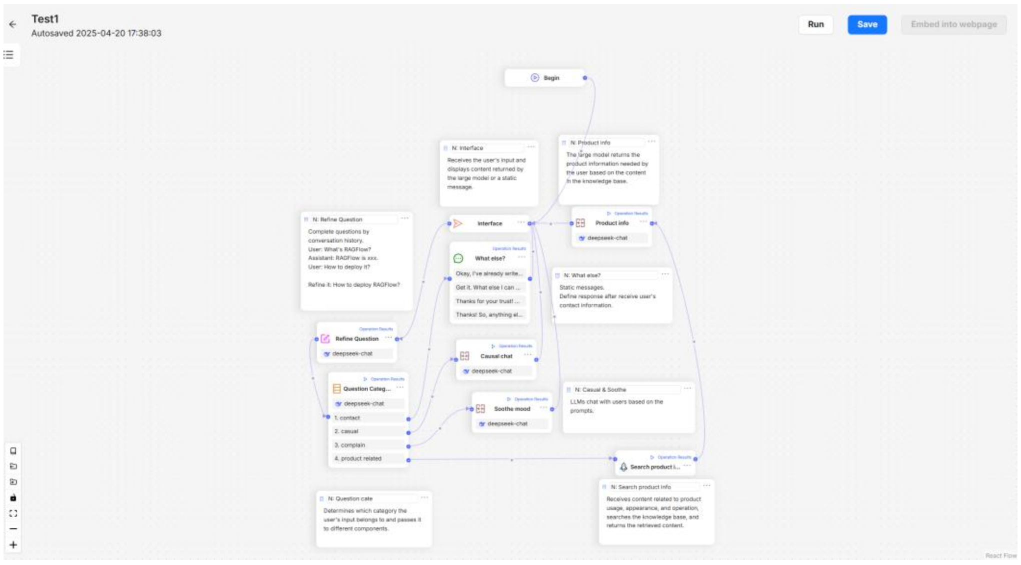
- [ ] Prioritize table
- [ ] Use MMR
- [✓] Use reranking
- [✓] Use LLM relevant scoring

Figure 2.6: Kotaemon setting interface

Kotaemon is another popular open-source RAG application that focuses on providing advanced document retrieval with precise document citation, suitable for experienced users or researchers who require clarity and precision in RAG processes, while still requiring an easy-to-use system. A notable feature of Kotaemon is its hybrid retrieval capabilities, which use both dense and sparse retrieval methods to increase the effectiveness of the retrieval. Additionally, Kotaemon is also capable of providing document preview with precise citations to the related section, which allows users to visually confirm the relevant information that is being used by the LLM to generate its response, ensuring the generation is based on grounded facts, increasing the reliability of the system. Kotaemon manages user-uploaded files in a centralized manner, maintaining a consistent file reference across different chat sessions. Like Open WebUI, this application also features web search functionality, enabling it to retrieve up-to-date information. Kotaemon also implements user access control to prevent unauthorized access in shared environments. Kotaemon is fully compatible with Ollama, which allows users to customize the LLM and embedding model to their liking.

Despite the advantage, Kotaemon also does not support multi-agent workflow, which limits its flexibility for different types of users' workflows, limiting its capability to become a more robust RAG system.

2.2.3 RAGFlow



N: Interface
Receives the user's input and displays content returned by the large model or a static message.

N: Product info
The large model returns the product information needed by the user based on the content in the knowledge base.

N: Refine Question
Complete questions by conversation history.
User: What's RAGFlow?
Assistant: RAGFlow is xxx.
User: How to deploy it?

Refine it: How to deploy RAGFlow?

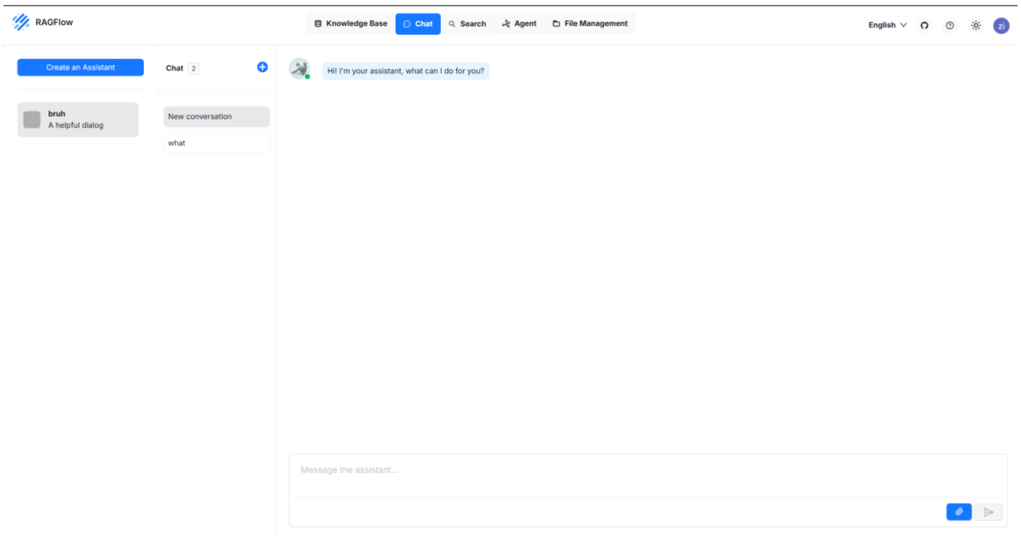
N: What else?
Static messages.
Define response after receive user's contact information.

N: Casual & Soothe
LLMs chat with users based on the prompts.

N: Question cate
Determines which category the user's input belongs to and passes it to different components.

N: Search product info
Receives content related to product usage, appearance, and operation, searches the knowledge base, and returns the retrieved content.

Figure 2.7: RAGFlow Agent workflow management

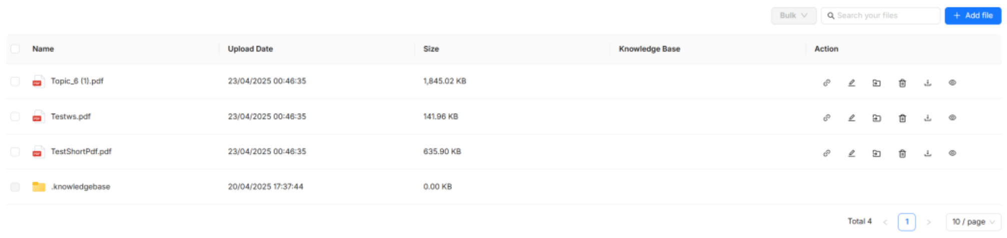


bruh
A helpful dialog

New conversation
what

Hi! I'm your assistant, what can I do for you?

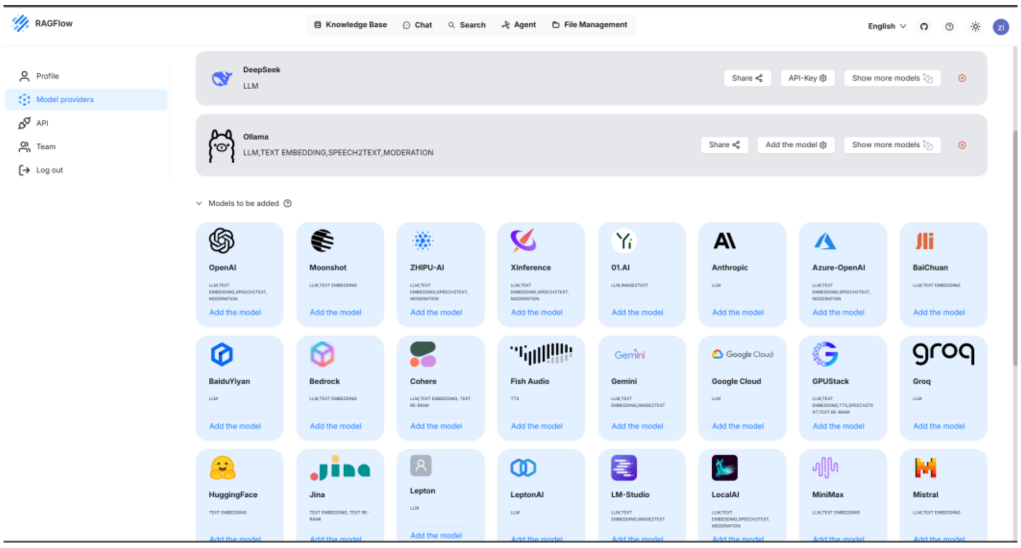
Figure 2.8: RAGFlow chat interface



Name	Upload Date	Size	Knowledge Base	Action
Topic_6 (1).pdf	23/04/2025 00:46:35	1,845.02 KB		
Testws.pdf	23/04/2025 00:46:35	141.96 KB		
TestShortPdf.pdf	23/04/2025 00:46:35	635.90 KB		
.knowledgebase	20/04/2025 17:37:44	0.00 KB		

Total 4 1 10 / page

Figure 2.9: RAGFlow knowledge base



DeepSeek
LLM

Ollama
LLM,TEXT EMBEDDING,SPEECH2TEXT,MODERATION

OpenAI	LLM,TEXT EMBEDDING,SPEECH2TEXT, MODERATION	Add the model
Moonshot	LLM,TEXT EMBEDDING	Add the model
ZHIPU-AI	LLM,TEXT EMBEDDING,SPEECH2TEXT, MODERATION	Add the model
Xinference	LLM,TEXT EMBEDDING,SPEECH2TEXT, MODERATION	Add the model
01.AI	LLM,IMAGE2TEXT	Add the model
Anthropic	LLM	Add the model
Azure-OpenAI	LLM,TEXT EMBEDDING,SPEECH2TEXT, MODERATION	Add the model
BaiChuan	LLM,TEXT EMBEDDING	Add the model
BaiduYiyan	LLM	Add the model
Bedrock	LLM,TEXT EMBEDDING	Add the model
Cohere	LLM,TEXT EMBEDDING, TEXT RE-RANK	Add the model
Fish Audio	TTS	Add the model
Gemini	LLM,TEXT EMBEDDING,IMAGE2TEXT	Add the model
Google Cloud	LLM	Add the model
GPUStack	LLM,TEXT EMBEDDING,TTS,SPEECH2TEXT,TEXT RE-RANK	Add the model
Groq	LLM	Add the model
HuggingFace	TEXT EMBEDDING	Add the model
Jina	TEXT EMBEDDING, TEXT RE-RANK	Add the model
Lepton	LLM	Add the model
LeptonAI	LLM	Add the model
LM-Studio	LLM,TEXT EMBEDDING,IMAGE2TEXT	Add the model
LocalAI	LLM,TEXT EMBEDDING,SPEECH2TEXT, MODERATION	Add the model
MiniMax	LLM,TEXT EMBEDDING	Add the model
Mistral	LLM,TEXT EMBEDDING	Add the model

Figure 2.10: RAGFlow model setting

RAGFlow is a flexible and robust open-source RAG application that allows users to create highly customized system pipelines with different components, allowing users to alter the pipeline of RAGFlow to make it more suitable for their use case. Unlike Kotaemon and Open WebUI, RAGFlow is one of the few open-source RAG applications that support a multi-agent architecture, which allows different components or agents to perform distinct functions for collaborative processing. RAGFlow also provides advanced web search capabilities that allow users to perform web searches on various sources such as Baidu, Google, Google Scholar, and Yahoo Finance. This application also implements user-access control as the first line of defence for unauthorized access to the system. RAGFlow is also entirely compatible with the Ollama platform, enabling users to change the model to their liking. Similar to Kotaemon, RAGFlow maintains a centralized knowledge base that allows continuous access to files throughout sessions.

Although RAGFlow is the most robust and adaptable RAG application compared, RAGFlow has the steepest learning curve required to operate the application effectively. Thus, it is more suitable for experienced users that extensive customization, agent orchestration, and system integration.

2.2.4 Comparison table for similar applications

Table 2.1: Comparison table for similar applications

Feature	**RAGFlow**	**Kotaemon**	**Open WebUI**
Key feature	Advance the RAG pipeline with a variety of tools to support	Advanced document interaction	Simple document interaction
Primary Focus	Enterprise-grade customizable RAG workflow	Advanced retrieval with a hybrid retrieval function and detailed citation	Easy-to-use document Q&A
Multi-agent support	✓	✗	✗
Web search	✓	✓	✓
User Access Control	✓	✓	✓
Compatible with Ollama model	✓	✓	✓
Ease of Use	Hard	Medium	Easy
File retrieval	Centralized file retrieval	Centralized file retrieval	Session-based file retrieval

In conclusion, the three application reviews in this section identify the varying functionalities and drawbacks offered by each tool in their RAG system that can be used as ideas for the development of this project. Leveraging the advantages of each platform, this project will include these essential features:

1. Multi-agent workflow to facilitate flexible workflow adjustment to specialized functions and a modular pipeline.

2. Collaboration with external document generation tools
   for smooth subsequent use.
3. Hybrid document retrieval techniques for improved response precision.
   Online search functionalities to enhance internal knowledge
   with current external data.
4. User verification and access management to facilitate secure, multi-user implementation.
5. A clear interface and minimal setup complexity for ease of use.
6. Compatibility with local models through Ollama, guaranteeing privacy and adaptability in choosing models.
7. Centralized document retrieval to ensure file continuity throughout different sessions.

2.3 Multi-agent system (MAS) design pattern

As the complexity of problems addressed by AI systems increases, MAS has developed into a strong approach to address these advancing challenges. Including several specialized agents that interact, cooperate, and compete, working towards achieving the system's objective. Therefore, this section will explore different design patterns commonly used in multi-agent systems to enhance understanding of these patterns and analyse the advantages and disadvantages of each pattern to decide on which pattern is more suitable to be implemented in this system.

2.3.1 Multi-agent collaboration pattern

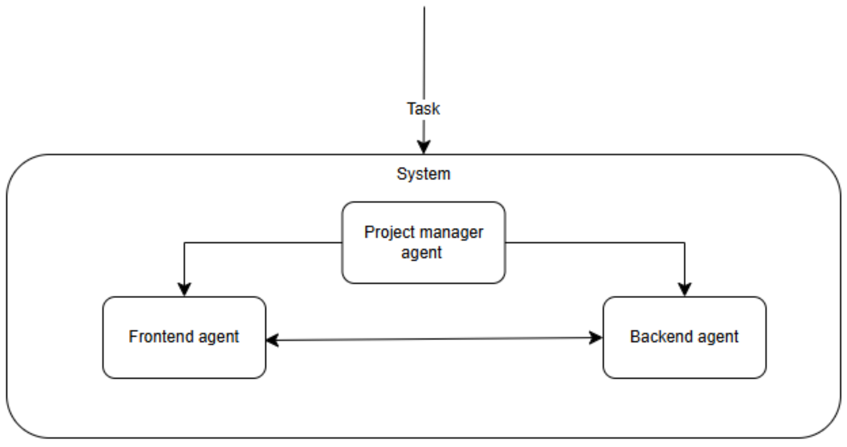


Figure 2.11: Multi-agent collaboration design pattern

Multi-agent collaboration pattern is a pattern that consists of specialized agents collaborating towards a shared objective by synchronizing their specific tasks. (Gustavo et al., 2025). Every agent in this pattern is created to execute a unique function or handle a problem area, such as data retrieval, classification, reasoning, or generation, instead of depending on one centralized entity to control and execute the entire workflow. This allows the system to be more robust and scalable by assigning specific tasks to each of the agents. For example, a specialized summarizer agent can be implemented in this system whose sole purpose is to perform summarization of documents, and which tools used in this agent can be fine-tuned to generate the best possible result. Moreover, this pattern can also allow parallel execution of tasks by distributing different segments of the task to different agents, which can improve the processing time of the system.

However, this design pattern requires a significant amount of computing power to perform, especially with agents that use LLM to perform their tasks. The scalability and performance of the agent will be greatly impacted when the computing power is limited (Guo et al., 2024). According to Singh et al. (2025), another challenge the multiagent collaboration pattern faces is the complex coordination between multiple agents, which requires the system to implement a robust orchestrator to manage the information flow between every agent to ensure seamless workflow between agents. If the orchestrator is not effective, problems such as agent information conflict and redundant processing could happen, which reduces the system's performance.

2.3.2 Hierarchical pattern

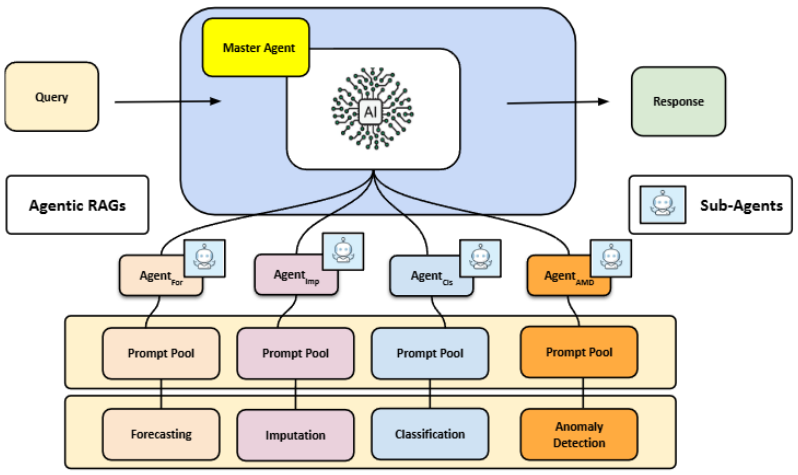


Figure 2.12: Hierarchical multi-agent design pattern (Ravuru et al., 2024)

Hierarchical pattern is a multi-agent design pattern that utilizes a structure and multi-level pattern, where the higher-level agent manages and guides subordinate agents to improve the effectiveness and strategic choices of the system (Singh et al., 2025). This pattern reflects actual hierarchical systems, like organizational management frameworks, in which senior decision-makers delegate duties to different departments according to their expertise, improving the decision-making, load balance, and modularity of the system by guaranteeing that tasks are allocated to the most suitable agents to be processed.

The higher-level agent in this pattern will function as the coordinator of the system, which is responsible for decision making and task assignment to the most suitable subagent (Li et al., 2024 The lower-level agents will focus on performing tasks, executing the operations designated to them, which can ensure the accuracy of the response. Moreover, the modular design of the sub-agent allows it to be modified independently, which increases the system's flexibility (Ravuru et al., 2024).

However, this pattern also faces some difficulties. Like the Multi-Agent Collaboration Pattern, it depends significantly on a capable orchestrator, since communication among sub-agents may raise orchestration overhead (Singh et al., 2025). Additionally, this pattern is also vulnerable to a single-point failure where the failure of the higher-level agent can affect the execution of its subagents, which can potentially result in system-wide breakdowns.

2.3.3 Decentralized pattern

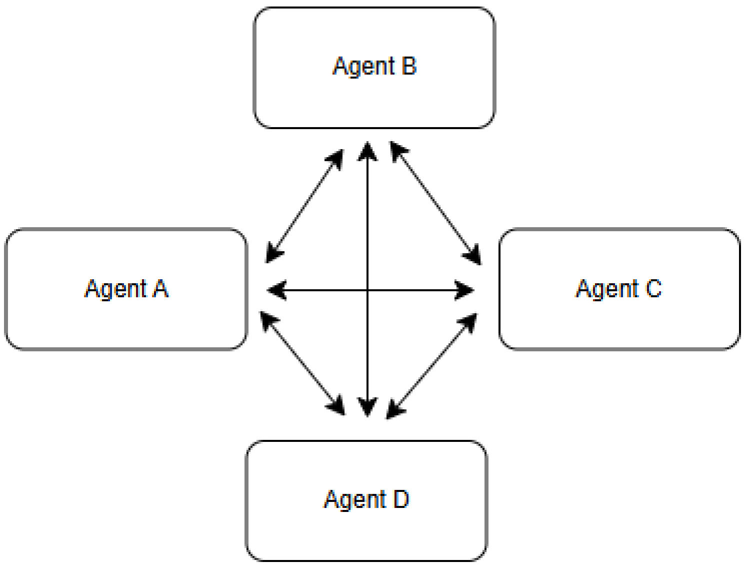


Figure 2.13: Decentralized multi-agent design pattern

A decentralized pattern is the pattern that removes the central control node in a multi-agent system, where the agents will link in a peer-to-peer basis and all agents can directly communicate with each other (Tran et al., 2025). In this pattern, every agent will function independently and can start communication and coordinate with any agent in the network. Thus, it eliminates the need for a central coordinator to perform decision-making and task distribution.

According to Jiménez et al. (2018), this pattern can increase the reliability of the system because if one agent node fails, the agent can still communicate with another replacement. In addition, this pattern can increase the scalability of the number of agents, where a new agent can be added to the decentralized network without affecting the overall workflow of the system. A decentralized pattern distributes the decision-making process across the system. While each agent has their responsibilities, the system can dynamically adjust the communication between agents, which can balance the load distributed to each agent.

Even with its strength, the decentralization pattern also presents some issues. In the absence of a central controller, the system necessitates a sophisticated coordination protocol to effectively synchronize decision-making and avoid inconsistencies in agent communication. As the number of agents grows, these challenges become clearer, which makes an effective communication protocol necessary to ensure consistent performance in a larger system.

2.3.4 Reflection pattern

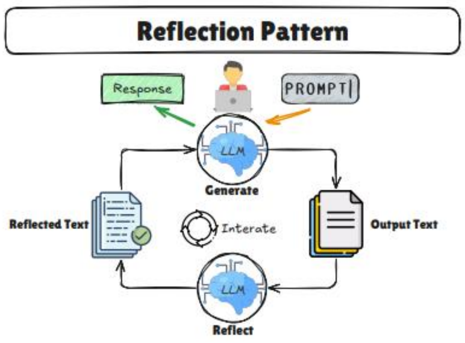


Figure 2.14: Reflection multi-agent design pattern (Gustavo et al., 2025)

The reflection pattern is the design pattern that focuses on enabling an LLM agent to critically examine and iteratively refine its own outputs by re-evaluating the initial output to perform error correction (Gustavo et al., 2025). This pattern resembles a human-like method of self-evaluation, in which an initial response is evaluated critically, and a polished version of the output is generated with the feedback from the evaluation.

This characteristic allows the system to generate a more accurate response by learning from earlier errors and correcting it, making this an ideal pattern to implement in a system where factual accuracy is crucial. The reflection pattern also increases the explainability of the output, where the reflection agent can explain the review and reasoning process for the iteration from the last output (Liu et al., 2024).

As for the downside, the reflection pattern strongly relies on the reflection agent’s reflective ability to maintain the consistency of the output or else it might lead to a lazy reflector where the result is similar or worse than the original result (Bo et al., 2024). Moreover, employing an LLM-based reflection agent requires the system to have extra computational power to handle various rounds of reflection, possibly affecting performance and response times

2.3.5 Comparison between different MAS design patterns

Table 2.2: Comparison between different MAS design patterns

Pattern	**Description**	**Strengths**	**Limitations**
Multi-Agent Collaboration Pattern	Specialized agents collaborate by synchronizing tasks towards a shared goal.	- Improves system domain knowledge and scalability. - Allows parallel execution of tasks.	- High computational power requirement. - Complex coordination needed to prevent redundancy and conflicts.
Hierarchical Pattern	Uses a structured multi-level approach where higher-level agents manage subagents.	- Improves decision-making, load balancing, and modularity. - Sub-agent can be modified independently to increase system flexibility	- High dependence on a capable orchestrator. - Vulnerable to single-point failures.
Decentralized Pattern	Removes the central control node; agents communicate directly in a peer-to-peer network.	- Increases system reliability and scalability. - Distributes decision-making across agents for better load-balancing.	- Requires sophisticated coordination protocols. - Risk of inconsistencies in agent communication.

Reflection Pattern	Implements a reflection agent that refines responses by re-evaluating initial output for error correction.	-Improves factual accuracy of the final output. -Increases explainability for the reasoning of outputs.	-Performance depends on the reflection agent's effectiveness. -Requires extra computational power.

2.4 Type of RAG model

With different MAS design patterns, RAG models have become very flexible in adapting to different use cases. Thus, Various RAG models can be created to enhance retrieval efficiency, improve response accuracy, and adjust to changing queries. Different models utilize different multi-agent design pattern, which impacts their effectiveness, scalability, and dependability. This section explores various RAG models, such as Naïve RAG, Agentic RAG, Corrective RAG, and Hierarchical RAG, to assess their structural strengths and weaknesses and how they work.

2.4.1 Naïve RAG

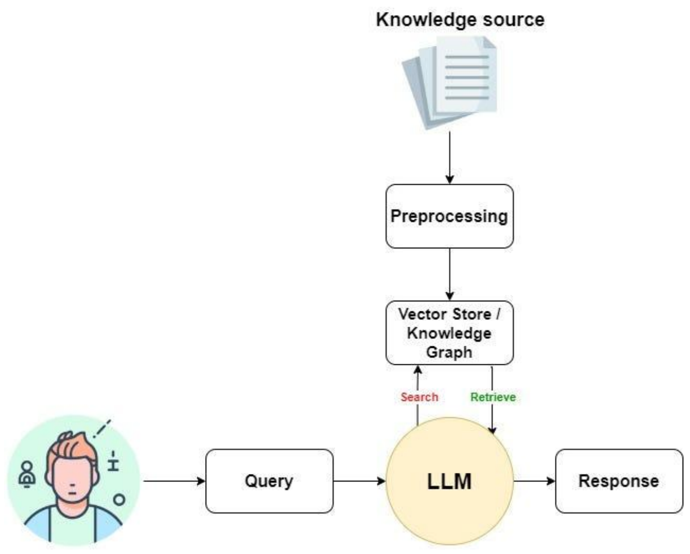


Figure 2.15: Naïve RAG flowchart (Homayoun S., 2025)

Naïve RAG is the simplest form of the RAG model, employing a straightforward, simple pipeline that obtains documents as a direct input into a generative AI to generate responses. Most of the Naïve RAG model only utilizes a single agent method that follows a basic retrieval-to-generation pipeline without extra process or enhancement, where the model will execute sequentially from fetching relevant information from the database based on the user’s query and generating a response based on the relevant information.

This model uses three major components in its core: the retriever, the generator, and the storage database. The retriever is the most important component in the Naïve RAG model because the output quality is directly related to the quality of the retrieved information (Cuconasu et al., 2024). The retriever is responsible for retrieving the most relevant information from the knowledge pile based on the user's query with different retrieval algorithms, such as BM25 and TF-IDF, to rank the relevance of the information. The database acts as the knowledge base, containing documents and information chunks, which are normally executed using a vector database such as ChromaDB and Weaviate. The Generator will be responsible for the last step of the process, which is to generate insight based on the relevant information fetched with the LLM model, such as qwen2.5 or Llama3.

A key strength of the Naïve RAG model is its simplicity, as it can be easily implemented and is less demanding in computational power, making it a good option for systems that value efficiency. Moreover, the model's sequential processing allows for quick response times, making it ideal for real-time use cases. However, this model has some noteworthy drawbacks. It has no corrective mechanisms, which means the produced responses rely solely on the capabilities of the retriever. If the retriever acquires unrelated information chunks, the generator cannot validate or improve its results, resulting in possible inaccuracy. Additionally, the lack of multi-agent cooperation limits the model's capacity to address complex queries since it has no capability to process specialized tasks to specialized agents.

Nevertheless, the Naïve RAG model provides a solid foundation for more sophisticated RAG architectures. Though it works well for fundamental uses like FAQ chatbots, document searches, and basic enterprise knowledge management, advanced models like Agentic RAG, Corrective RAG, and Knowledge Graph RAG have emerged to meet the growing need for accuracy, reasoning ability in practical applications

which introduced a model that

2.4.2 Agentic RAG

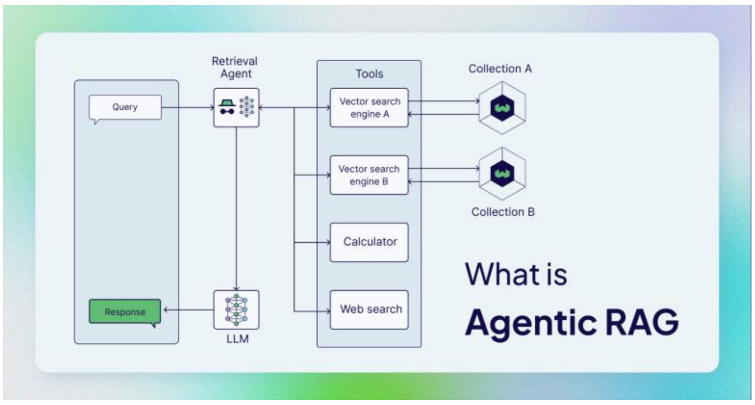


Figure 2.16: Weaviate Agentic RAG flowchart
(Cardenas and Monigatti, 2024)

The agentic RAG model is an updated version of the naïve RAG pipeline by integrating different autonomous and specialized agents to supervise various process phases. But unlike Naïve RAG, which functions in a linear process, Agentic RAG's specialized agent will allocate different tasks to specific agents based on the nature of the task, which introduces decision-making capabilities to the system that allow agents to work together to optimize retrieval, boost response generation, and increase the overall adaptability of the system

The most significant difference between Agentic RAG and Naïve RAG is the multi-agent architecture, where Agentic RAG consists of an orchestrator agent to coordinate the communication between multiple agents specialized in tasks such as query understanding, fact-checking, or agents with domain-specific knowledge, guaranteeing that the obtained information is precise and contextually accurate.

Agentic RAG normally utilizes the Multi-Agent Collaboration Pattern found in the MAS design pattern. In this design pattern, specialized agents collaborate and work towards a common objective, aligning their efforts to improve efficiency. This design enables the system to parallelize tasks like retrieval and validation, enhancing overall efficiency when managing intricate queries or extensive document collections

The strength of Agentic RAG is that it can enhance the precision, flexibility, and scalability of the system. By distributing tasks to independent agents, the model can improve its retrieval and response methods, reducing the chance of generating misinformation. Moreover, an agent-based design allows the developer to tweak the agent's function to better suit the specialized task to make the system perform better. Nonetheless, these advantages increase the computational requirements, as handling several agents necessitates effective coordination and resource management (Singh et al., 2025).

2.4.3 Corrective RAG

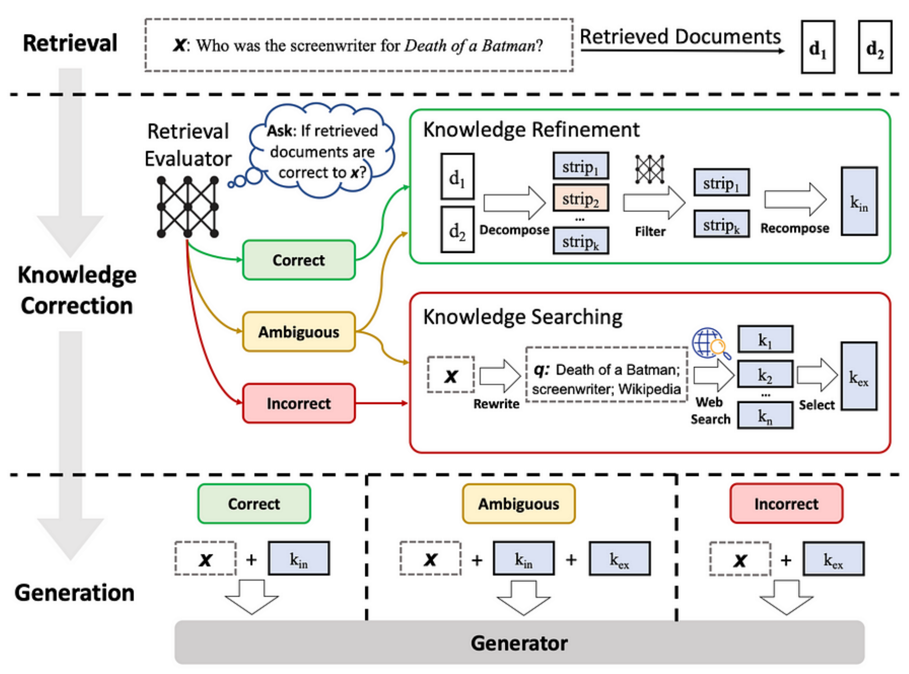


Figure 2.17: Corrective RAG flowchart (Yan et al., 2024)

The corrective RAG model is another extension of the traditional RAG model, which incorporates an additional corrective mechanism to revise the response from the retrieval module or the final Generative AI. Thus, the corrective loop greatly reduces the potential of hallucinations and retrieval inaccuracy of the model, making Corrective Rag a great option for tasks that need high accuracy and minimal margin of error.

Corrective RAG's automated feedback system is the main difference separating it from Naïve RAG. Rather than producing a response in one cycle, Corrective RAG closely adheres to the Reflection Pattern in the MAS design pattern. In this model, a reflection module constantly evaluates and enhances outputs from the generator, confirming that earlier errors are rectified in subsequent versions. This guarantees that the result is both accurate and factually consistent with the retrieved documents. Additionally, the feedback system can also be implemented at the retrieval layer. According to Yan et al. (2024), a corrective module can also be implemented at the retrieval layer of the system which can evaluate the retrieved information and execute corrective action if the information is unrelated to the query, this can greatly reduce the potential of inaccuracy because the retrieval module is the most important part of the RAG system

However, the downside of Corrective RAG is that it demands extra computational resources to handle various cycles of retrieval and generation, which may lead to longer response times and increased resource consumption. Furthermore, excessive correction may arise if the validation process is too intense, resulting in unwarranted changes that fail to enhance the response and instead sabotage the final accuracy.

2.4.4 Knowledge Graph RAG

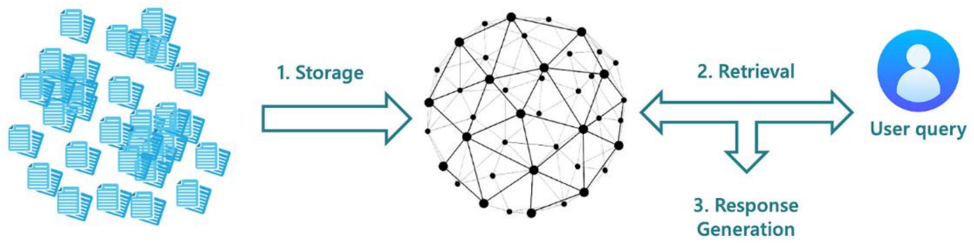


Figure 2.18: Knowledge Graph RAG flowchart (Sanmartin, 2024)

The Knowledge Graph RAG is a special type of RAG model that integrates a Knowledge Graph to perform retrieval and reasoning that replaces the retriever in the Naïve RAG model. Li, Miao, et al. (2024) stated that, unlike the retrieval method used in Naïve RAGs such as BM25 and TF-IDF that uses similarity or keyword for searching, Knowledge Graph provides the ability to reason that fills the gap for tasks that require complex reasoning to solve. Thus, Knowledge Graph RAG is effective in handling complex queries and data relationships that require deep reasoning. Moreover, the Knowledge graph can combine different sources of both structured and unstructured data, offering a more comprehensive knowledge base compared to a vector database (Peng et al., 2023). According to Sanmartin (2024), Knowledge Graph RAG shows a significant reduction in the hallucination rate, which means that the Knowledge Graph retrieval method can help to retrieve grounded information that is more accurate.

However, Knowledge Graph RAG, building and sustaining a high-quality Knowledge Graph demands a large amount of effort, as it needs to be frequently refreshed to stay accurate and thorough. Moreover, the performance of KG-RAG also heavily relies on the quality of the foundational Knowledge graph. Consequently, if the graph is of low quality or incomplete, the accuracy of the retrieval could be impacted, since the retrieval is fully based on the connectivity of the information within the foundation graph. (Peng et al., 2023)

2.4.5 Comparison between different types of RAG

Table 2.3: Comparison between different types of RAG

Type of RAG	**Naïve RAG**	**Agentic RAG**	**Corrective RAG**	**Knowledge Graph RAG**
Core Architecture	Linear and single-agent pipeline	Multi-agent pipeline with various specialized agents	RAG with a corrective feedback mechanism	RAG with knowledge graph-based retrieval and reasoning
Key Mechanism	Basic retrieval to the generation pipeline	Orchestrator agent coordinating specialized agents	Automated feedback system for error correction	Reasoning via linked knowledge entities from a knowledge graph
Retrieval Method	BM25, TF-IDF similarity search	BM25, TF-IDF similarity search with parallel capability	Standard retrieval with corrective layer	Reasoning based from knowledge graph
Strengths	-Simple implementation -Low computational resources required	-Enhanced precision -Flexible task allocation - Improved adaptability	-Lower hallucination rate -Higher response factual accuracy	-Possesses reasoning capabilities -Can handle complex data

	-Short response times			relationships - Reduce hallucination rate
Limitations	-No corrective mechanisms - Limited complex query handling - Depends only on the retriever's capabilities	- High computational power needs -Complex agent orchestration needed	-High computational resources -Potential over-correction -Longer response times	-Complex graph maintenance - Performance depends on knowledge graph quality

2.5 Multi-agent framework

Multi-agent frameworks offer a uniform setting that enhances communication, collaboration, and decision-making among agents. These frameworks provide crucial tools, libraries, and protocols that facilitate the development of MAS while guaranteeing scalability and effectiveness. This chapter examines different multi-agent frameworks, emphasizing their structures, capabilities, and appropriateness for various application areas. Understanding these frameworks can help with the choices regarding the appropriate tools to create intelligent, adaptive, and cooperative multi-agent RAG systems.

2.5.1 LangChain

LangChain is a robust framework for developing and orchestrating LLM-based applications. It provided many intuitive tools and APIs for developer to implement in their LLM-driven application, which simplified the complexity of the development. (IBM, 2023). LangChain will serve as the abstraction layer for complex data source integration and the LLM mechanism, significantly reducing the time required for application development. This allows developers to modify just the template and library, rather than coding business logic from scratch. (Amazon Web Services, Inc., n.d.). Another key feature of LangChain is that it provides ways for developers to develop their custom pipeline with tools of their choice by providing chains (user-defined pipeline) in the framework (Oguzhan Topsakal and Tahir Cetin Akinci, 2023)

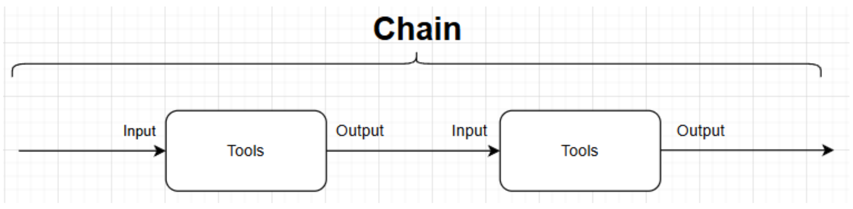


Figure 2.19: LangChain Tools Chaining Feature

A major advantage of LangChain is its capability to improve information retrieval through integration with vector databases, APIs, and various external information sources. Hence, this strength is especially beneficial in multi-agent RAG systems, where specialized agents need to gather, evaluate, and link relevant information effectively to generate an accurate result.

Moreover, the modular framework design also increases its scalability, enabling it to adapt to progressively complicated agent-based architectures. Lastly, Langchain's strong community backing is also one of its strengths, as extensive documentation is provided by the team along with example products for developers to learn from their peers.

Despite its advantages, the lack of a built-in orchestration feature for multi-agent systems is a major drawback for a multi-agent-based application, which means an extra solution needs to be developed to handle the interaction and synchronization for multiple agents. Additionally, it is a significant computational requirement, especially when maintaining long-term memory and executing ongoing reasoning in an advanced setting.

2.5.2 LangGraph

LangGraph is another AI agent workflow orchestrator created by the same team as LangChain as its successor. Therefore, LangGraph also possesses support from a strong community. But unlike LangChain, LangGraph uses a graph-based architecture to manage the relationship between each agent (Clark, 2025). This allows the developer to define the communication between the agents with a directed graph that provides a more dynamic and stateful workflow. LangGraph also provides persistent state management, allowing the agents to remember the conversation history and use it to provide contextual information to the task execution. (Wang and Duan, 2024). According to (Wang and Duan, 2024), LangGraph also provides the necessary tools to support different types of workflows such as hierarchical, human-in-the-loop, and single-agent systems which greatly improve the flexibility of this framework.

However, LangGraph also has some issues. Although its structured workflow provides flexibility, a solid foundation of graph-based programming knowledge is necessary to fully leverage its features, which can increase the learning curve for developers. Moreover, LangGraph incurs high computational costs to maintain stateful workflows and handle numerous agent interactions, especially in a multi-agent RAG system where the system must manage extensive data retrieval and synthesis assignments to produce accurate responses.

2.5.3 CrewAI

CrewAI is another lightweight and flexible multi-agent framework worth considering. The key characteristic for CrewAI is its emphasis on a role-playing architecture, in which CrewAI assigns specific roles to each agent and efficiently distributes tasks among them to achieve the ultimate goal (Winland et al., 2024). Through a role-based architecture, CrewAI allows developers to build specialized agents that collaborate dynamically, enhancing task execution while ensuring modularity and scalability.

A significant advantage of CrewAI is its simplicity and easy configuration, which makes it an ideal option for developers aiming to establish agent-based workflows quickly without the complexities of dependency-driven orchestration. Additionally, CrewAI also provides an automatic task delegation, and flexibility in task management feature which allows agents to communicate and assign tasks effectively (Duan and Wang, 2024). Additionally, CrewAI is capable of parallel task execution, which can increase both processing speed and efficiency in environments with multiple agents.

However, CrewAI is not suitable for very complex agent workflows, since it does not provide the sophisticated decision-making and dependency-tracking features available in graph-based systems such as LangGraph, which can limit the flexibility of this framework, making it not suitable for dynamically changing workflows that need to adapt constantly.

2.5.4 AutoGen

AutoGen is another multi-agent development framework developed by Microsoft. Similar to CrewAI, AutoGen utilizes a role-specific architecture, but it places more focus on agent-to-agent communication to maintain a multi-agent operation (Zeeshan et al., 2025). This allows AutoGen to ensure a modular system design where every agent can engage in a structured discussion to produce the most accurate result. Furthermore, AutoGen also supports different patterns of conversation to be implemented, which can increase the customization ability and flexibility of the agent structure (Wu, Bansal, et al., 2023)

As for the disadvantage of AutoGen. Its dependence on LLM-based decision-making is computationally heavy and requires significant processing capabilities to maintain ongoing agent interactions. Moreover, the coordination of its conversation agents adds complexity since overseeing several independent agents needs precise adjustments to avoid unnecessary processing, miscommunication, or contradictory actions, especially in a large conversational network. Lastly, AutoGen provides less defined task sequencing compared to other similar frameworks, potentially resulting in more unpredictable execution flows in complex implementations.

2.5.5 Comparison between different MAS development frameworks

Table 2.4: Comparison between different MAS development frameworks

Framework	Strengths	Weaknesses
LangChain	-Simplifies LLM-based development. - Flexible for custom pipelines. - Supports vector databases & external sources. - Strong community support.	- No built-in multi-agent orchestration. - High computational requirements.
LangGraph	- Graph-based structured workflows. - Persistent state management. - Supports flexible workflows. - Strong community support.	- Requires graph-based programming knowledge. - High computational cost.
CrewAI	- Role-based task delegation. - Lightweight & easy to configure. - Built-in task delegation. - Supports parallel execution.	- Lacks advanced decision-making & tracking. - Less flexible for dynamic workflows.
AutoGen	- Strong agent-to-agent communication. - Highly modular & customizable. - Supports various conversation patterns.	- High computational demand. - Complex agent coordination. - Less defined task sequencing.

2.6 Retrieval method

Two of the most important components in the RAG system are the retrieval component and the generation component. The quality of both the retrieval and generation components will directly impact the accuracy and quality of the system's final response to the user query. The retrieval component will be in control for recognizing and extracting the relevant information from any given database based on the user's query before passing the relevant information to the generation component for response elaboration and generation. Thus, without an effective retrieval component, the generation component's response accuracy will also be hindered. This section will explore different retrieval methods that are commonly used in a RAG system.

2.6.1 Sparse retrieval

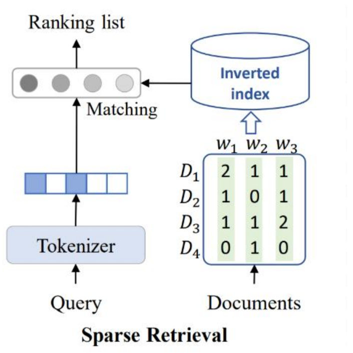


Figure 2.20: Sparse retrieval flowchart (Kumar, 2023)

Sparse retrieval is a retrieval algorithm that focuses more on keyword matching. which Sparser retrieval will compare the input document and text as a high-level vector where the values are mostly zero. Then, the input text is compared based on the existence of a specific keyword, where the more relevant of the current word is to the current input text, the higher score the sparse will return in the high value. One of the most popular sparse retrieval algorithms is BM25, which is the extension of TF-IDF. Thus, Sparse retrieval is excellent in use cases where keyword matching is crucial. Sparse retrieval's advantage is its lightweight nature, which demands little computing resources while delivering high-speed performance, making it work well for general search cases. As for the downside, sparse retrieval lack the ability to capture the underlying context and meaning of the text, which causes it to struggle with synonyms and complex text structure. (Milvus, 2025)

2.6.2 Dense retrieval

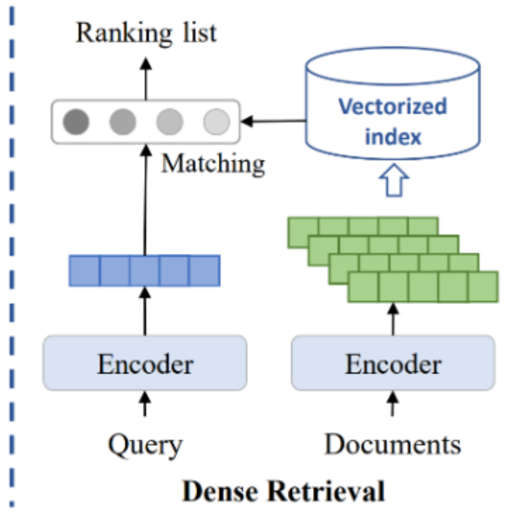


Figure 2.21: Dense retrieval flowchart (Kumar, 2023)

Dense retrieval is another popular retrieval method that is widely used in RAG systems and differs from sparse retrieval it focuses more on the semantic similarity of the text provided. First, the retrieval will utilize a pre-trained embedding model to convert the text provided into a dense group of vector representations in which more related text, such as ‘car’, ‘engine’ will be closer to each other in the vector space. These data will then be stored in a vector database and use algorithms such as, FANN search algorithm to retrieve the relevant information to the user’s query. Unlike sparse retrieval, this retrieval method can understand the underlying meaning of the text, which makes this method effective at getting the user's intent and allows the text structure and query to be more flexible. However, the downsides of this retrieval method are that it demands more computational resources due to the execution of the embedding model and the reduced interpretability since the vector format outcomes may be less clear compared to conventional keyword-driven approaches. (Milvus, 2025)

2.6.3 Hybrid retrieval

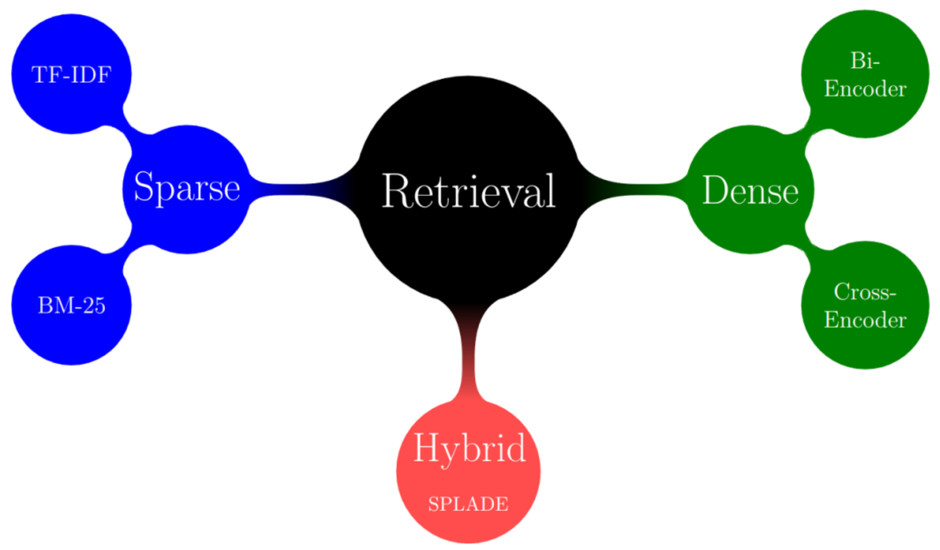


Figure 2.22: Hybrid retrieval flowchart (Khan, 2024)

Hybrid retrieval is a combined retrieval method that utilizes dense and sparse retrieval methods to perform searching. This can be achieved by combining the scores from a sparse retriever's result with a dense retriever's result or by using a re-ranker module to rank the results obtained from one approach with the other. This way, hybrid retrieval can be more flexible to adapt to the complex scenario where both keyword and semantic similarity are required to retrieve an accurate piece of information. While hybrid systems tend to be complex to coordinate and it is more resource-heavy because of the retrieval, reranking or weighting, they frequently produce more reliable outcomes, making them an excellent choice for multi-agent RAG frameworks that manage various document structures.

2.6.4 Knowledge Graph Retrieval

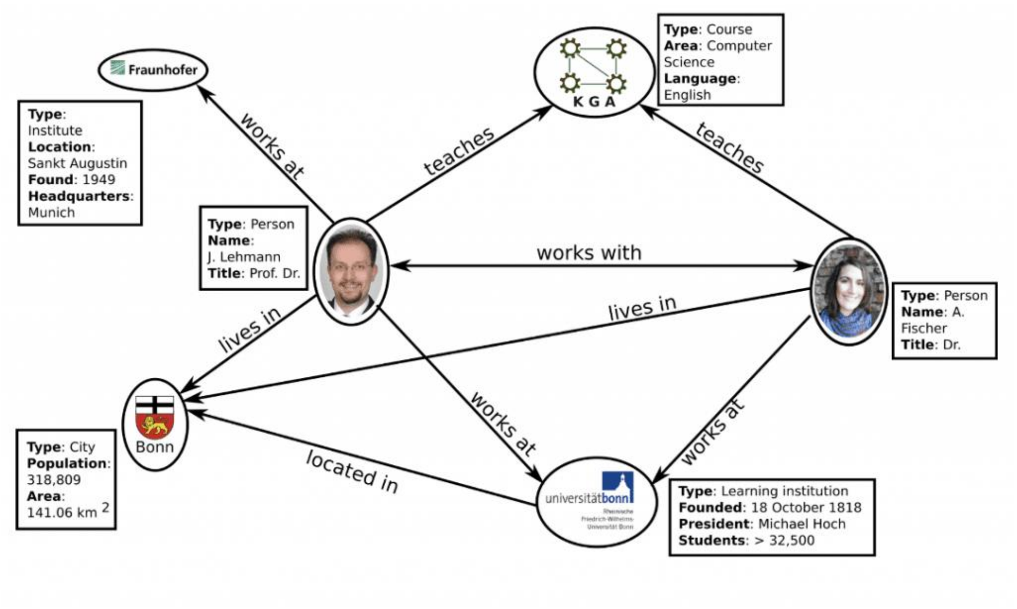


Figure 2.23: Knowledge Graph retrieval flowchart (Knight, 2025)

Knowledge Graph retrieval is a retrieval method that utilizes a more sophisticated Knowledge Graph for information retrieval. Knowledge Graph is a technique where data is stored as a node with an interconnected link to relevant nodes with a directed link. Thus, this structure allows meaningful information to be extracted with reason-based retrieval, which is particularly important in fields such as law and medicine where information relationships and logical reasoning are essential to retrieve relevant information. Although it provides significant accuracy and contextual richness, Knowledge Graph retrieval the Knowledge Graph retrieval method retrieval quality may vary depending on the completeness of the Knowledge Graph, and the Knowledge Graph requires regular maintenance to keep the knowledge base updated. (Peng et al., 2023).

2.6.5 Comparison between different retrieval methods

Table 2.5: Comparison between different retrieval methods

Retrieval Method	**Advantages**	**Disadvantages**
Sparse Retrieval	- Fast, lightweight, effective for keyword-based queries	- Cannot understand context or synonyms, - Limited to exact matches
Dense Retrieval	- Understands context and intent - Flexible query structure	- High computational cost - Low interpretability
Hybrid Retrieval	- Balances keyword precision with semantic understanding - Improved accuracy	- Complex integration - Resource-intensive
Knowledge Graph Retrieval	- Provides reasoning-based results - Highly contextual	- Requires complex setup, ongoing maintenance - Dependency on knowledge graph quality

2.7 Conclusion

To summarize, the literature review on design patterns for multi-agent systems, types of RAG models, and multi-agent frameworks offers important insights for creating an efficient local multi-agent RAG system. The results emphasize the need to employ different strategies to tackle issues in information retrieval, agent collaboration, and response formulation for a multi-agent RAG system. The strengths and weaknesses of different aspects are also reviewed to find the appropriate stacks and designs to be implemented in this project.

The following components will be considered for the development of this project.

1. A hierarchical multi-agent architecture
   This architecture is used to coordinate specialized agents and ensure effective task distribution. Streamlining the task distribution process and increasing the modularity of this system.
2. Agentic mechanisms
   This design is used in this system to effectively distribute tasks to the most suitable agent. Increasing the adaptability of the system.
3. Graph-based workflow orchestration
   This orchestration technique is used to manage the task distribution, communication, and relationships between agents, aiming to increase the modularity and the clarity of the system
4. Hybrid-based retrieval techniques
   This retrieval technique is used to ensure more accurate retrieval results and reduce the rate of hallucination for both keywords and similarity retrieval tasks.

CHAPTER 3

METHODOLOGY AND WORK PLAN

3.1 Introduction

This section will discuss the development methodology, work plan, and development tools used to develop this project. A systematic development methodology is implemented to ensure steady improvement, modular execution, and continuous refinement. Each phase in the development methodology will be thoroughly discussed with its intention and aim. Additionally, this section will break down each phase into well-defined tasks, which will be mapped into a comprehensive Work Breakdown Structure (WBS) to gain. Lastly, this section will outline the tools and framework used during the project development, such as a vector database and chunking tools. These technologies support the system's fundamental functions, including document intake, semantic search, natural language creation, and performance assessment while maintaining self-host.

3.2 Methodology

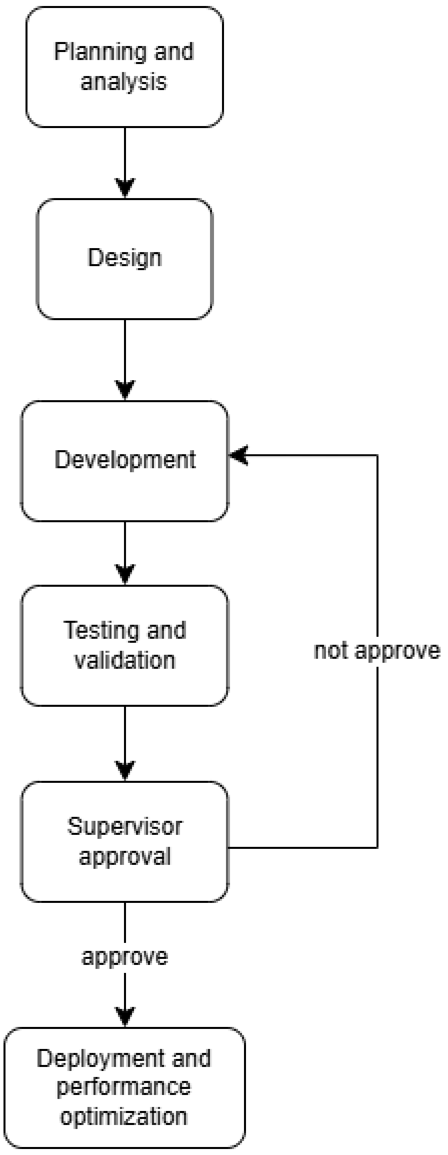


Figure 3.1: Project methodology

The development of the Self-Hosted Multi-Agent RAG System for Contextual Document Processing will follow the Prototype Development Model. This method is selected because of its iterative characteristics, enabling continuous refinement and the ability to incorporate user input to ensure the system maintains certain standards while preserving performance and security. The prototype approach allows for quick testing and verification of various multi-agent strategies before the system architecture is finalized, minimizing inefficiencies and ensuring a robust solution. The initiative will be split into five iterative stages, with every stage enhancing the system's capabilities according to testing outcomes and user feedback, ensuring a continuous enhancement toward a complete system.

The first phase is requirement analysis and project planning. During this phase, the project objective will be defined along with the problem statement and the scope of the project, which will ensure the objective is feasible to develop. The requirements will also be collected from surveys to collect requirements regarding the functional and non-functional requirements of the project. The multi-agent design will also be established by specifying agent roles and the tools involved for each agent during this phase. Additionally, the tools that are required for the development of this system will be properly identified and studied to ensure that they are compatible with the system's development.

For the second phase, the system design phase will be carried out to extend from the last phase to convert the collected information into a comprehensive technical outline. A system architecture will be designed during this phase to outline the overall system flow from ingress to response, agent functionality, and communication between different agents. In addition to initial use-case situations, with use-case diagram and use-case description are outlined. A rough UI/UX design prototype will also be developed with Streamlit to outline the user interaction process via the interface with some basic functionalities.

After the system design phase, the prototype development stage will be performed. The prototype of the preprocessing agent will be developed with Chonkie and marker-pdf for document chunking, and the primary hybrid retrieval system will be established by combining a document database, Weaviate which is capable of performing hybrid search while capable of scaling to high storage. A basic RAG-based retrieval and generation model will be created to lay a foundation for the more complex development needed in the future. The basic agent coordination for task delegation will be established, along with initial testing using RAGAS with sample datasets to confirm system workflow and fundamental retrieval accuracy. The MAS design will be introduced with different agent tasks and roles, including a reranking system using CrossEncoder to prioritize relevant information and a validation system for verifying facts. Subsequently, communication between agents will be established via the MAS framework, LangGraph, to support cooperative decision-making.

The following stage, Testing and Validation, will focus on assessing the system to measure its robustness via unit testing and integration testing. Functional testing will ensure each agent operates properly, while performance testing will evaluate the performance of the system component with various metrics, such as the accuracy metric. During the development, supervisor approval will be collected according to test outcomes to determine if the system's development is suitable to move into the next phase. If not, any recognized problems will be recorded, and the development process will return to the last prototype development phase to address the identified issues before proceeding.

For the last stage, Deployment and Performance Optimization, the system will be packaged for real-world deployment. Measures for security and privacy, including access control and options for on-premises deployment, will be implemented.

3.3 Development Tools

3.3.1 Marker

Marker is a Python library that is capable of converting content from a complex file format, such as PDF, into a separate markdown file while maintaining the original formatting. Different from other PDF file content scrapers, Marker is capable of both the standard PDF parsing method and the OCR method, making it capable of extracting content from a scanned or image-based PDF file. Additionally, Marker is also capable of extracting and preserving the content in tabular format, an important feature that other PDF parsing solutions lack. In this system, Marker will be the ingest point of the user's uploaded file, used to parse the content of the PDF file into markdown format to assist later processes like information chunking and storing.

3.3.2 Chonkie

Chonkie is an easy-to-use and lightweight Python library that is designed to breakdown long document content into smaller, meaningful chunks of text. This process is essential for this system because it allows the system to extract only the most relevant sections of a document to the user query, instead of returning complete documents. Chonkie supports various built-in chunking strategies such as SemanticChunking, SentenceChunking and SDPMChunking that can change based on user preference. Thus, Chonkie will be implemented into the Preprocessing agent, making sure that the document can be segmented for more effective retrieval.

3.3.3 LangGraph & Langchain

Langchain will be integrated into this system to utilize some tools like ChatOllama which will streamline the integration with the Ollama LLM model, MarkdownHeaderTextSplitter that will support the chunking process and more that will speedup the development of this project. In this system. LangGraph will be used to turn the system into a graph-based workflow, and the agents, such as preprocessing and generation agents, will function as nodes with edges to link nodes from one to another and to show data passing and flow between nodes. By organizing the workflow as a graph, a new agent can be added into the system with minimum interference to the current workflow, improving the flexibility of the system.

3.3.4 CrossEncoder reranking

To implement a hybrid retrieval method into the system, CrossEncoder reranking will be utilized to evaluate the relevance of the information retrieved from the hybrid retrieval (BM-25) and rank the outcome based on a deeper understanding of the context with a pre-trained CrossEncoder model. Despite being more computationally demanding than the other retrieval methods, CrossEncoder reranking can greatly improve the quality of the retrieval information by minimizing irrelevant or unclear results. Thus, CrossEncoder reranking will be used in this system to evaluate and rank the results from both the hybrid retrieval.

3.3.5 Ollama

Ollama is an open-source project that can act as a platform to run various and download open-source LLM models such as Llama, Gemma, and Mistral locally. Ollama provides an easy-to-use solution that allows users to interact directly via a command-line interface for model installation and execution. Additionally, Langchain provided seamless integration with Ollama via the LLM wrappers, which can simplify the LLM handling operation in this system. In this system, Ollama is used as the local LLM hosting engine that will be used in various phases in this project, such as the generation agent and query-enhancing agent. Thus, Ollama will be used in this system to integrate local LLM into this system.

3.3.6 RAGAS

RAGAS is an open-source LLM benchmark framework to evaluate the accuracy and performance of LLM-based applications, which provides built-in methods and metrics, such as Contextual Relevancy and Faithfulness, that can be used to measure the performance of the generated output. In this project, RAGAS will be used to assess the retrieval and generation accuracy of the RAG component by comparing produced responses with another built in LLM-as-judge approach to determine the effectiveness of the model in obtaining contextually relevant data and determining if the produced responses are accurate and consistent with the initial source materials. Thus, RAGAS will be a tool to evaluate the performance of the agents in this system.

3.3.7 Streamlit

Streamlit is an open-source Python framework that allows users to build an interactive data Python application with minimal coding requirements. Streamlit automatically handles the element rendering without needing complex syntax, making it suitable for the quick prototyping nature for the development of this project. In this project, Streamlit will be used to create the first prototype for Project 1, focusing on showing the core functionality of the system in an easy-to-use web-based interface.

3.4 Proposed Workplan

3.4.1 Work Breakdown Structure

1. Project Planning & Analysis

1.1 Conduct Title research

1.2 Outline problem statements and possible problem solution

1.3 Define project objective and scope

1.4 Requirement collection

1.5 Conduct literature review

1.5.1 Study of different Multi-Agent Systems (MAS)

1.5.2 Study of different RAG architecture types

1.5.3 Analysis of different retrieval methods

1.5.4 Analysis of different MAS development frameworks

1.5.5 Identification of suitability for things covered in this system

2. System Design

2.1 Design system architecture

2.1.1 Define agent functionalities

2.1.1.1 Identify functions needed

2.1.1.2 Identify the tools needed for agent

2.1.2 Define communication flow between agent

2.2 Design UI/UX interface and user interaction flow

2.3 Create UML diagrams

2.3.1 Create use case diagram

2.3.2 Create activity diagram

3. System prototype development

3.1 Set up local development environment

3.1.1 Install dependencies and libraries

3.1.2 Install LLM models

3.1.3 Configurate database

3.2 System module development

3.2.1 Develop document management module

3.2.1.1 Develop document upload system

3.2.1.2 Develop CRUD method for documents

3.2.2 Develop data preprocessing pipeline

3.2.2.1 Implement Marker for PDF extraction

3.2.2.2 Implement Chonkie chunking module

3.2.2.3 Implement Nomic for embedding generation

3.2.2.4 Develop storing function for relevant information

3.2.3 Develop retrieval pipeline

3.2.3.1 Create Hybrid retrieval method

3.2.3.2 Implement Reranker with Cross Encoder

3.2.4 Integrate LLM for response generation module

3.2.5 Conversation History Management Module

3.2.5.1 Develop user interactions and response history module

3.2.5.2 Referencing response history

3.3 Multi-Agent Workflow Implementation

3.3.1 Establish specialized agent role

3.3.2 Implement LangGraph framework for multiagent workflow

3.3.3 Develop agent orchestrator

3.3.4 Integrate developed module into multi-agent workflow

3.3.4.1 Integrate data preprocessing pipeline into workflow

3.3.4.2 Integrate retrieval pipeline into workflow

3.3.4.3 Integrate generation pipeline into workflow

3.3.5 Implement web search

3.3.5.1 Develop general web search

3.3.5.2 Develop advance search from specialize database

3.3.6 Implement automated note generator agent

4. System testing and validation

4.1 Perform unit testing

4.2 Perform functional testing for modules

4.3 Evaluate the accuracy of retrieval and generation

4.4 Supervisor feedback

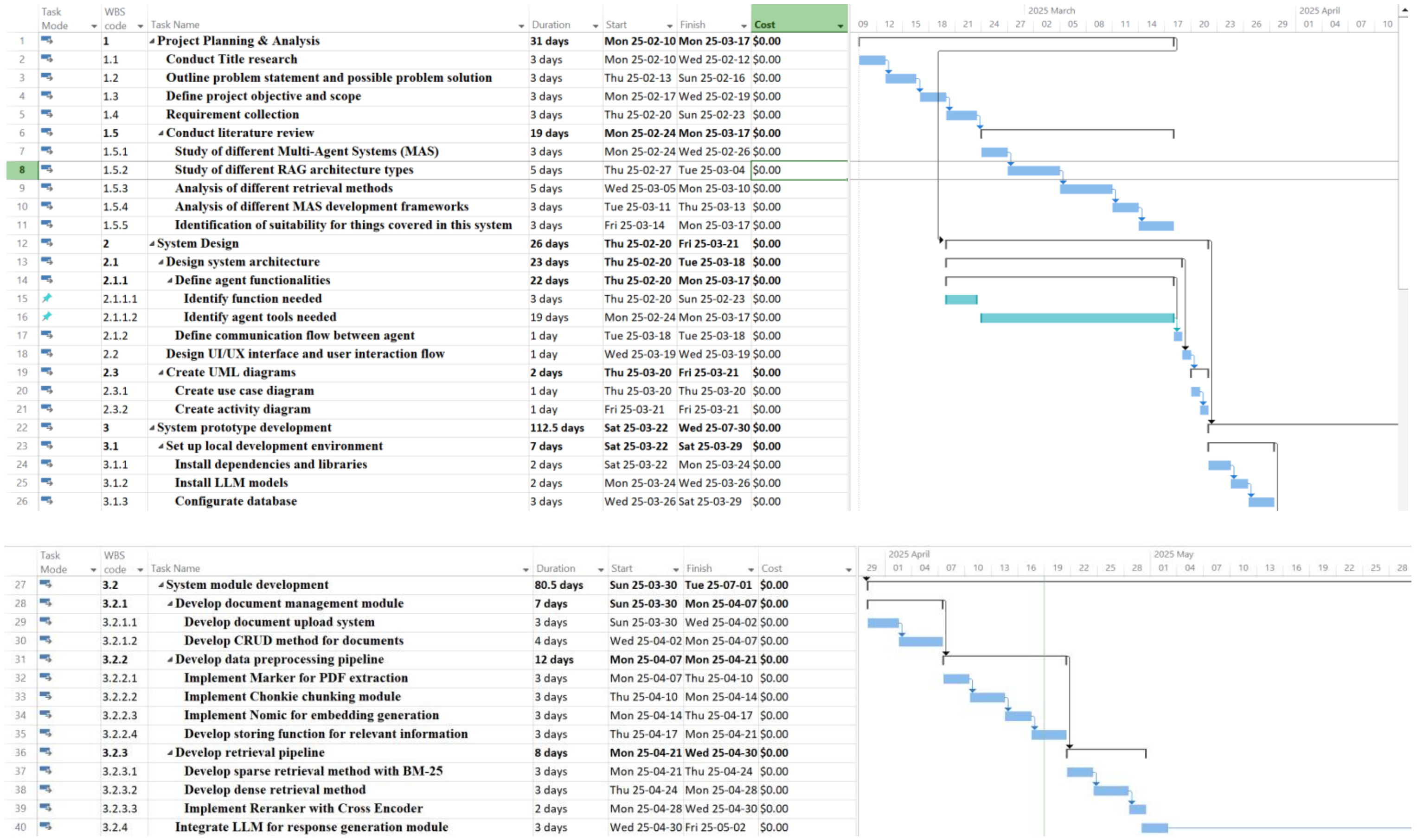
5. Deployment and Performance Optimization

5.1 Design and implement access control system

5.2 Final Testing and Verification

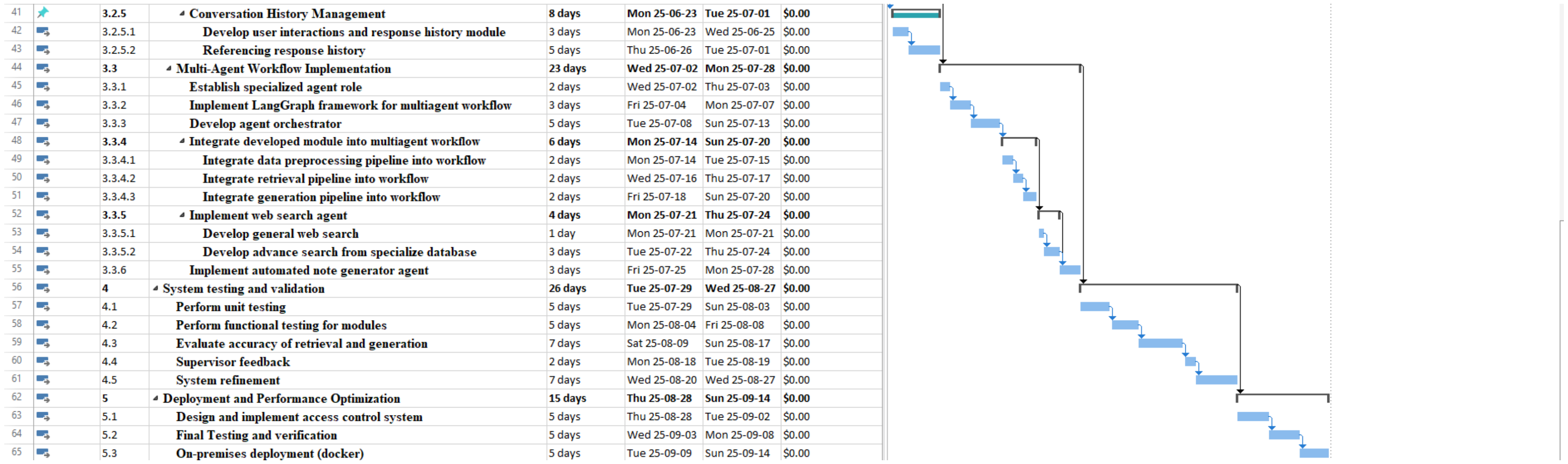
5.3 On-premises deployment (docker)

3.4.2 Gantt chart



	Task Mode	WBS code	Task Name	Duration	Start	Finish	Cost
1	➡	**1**	**Project Planning & Analysis**	**31 days**	**Mon 25-02-10**	**Mon 25-03-17**	**$0.00**
2	➡	1.1	Conduct Title research	3 days	Mon 25-02-10	Wed 25-02-12	$0.00
3	➡	1.2	Outline problem statement and possible problem solution	3 days	Thu 25-02-13	Sun 25-02-16	$0.00
4	➡	1.3	Define project objective and scope	3 days	Mon 25-02-17	Wed 25-02-19	$0.00
5	➡	1.4	Requirement collection	3 days	Thu 25-02-20	Sun 25-02-23	$0.00
6	➡	**1.5**	**Conduct literature review**	**19 days**	**Mon 25-02-24**	**Mon 25-03-17**	**$0.00**
7	➡	1.5.1	Study of different Multi-Agent Systems (MAS)	3 days	Mon 25-02-24	Wed 25-02-26	$0.00
8	➡	1.5.2	Study of different RAG architecture types	5 days	Thu 25-02-27	Tue 25-03-04	$0.00
9	➡	1.5.3	Analysis of different retrieval methods	5 days	Wed 25-03-05	Mon 25-03-10	$0.00
10	➡	1.5.4	Analysis of different MAS development frameworks	3 days	Tue 25-03-11	Thu 25-03-13	$0.00
11	➡	1.5.5	Identification of suitability for things covered in this system	3 days	Fri 25-03-14	Mon 25-03-17	$0.00
12	➡	**2**	**System Design**	**26 days**	**Thu 25-02-20**	**Fri 25-03-21**	**$0.00**
13	➡	**2.1**	**Design system architecture**	**23 days**	**Thu 25-02-20**	**Tue 25-03-18**	**$0.00**
14	➡	**2.1.1**	**Define agent functionalities**	**22 days**	**Thu 25-02-20**	**Mon 25-03-17**	**$0.00**
15	📌	2.1.1.1	Identify function needed	3 days	Thu 25-02-20	Sun 25-02-23	$0.00
16	📌	2.1.1.2	Identify agent tools needed	19 days	Mon 25-02-24	Mon 25-03-17	$0.00
17	➡	2.1.2	Define communication flow between agent	1 day	Tue 25-03-18	Tue 25-03-18	$0.00
18	➡	2.2	Design UI/UX interface and user interaction flow	1 day	Wed 25-03-19	Wed 25-03-19	$0.00
19	➡	**2.3**	**Create UML diagrams**	**2 days**	**Thu 25-03-20**	**Fri 25-03-21**	**$0.00**
20	➡	2.3.1	Create use case diagram	1 day	Thu 25-03-20	Thu 25-03-20	$0.00
21	➡	2.3.2	Create activity diagram	1 day	Fri 25-03-21	Fri 25-03-21	$0.00
22	➡	**3**	**System prototype development**	**112.5 days**	**Sat 25-03-22**	**Wed 25-07-30**	**$0.00**
23	➡	**3.1**	**Set up local development environment**	**7 days**	**Sat 25-03-22**	**Sat 25-03-29**	**$0.00**
24	➡	3.1.1	Install dependencies and libraries	2 days	Sat 25-03-22	Mon 25-03-24	$0.00
25	➡	3.1.2	Install LLM models	2 days	Mon 25-03-24	Wed 25-03-26	$0.00
26	➡	3.1.3	Configurate database	3 days	Wed 25-03-26	Sat 25-03-29	$0.00
27	➡	**3.2**	**System module development**	**80.5 days**	**Sun 25-03-30**	**Tue 25-07-01**	**$0.00**
28	➡	**3.2.1**	**Develop document management module**	**7 days**	**Sun 25-03-30**	**Mon 25-04-07**	**$0.00**
29	➡	3.2.1.1	Develop document upload system	3 days	Sun 25-03-30	Wed 25-04-02	$0.00
30	➡	3.2.1.2	Develop CRUD method for documents	4 days	Wed 25-04-02	Mon 25-04-07	$0.00
31	➡	**3.2.2**	**Develop data preprocessing pipeline**	**12 days**	**Mon 25-04-07**	**Mon 25-04-21**	**$0.00**
32	➡	3.2.2.1	Implement Marker for PDF extraction	3 days	Mon 25-04-07	Thu 25-04-10	$0.00
33	➡	3.2.2.2	Implement Chonkie chunking module	3 days	Thu 25-04-10	Mon 25-04-14	$0.00
34	➡	3.2.2.3	Implement Nomic for embedding generation	3 days	Mon 25-04-14	Thu 25-04-17	$0.00
35	➡	3.2.2.4	Develop storing function for relevant information	3 days	Thu 25-04-17	Mon 25-04-21	$0.00
36	➡	**3.2.3**	**Develop retrieval pipeline**	**8 days**	**Mon 25-04-21**	**Wed 25-04-30**	**$0.00**
37	➡	3.2.3.1	Develop sparse retrieval method with BM-25	3 days	Mon 25-04-21	Thu 25-04-24	$0.00
38	➡	3.2.3.2	Develop dense retrieval method	3 days	Thu 25-04-24	Mon 25-04-28	$0.00
39	➡	3.2.3.3	Implement Reranker with Cross Encoder	2 days	Mon 25-04-28	Wed 25-04-30	$0.00
40	➡	3.2.4	Integrate LLM for response generation module	3 days	Wed 25-04-30	Fri 25-05-02	$0.00

Figure 3.2: Project 1 timeline



41		3.2.5	Conversation History Management	8 days	Mon 25-06-23	Tue 25-07-01	$0.00
42		3.2.5.1	Develop user interactions and response history module	3 days	Mon 25-06-23	Wed 25-06-25	$0.00
43		3.2.5.2	Referencing response history	5 days	Thu 25-06-26	Tue 25-07-01	$0.00
44		3.3	Multi-Agent Workflow Implementation	23 days	Wed 25-07-02	Mon 25-07-28	$0.00
45		3.3.1	Establish specialized agent role	2 days	Wed 25-07-02	Thu 25-07-03	$0.00
46		3.3.2	Implement LangGraph framework for multiagent workflow	3 days	Fri 25-07-04	Mon 25-07-07	$0.00
47		3.3.3	Develop agent orchestrator	5 days	Tue 25-07-08	Sun 25-07-13	$0.00
48		3.3.4	Integrate developed module into multiagent workflow	6 days	Mon 25-07-14	Sun 25-07-20	$0.00
49		3.3.4.1	Integrate data preprocessing pipeline into workflow	2 days	Mon 25-07-14	Tue 25-07-15	$0.00
50		3.3.4.2	Integrate retrieval pipeline into workflow	2 days	Wed 25-07-16	Thu 25-07-17	$0.00
51		3.3.4.3	Integrate generation pipeline into workflow	2 days	Fri 25-07-18	Sun 25-07-20	$0.00
52		3.3.5	Implement web search agent	4 days	Mon 25-07-21	Thu 25-07-24	$0.00
53		3.3.5.1	Develop general web search	1 day	Mon 25-07-21	Mon 25-07-21	$0.00
54		3.3.5.2	Develop advance search from specialize database	3 days	Tue 25-07-22	Thu 25-07-24	$0.00
55		3.3.6	Implement automated note generator agent	3 days	Fri 25-07-25	Mon 25-07-28	$0.00
56		4	System testing and validation	26 days	Tue 25-07-29	Wed 25-08-27	$0.00
57		4.1	Perform unit testing	5 days	Tue 25-07-29	Sun 25-08-03	$0.00
58		4.2	Perform functional testing for modules	5 days	Mon 25-08-04	Fri 25-08-08	$0.00
59		4.3	Evaluate accuracy of retrieval and generation	7 days	Sat 25-08-09	Sun 25-08-17	$0.00
60		4.4	Supervisor feedback	2 days	Mon 25-08-18	Tue 25-08-19	$0.00
61		4.5	System refinement	7 days	Wed 25-08-20	Wed 25-08-27	$0.00
62		5	Deployment and Performance Optimization	15 days	Thu 25-08-28	Sun 25-09-14	$0.00
63		5.1	Design and implement access control system	5 days	Thu 25-08-28	Tue 25-09-02	$0.00
64		5.2	Final Testing and verification	5 days	Wed 25-09-03	Mon 25-09-08	$0.00
65		5.3	On-premises deployment (docker)	5 days	Tue 25-09-09	Sun 25-09-14	$0.00

Figure 3.3: Project 2 timeline

CHAPTER 4

PROJECT SPECIFICATION

4.1 Introduction

This chapter will properly identify the project's key requirements. Additionally, two UML diagrams: a use case diagram and an activity diagram, will be included to support the project requirements. Lastly, screenshots of the system's initial prototype will be provided to demonstrate and reinforce the system specifications.

4.2 Requirement specification

4.2.1 Functional requirement specification

Requirement ID	Requirement description	Status
FR01	The System shall allow users to log in to their existing account	Done
FR02	The system shall allow users to sign up for a new account	Done
FR03	The system shall allow users to upload their PDF document	Done
FR04	The system shall allow users to view a list of their uploaded files	Done
FR05	The system shall allow users to delete the file from the database	Done
FR06	The system shall allow users to submit a natural language query	Done
FR07	The system shall allow users to view the relevant information that the system has retrieved	Done
FR08	The system shall allow users to view the LLM-generated response	Done
FR09	The system shall allow users to generate a note via a query automatically	Done

FR10	The system shall allow users to change the system LLM into their desired local LLM model	Done
FR11	The system shall allow users to toggle the status of MCP services and web search	Done
FR12	The system shall retrieve the content of the internet if the web search setting is toggled on	Done
FR13	The system shall allow users to include their own MCP server details if the MCP service is used	Done
FR14	The system shall dynamically adjust the workflow based on the user's query	Done
FR15	The system shall retrieve the correct relevant chunk from the document based on the user query	Done
FR16	The system shall allow the user to summarize the content in the uploaded document	Done

4.2.2 Non-functional requirement specification

Requirement ID	Requirement description	Status
NFR01	The System shall be intuitive and easy to use for all types of users	Done
NFR02	The system shall be implemented entirely locally without depending on external services	Done
NFR03	The system shall prevent unauthorized access	Done

4.3 Use Case Diagram

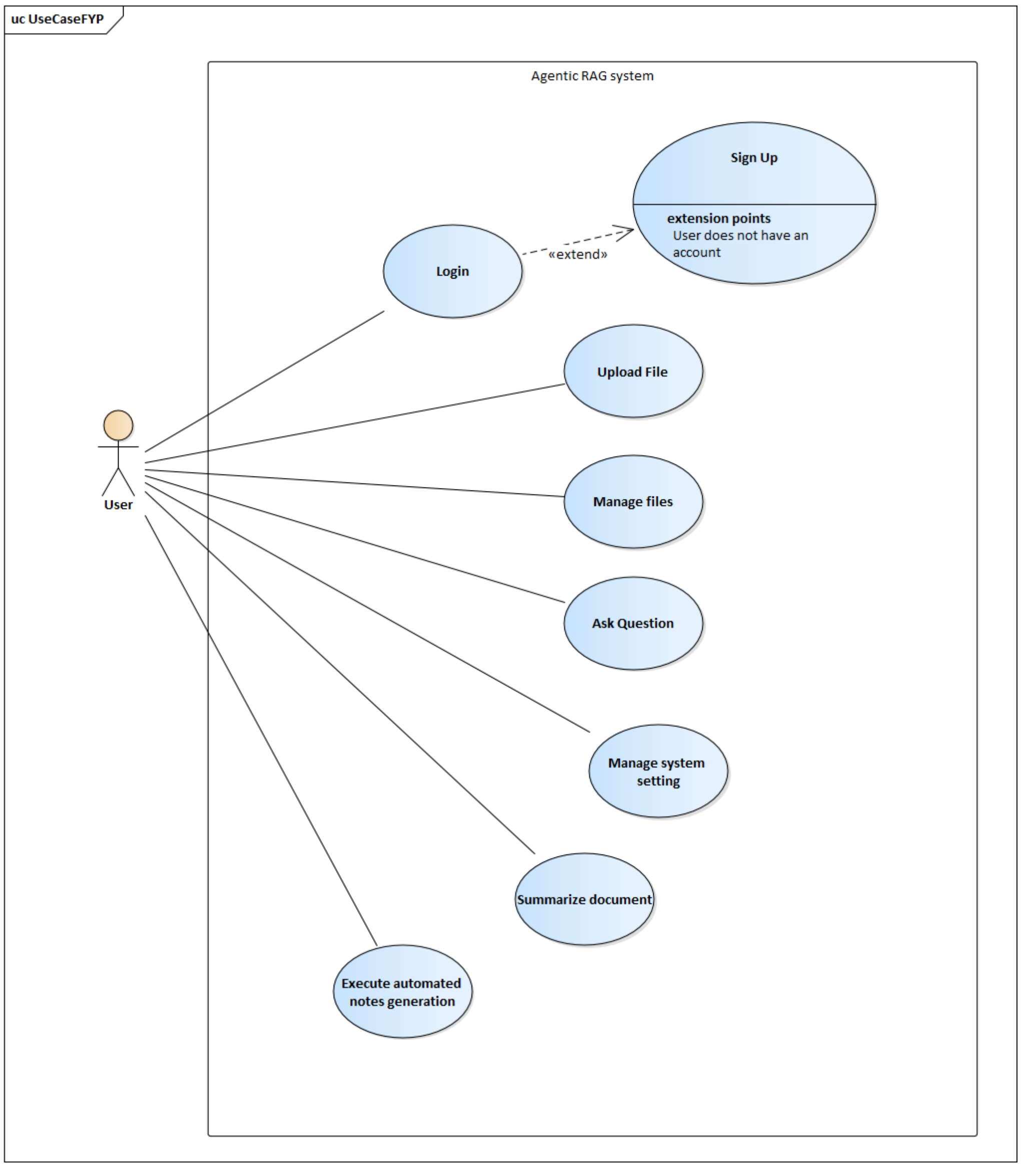


Figure 4.1: Use case diagram

4.4 Use Case description

4.4.1 Login Use Case

Use Case Name: **Login**		ID: **US01**	Importance Level: **High**
Primary Actor: **User**	Use Case Type: **Detail, Essential**		
Stakeholders and Interests:-			

Brief Description: **This use case describes how the user can login to the system**
Trigger: **The user wants to access the system.**
Relationships: Association **: User** Include **:-** Extend **: Sign Up- If user does not have an account** Generalization: **-**
Normal Flow of Events: **1. The user accesses the system** **2. The system displays the login interface** **3. The user enters email and password** **4. The system validates the user credentials** **5. If the user enters invalid credential, proceed to sub-flow 5.1** **6. If the user enters valid credential, proceed to sub-flow 6.1**
Sub-flows: **5.1 The user enters the wrong credentials** **5.1.1 The system displays the error message and prompts the user to re-enter** **5.1.2 Proceed to flow 3** **6.1 The user enters the correct credentials.** **6.1.1 The user proceeds to the system's main interface**

4.4.2 Sign Up Use Case

Use Case Name**: Sign Up**		ID**: US02**	Importance Level**: High**
Primary Actor**: User**	Use Case Type**: Detail, Essential**		
Stakeholders and Interests**:-**			
Brief Description**: This use case describes how the user can sign up for a new account**			
Trigger**: If the user has no existing account and wishes to sign up for one**			
Relationships: Association **: User** Include **: -** Extend **: -** Generalization**: -**			
Normal Flow of Events**:** **1. The user accesses the system** **2. The user selects the sign-up option in the login interface** **3. System display sign-up interface** **4. The user enters their email address, password, and confirms password** **5. The system validates the credentials entered** **6. If the user enters invalid information, proceed to sub-flow 6.1** **7. If the user enters valid information, proceed to sub-flow 7.1**			

Sub-flows: **6.1 The user enters invalid information** **6.1.1 The system displays the error message and prompts the user to re-enter, proceed to flow 3** **7.1 The user enters the valid information** **7.1.1 The user proceeds to the system's main interface**

4.4.3 Upload File Use Case

Use Case Name: **Upload File**	ID: **US03**	Importance Level: **High**
Primary Actor: **User**	Use Case Type: **Detail, Essential**	
Stakeholders and Interests:**-**		
Brief Description: **This use case describes how the user can upload their document into the system database**		
Trigger: **The user wants to upload their document to the system database for the system to reference**		
Relationships: Association **: User** Include **: -** Extend **: -** Generalization**: -**		
Normal Flow of Events: **1. The user selects the file upload in the navigation menu** **2. The system navigate to file upload interface**		

	3. **The user clicks the upload file option in the interface** 4. **The user selects one or more files from their local device with a file browser.** 5. **The user selects the submit option** 6. **The system checks the file** 7. **If the file is invalid, proceed to sub-flow 7.1** 8. **The system starts file processing** 9. **The system prompts a successful message**
Sub-flows: **7.1 The user's file is too big or an unsupported file format** **7.1.1 The system stops the upload process** **7.1.2 The system prompts an error message, proceed to flow 3**	

4.4.4 Manage File Use Case

Use Case Name: **Manage File**	ID: **US04**	Importance Level: **High**
Primary Actor: **User**	Use Case Type: **Detail, Essential**	
Stakeholders and Interests: **-**		
Brief Description: **This use case describes how the user can manage their uploaded document**		
Trigger: **The user wants to manage their file in the system**		
Relationships: Association **: User** Include **: -**		

Extend : - Generalization: -
Normal Flow of Events: 1. **The user selects the Manage file in the navigation menu** 2. **The system navigates to the Manage file interface** 3. **The system fetches files information from the database** 4. **If no file is found in the database, proceed to sub-flow 4.1** 5. **The system displays file information** 6. **The user checks the delete option for the document, proceeds to sub-flow 4 .1**
Sub-flows: **4.1 If no file is found in the database** **4.1.1 The system displays an error message** **5.1 If the user chooses to delete the document** **6.1.1 The user clicks the delete button** **6.1.2 The file will be deleted from the document.** **6.1.3 The system prompts a success message**

4.4.5 Ask question Use Case

Use Case Name: **Ask question**	ID: **US05**	Importance Level: **High**

Primary Actor**: User**	Use Case Type**: Detail, Essential**
Stakeholders and Interests**: -**	
Brief Description**: This use case describes how the system will respond when the user enters a query into the system**	
Trigger**: The user wants to ask a question**	
Relationships: Association **: User** Include **: -** Extend **: -** Generalization**: -**	
Normal Flow of Events**:** **1. The user navigates to the chat interface.** **2. The system navigates to chat interface** **3. The user enters their question in the text input field.** **4. The user clicks the submit button.** **5. The system analyzes the query.** **6. The system router determines the most suitable handling strategy for the query:** **7. If the request can be answered directly using the LLM's internal knowledge, proceed to Sub-flow 7.1.** **8. If the request requires additional information from documents or external sources, proceed to Sub-flow 8.1.** **9. The Generation Node creates a grounded response using both the LLM's knowledge and the retrieved information.** **10. The system prompt the final response to the user.**	

Sub-flows: **7.1 If query handled directly by LLM** **7.1.1 The router pass the query directly to generation node, Proceed to flow 9** **8.1 If query requires retrieval** **8.1.1 The Workflow Router forwards the query to the Retrieval Node.** **8.1.2 The Retrieval Node determines whether to use uploaded documents or external source** **8.1.3 If the system uses external search, proceed to sub-flow 8.1.3** **8.1.4 If the system uses document search, proceed to sub-flow 8.1.4** **8.1.3.1 If the system uses external search** **8.1.3.1.1 The ReAct agent dynamically selects the most suitable external tool:** • **General Web Search: Fetches the top 5 most relevant web pages, preprocesses the content, and applies hybrid retrieval to extract useful information.** • **Medical Knowledge (PubMed): Retrieves relevant research articles for medical-related queries.** • **Financial News (Yahoo Finance): Retrieves the latest finance and business news.** **8.1.3.1.2 The retrieved information is passed to the Generation Node for response generation, proceed to flow 9** **8.1.4.1 If the system uses document search** **8.1.4.1.1 The Retrieval Node fetches relevant passages from user-uploaded documents.** **8.1.4.1.2 The retrieved information is passed to the Generation Node for response generation, proceed to flow 9**

4.4.6 Change system setting Use Case

Use Case Name: **Change system setting**		ID: **US06**	Importance Level: **High**
Primary Actor: **User**	Use Case Type: **Detail, Essential**		
Stakeholders and Interests: **-**			
Brief Description: **This use case describes the user flow to change the system operational setting**			
Trigger: **The user wants to change the details of the system**			
Relationships: Association **: User** Include **: -** Extend **: -** Generalization: **-**			
Normal Flow of Events: 1. **The user selects the setting in the navigation menu** 2. **The system navigates to the setting interface** 3. **The system displays the setting menu with available system settings such as, LLM model selection, Web search configuration** 4. **The user modifies the setting** 5. **The user selects the save option in the settings page** 6. **The System checks the validation of the setting change** 7. **If the setting is valid, proceed to sub-flow 7.1** 8. **If any error occurs during validation, proceed to sub-flow 8.1**			

Sub-flows: **7.1 The setting is valid** **7.1.1 The system will save the setting** **7.1.2 The system prompts a success message** **7.1.3 The system will alter the system workflow according to the new settings.** **8.1 The setting is invalid** **8.1.1 The system will prompt an error message, proceed to flow 4**

4.4.7 Execute automated note generation Use Case

Use Case Name: **Execute automated note generation**	ID: **US07**	Importance Level: **High**
Primary Actor: **User**	Use Case Type: **Detail, Essential**	
Stakeholders and Interests: **-**		
Brief Description: **This use case describes how the user can initiate the automation note generation workflow**		
Trigger: **The user wants to create generate note for the last chat history or the response of current question**		
Relationships: Association **: User** Include **: -** Extend **: -** Generalization: **-**		
Normal Flow of Events: **1. The user selects the chat interface** **2. The system navigates to chat interface**		

3. **The user enters a prompt related to generating a note** 4. **The system analyzes the user's intent** 5. **The system generation the response** 6. **The system analyse if query require automated workflow** 7. **The system executes steps and tools in the workflow** 8. **If the generation is successful, proceed to sub-flow 8.1** 9. **If any error occurs during generation, proceed to sub-flow 9.1**
Sub-flows: **8.1 The generation is successful** **8.1.1 The system prompt success message** **9.1 Error occurs during generation** **9.1.2 The system stops the generation process** **9.1.3 The system prompt error message, proceed to flow 3**

4.4.8 Summarize document Use Case

Use Case Name: **Execute document summarization**	ID: **US08**	Importance Level: **High**
Primary Actor: **User**	Use Case Type: **Detail, Essential**	
Stakeholders and Interests: **-**		
Brief Description: **This use case describes how the user can initiate the document summarization workflow**		
Trigger: **The user wants to summarize the content in the uploaded document**		

Relationships:

Association : **User**

Include **: -**

Extend **: -**

Generalization**: -**

Normal Flow of Events**:**

1. **The user selects the chat interface**
2. **The system navigates to chat interface**
3. **The user enters a prompt related to document summarization**
4. **The system analyzes the user's intent**
5. **The system process to the summarization node**
6. **The system responds with 5 summarized passage to the frontend**
7. **If the summarization is successful, proceed to flow 7.1**
8. **If any error occurs during summarization, proceed to sub-flow 8.1**

Sub-flows**:**

7.1 The summarization is successful

7.1.1 The system prompt the final summarized content

8.1 Error occurs during summarization

8.1.2 The system stops the summarization process

8.1.3 The system prompt error message, proceed to flow 3

4.5 Activity diagram

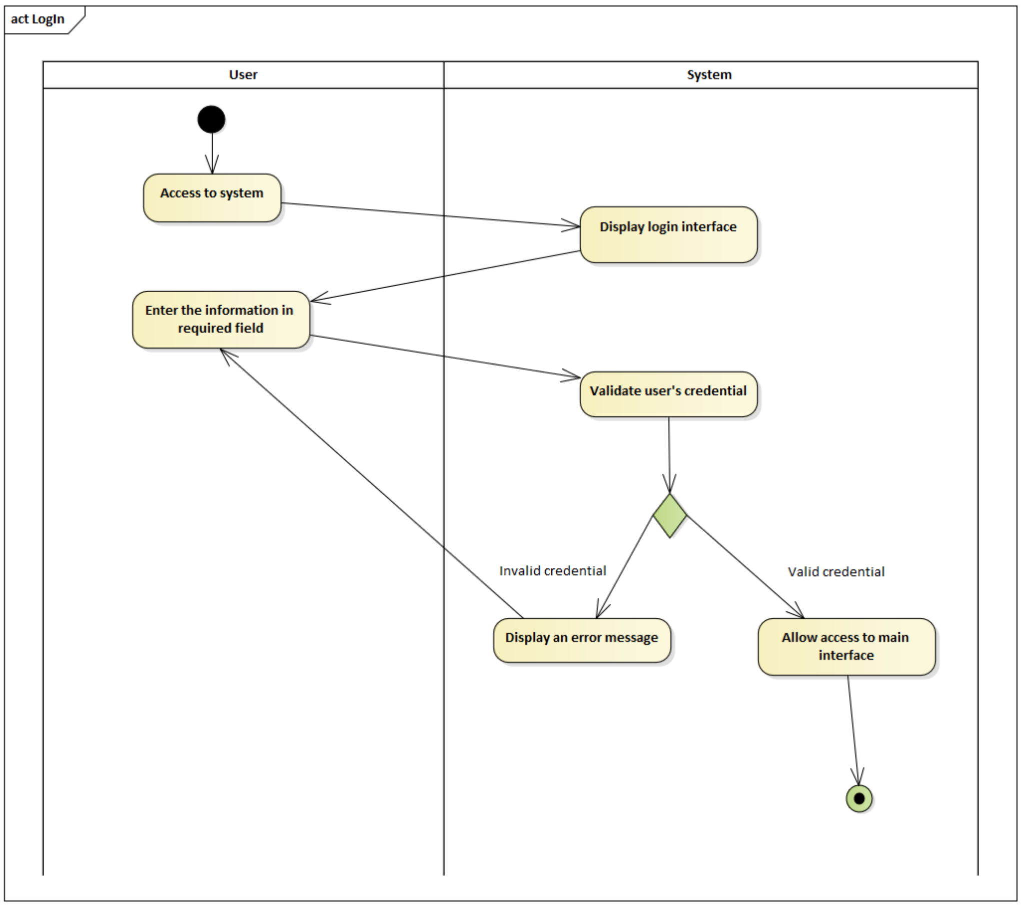


Figure 4.2: Login Activity diagram

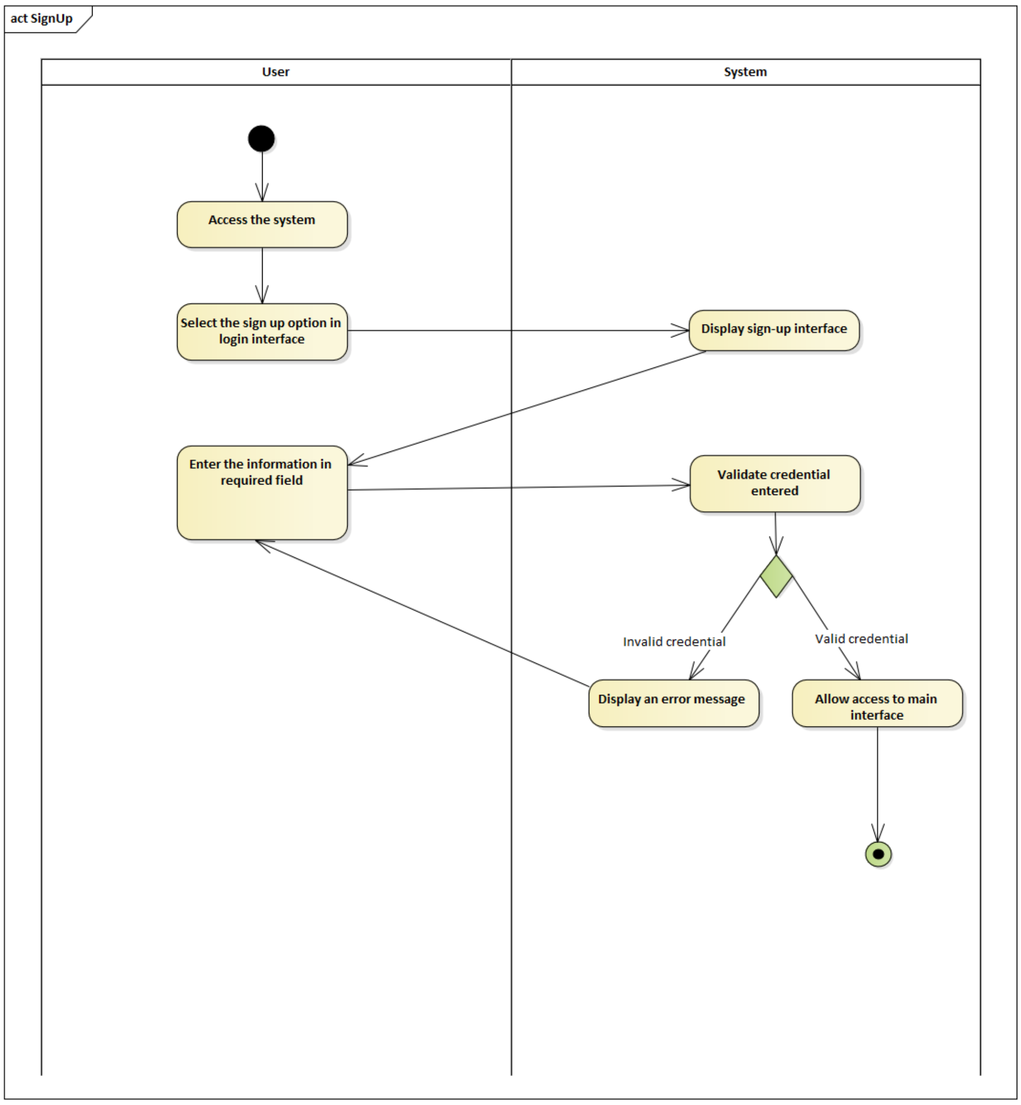

act SignUp
User
System
Access the system
Select the sign up option in login interface
Display sign-up interface
Enter the information in required field
Validate credential entered
Invalid credential
Valid credential
Display an error message
Allow access to main interface

Figure 4.3: Sign Up Activity diagram

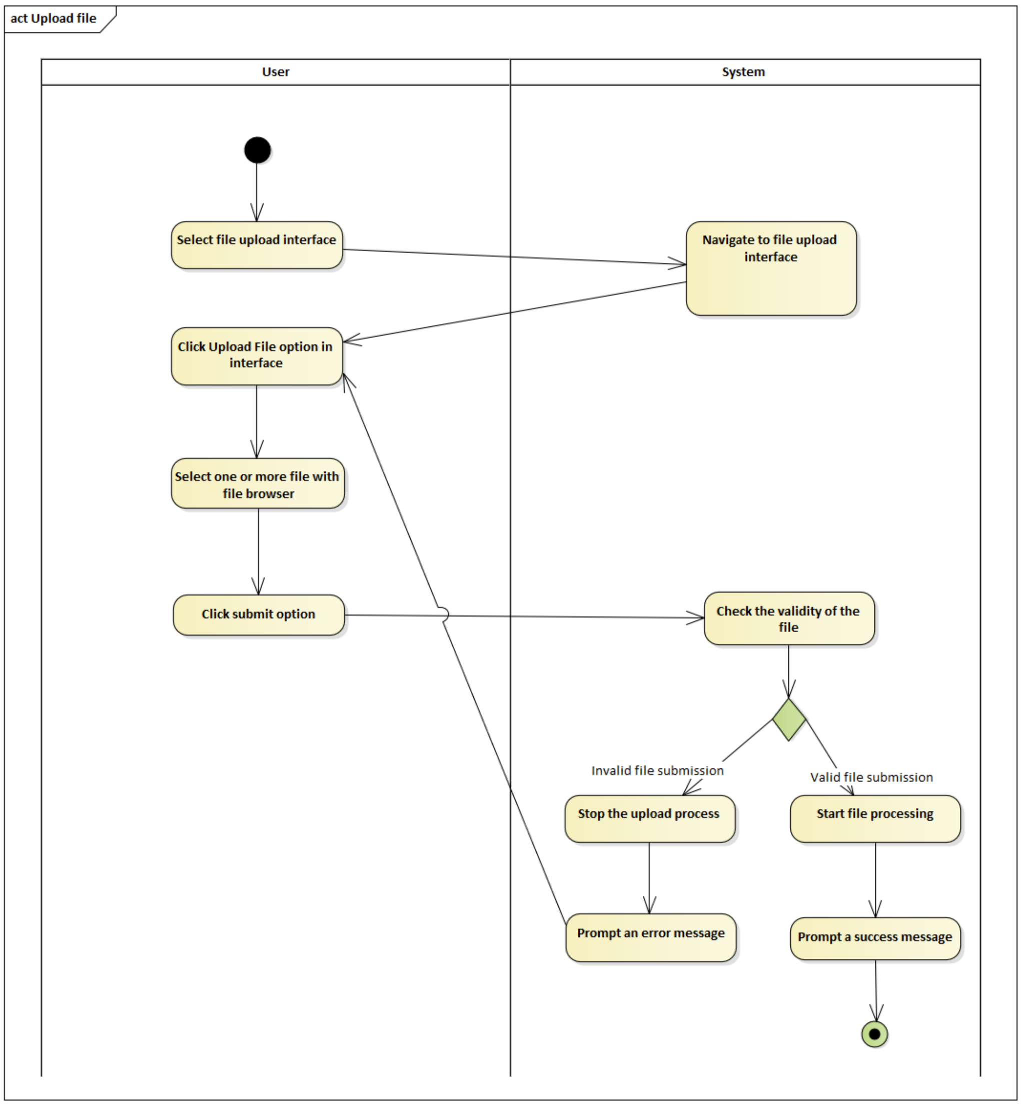


Figure 4.4: Upload file Activity diagram

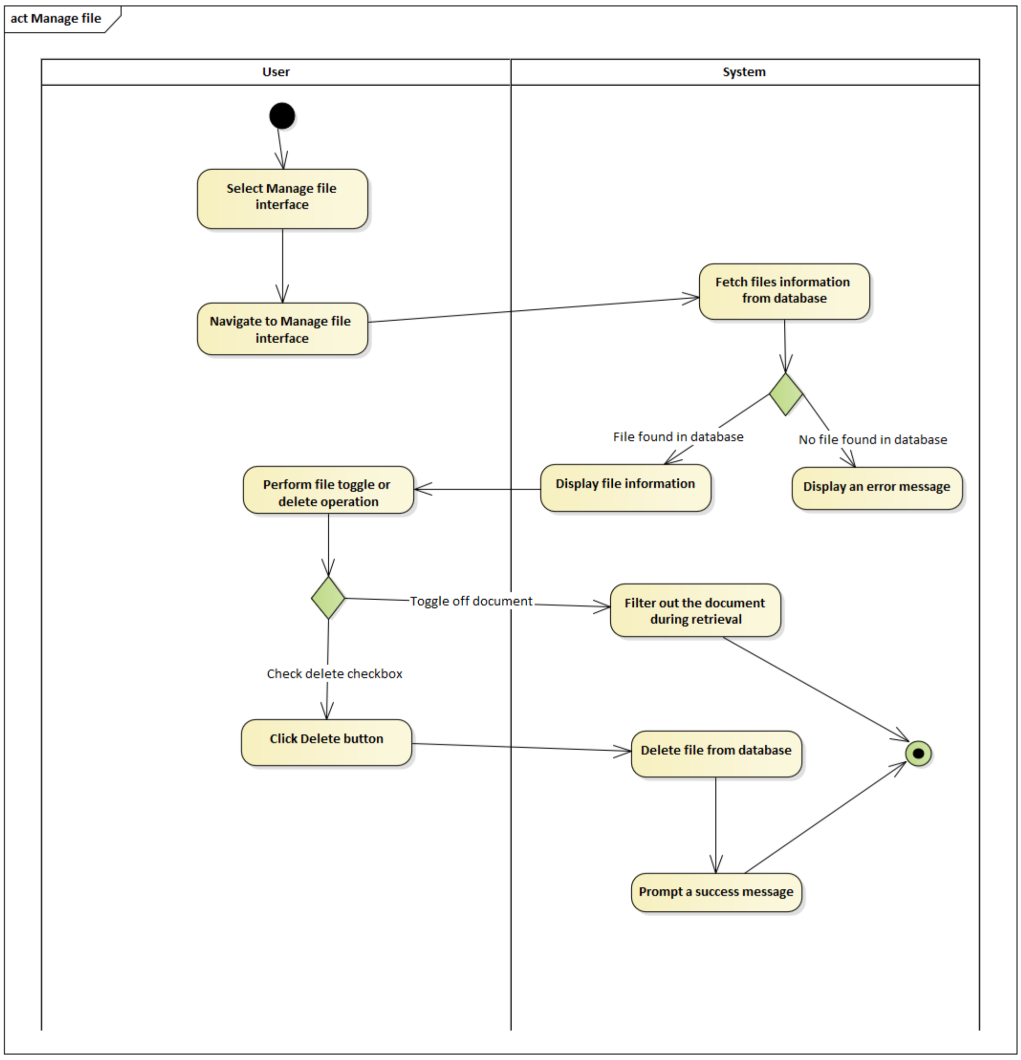

act Manage file
User
System
Select Manage file interface
Navigate to Manage file interface
Fetch files information from database
File found in database
No file found in database
Display file information
Display an error message
Perform file toggle or delete operation
Toggle off document
Filter out the document during retrieval
Check delete checkbox
Click Delete button
Delete file from database
Prompt a success message

Figure 4.5: Manage file Activity diagram

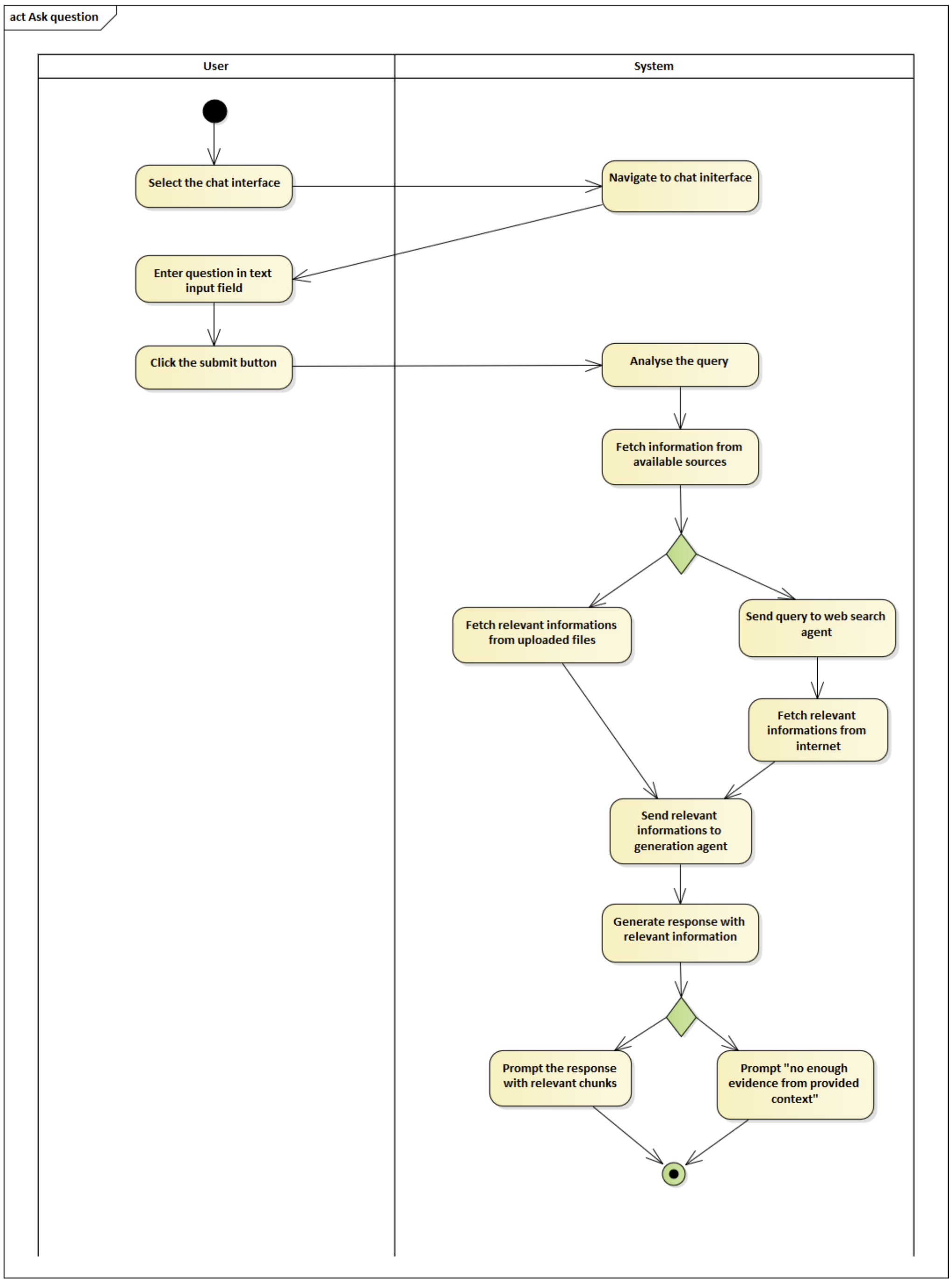

act Ask question
User
System
Select the chat interface
Navigate to chat inter̀face
Enter question in text input field
Click the submit button
Analyse the query
Fetch information from available sources
Fetch relevant informations from uploaded files
Send query to web search agent
Fetch relevant informations from internet
Send relevant informations to generation agent
Generate response with relevant information
Prompt the response with relevant chunks
Prompt "no enough evidence from provided context"

Figure 4.6: Ask Question Activity diagram

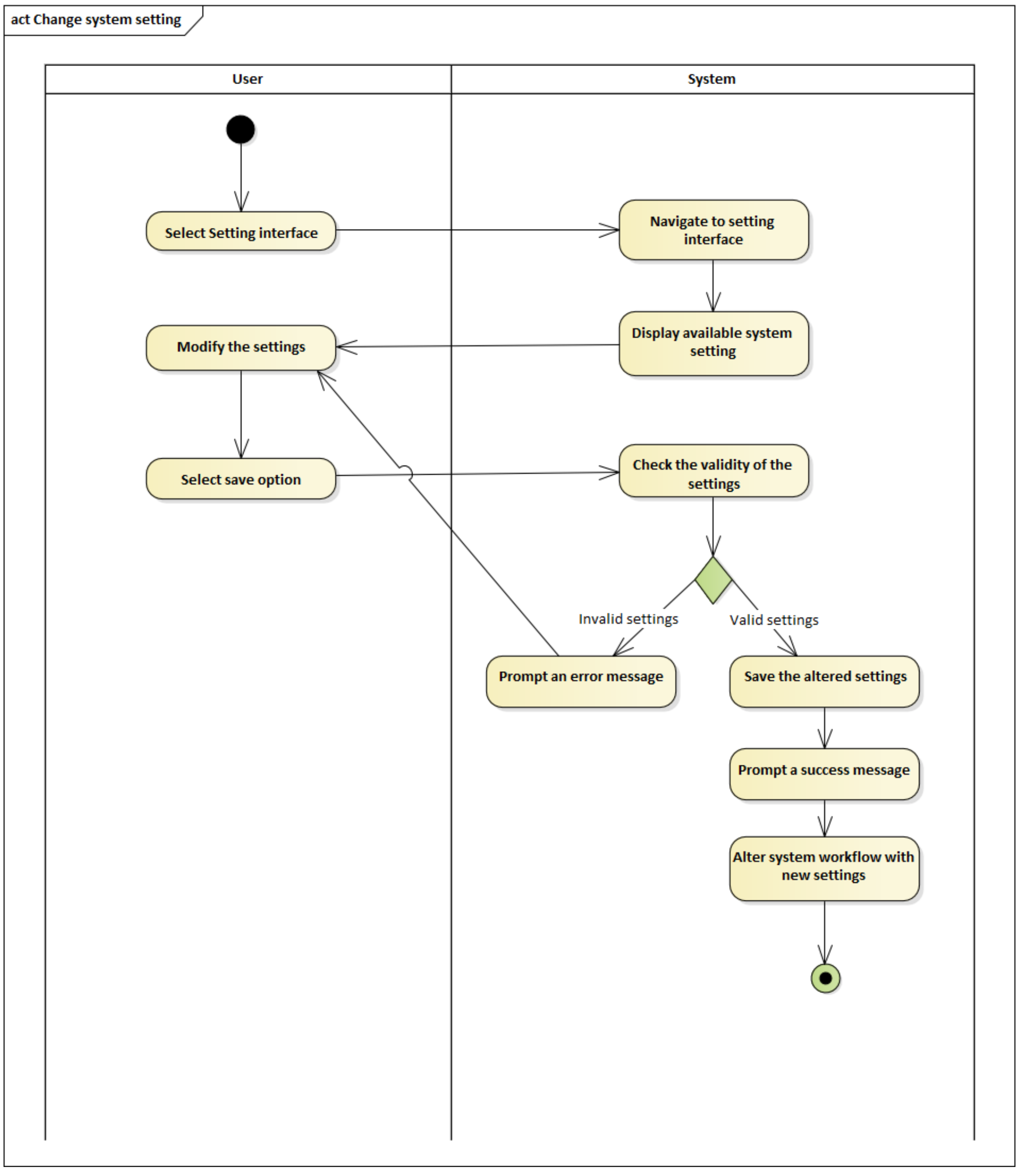

act Change system setting
User
System
Select Setting interface
Navigate to setting interface
Display available system setting
Modify the settings
Select save option
Check the validity of the settings
Invalid settings
Valid settings
Prompt an error message
Save the altered settings
Prompt a success message
Alter system workflow with new settings

Figure 4.7: Change system setting Activity diagram

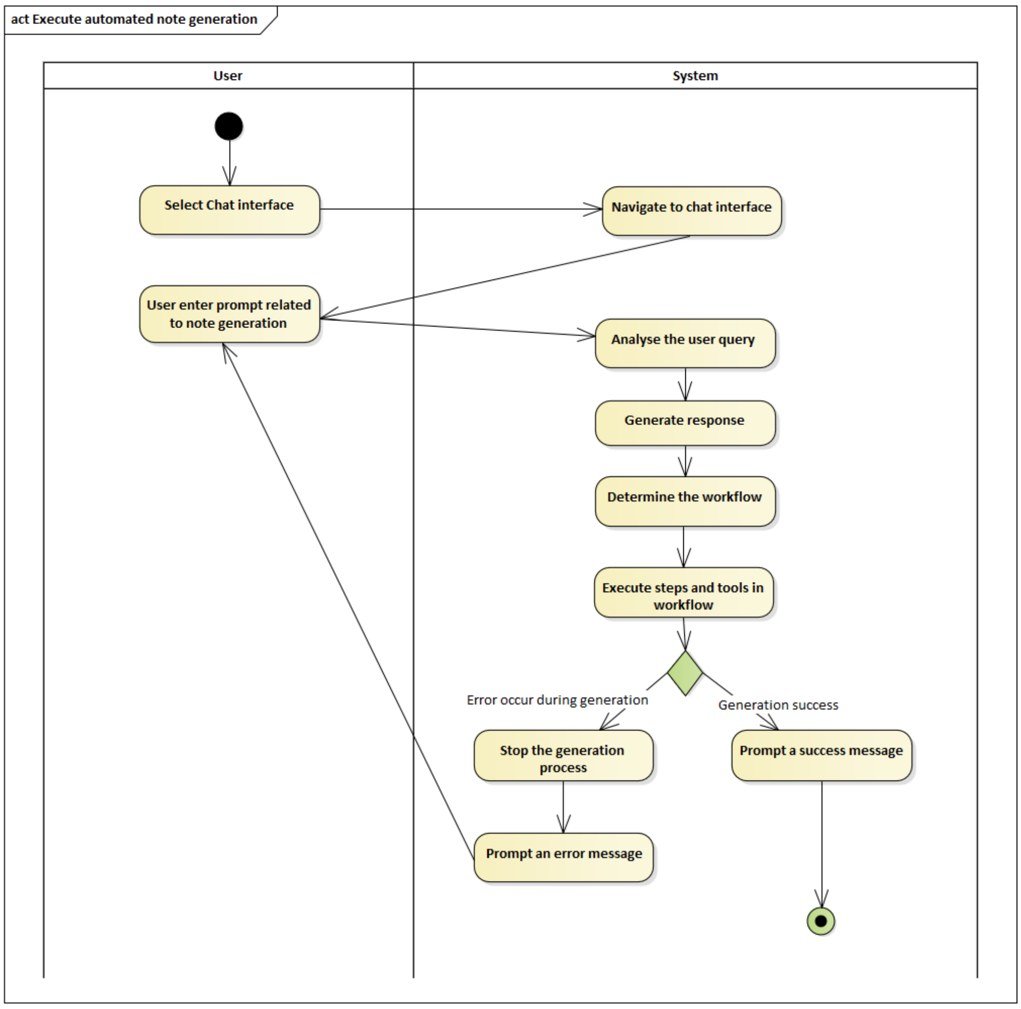

act Execute automated note generation
User
System
Select Chat interface
Navigate to chat interface
User enter prompt related to note generation
Analyse the user query
Generate response
Determine the workflow
Execute steps and tools in workflow
Error occur during generation
Generation success
Stop the generation process
Prompt a success message
Prompt an error message

Figure 4.8: Execute automated workflow Activity diagram

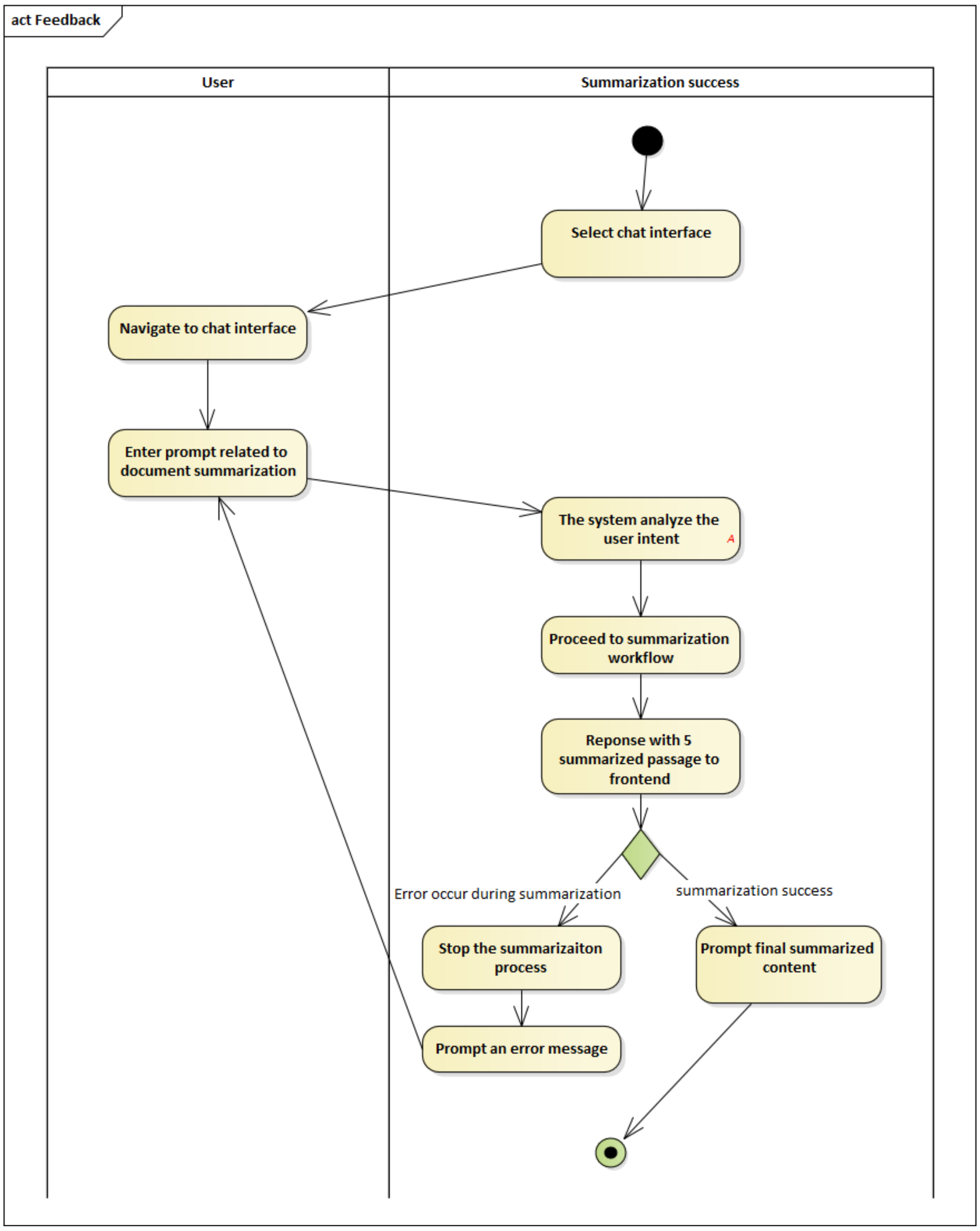

act Feedback
User
Summarization success
Select chat interface
Navigate to chat interface
Enter prompt related to document summarization
The system analyze the user intent
Proceed to summarization workflow
Reponse with 5 summarized passage to frontend
Error occur during summarization
summarization success
Stop the summarizaiton process
Prompt final summarized content
Prompt an error message

Figure 4.9: Summarized document activity diagram

4.6 Prototype section

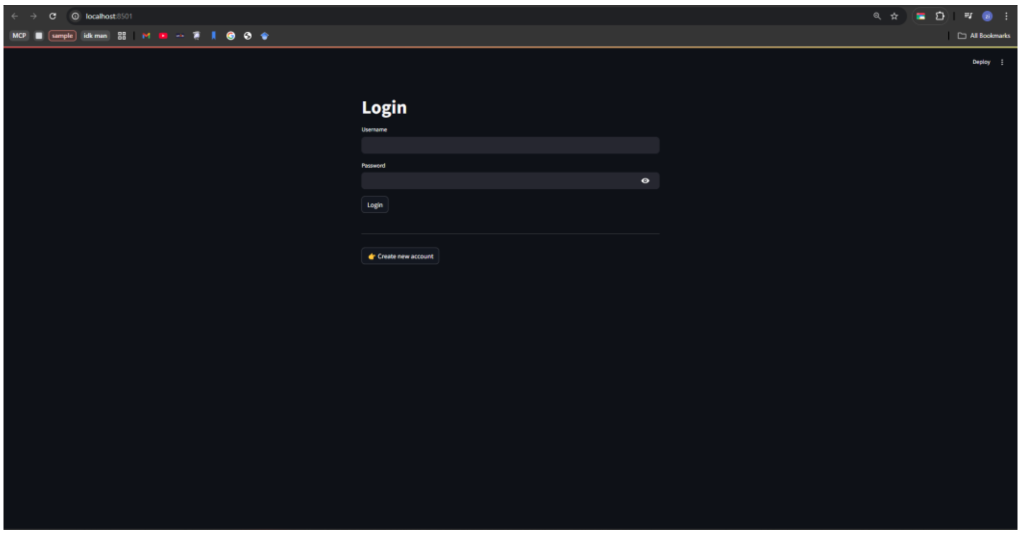
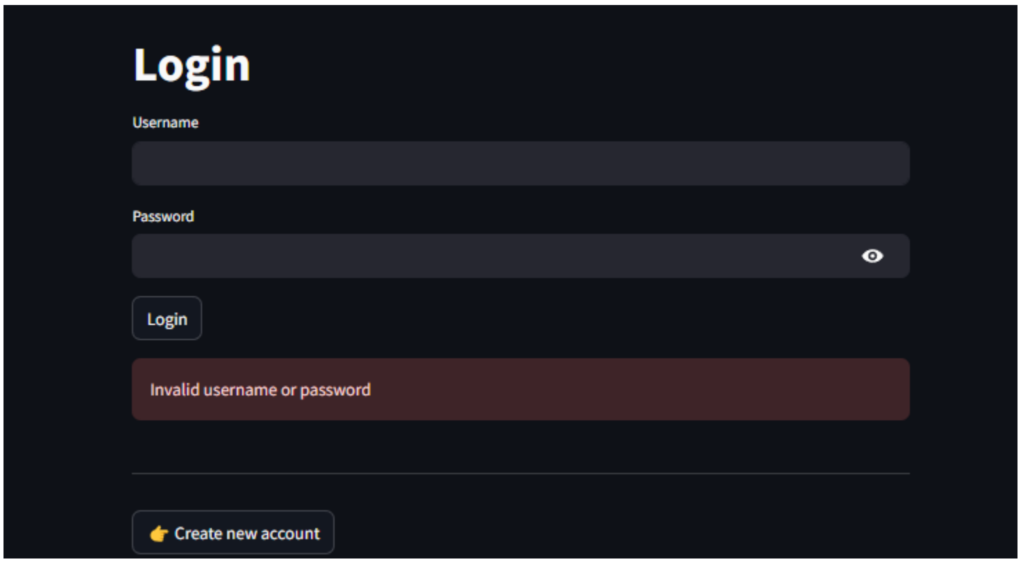


Figure 4.10: Login interface



Invalid username or password

Figure 4.11: Error in login interface

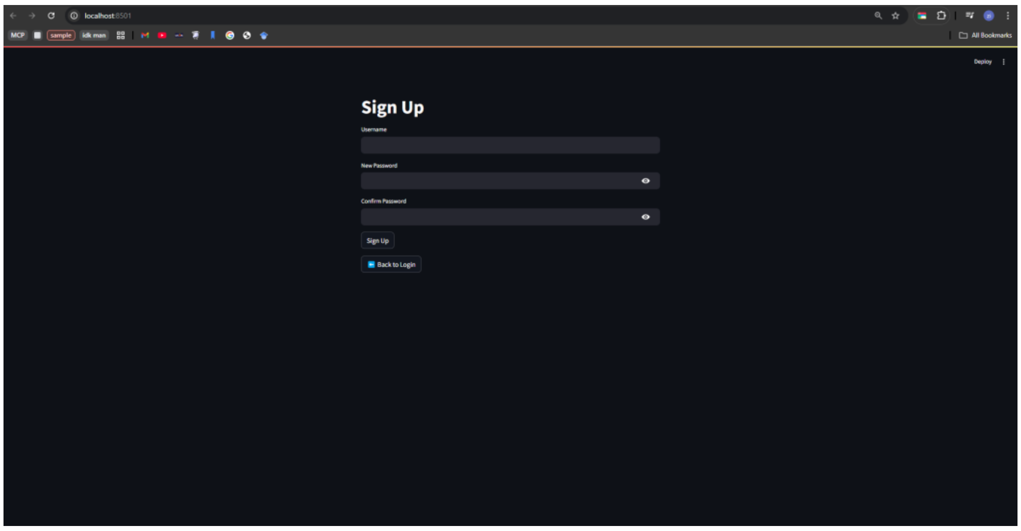
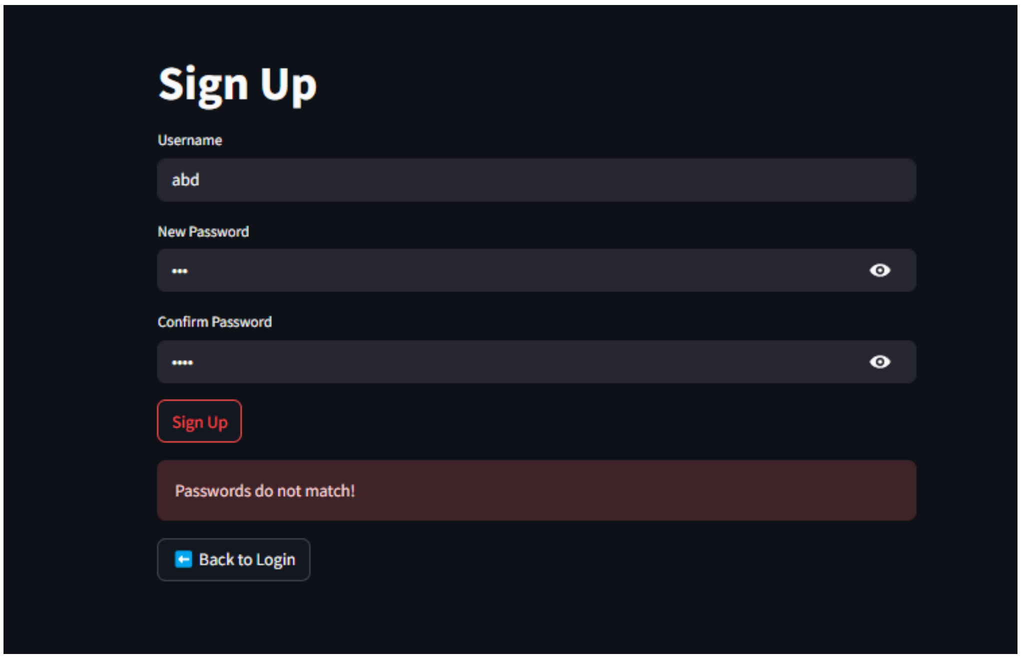
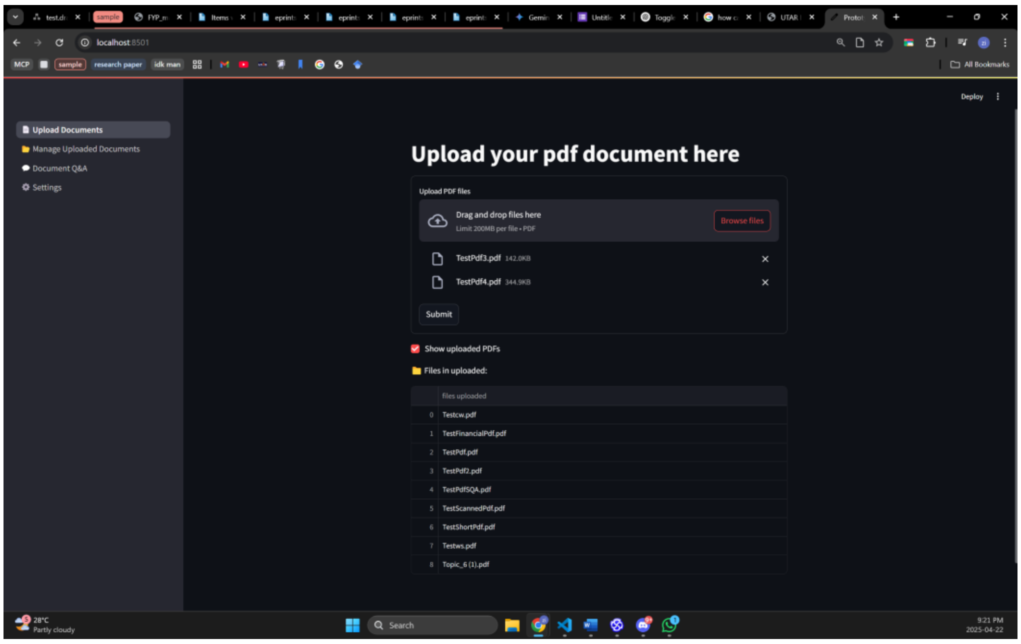


Figure 4.12: Sign Up interface



Passwords do not match!

Figure 4.13: Error in Sign Up interface



Upload your pdf document here

Upload PDF files

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

Browse files

TestPdf3.pdf 142.0KB

TestPdf4.pdf 344.9KB

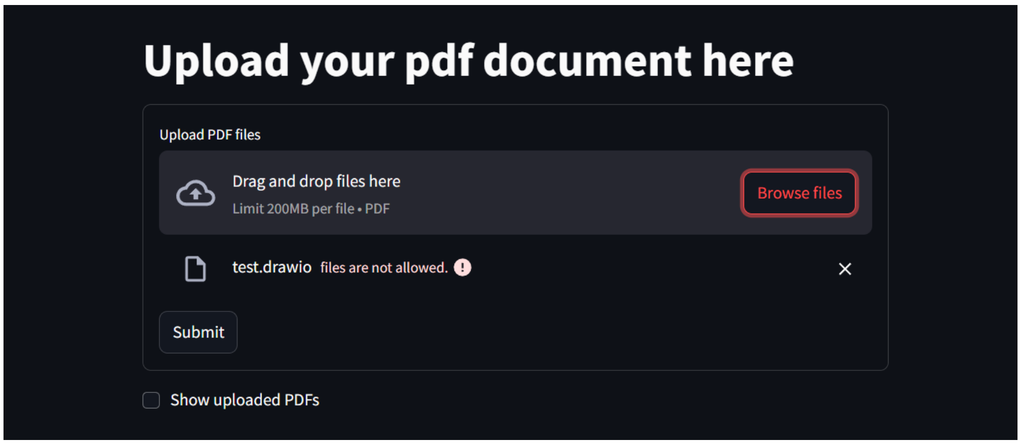
Submit

Show uploaded PDFs

Files in uploaded:

	files uploaded
0	Testcw.pdf
1	TestFinancialPdf.pdf
2	TestPdf.pdf
3	TestPdf2.pdf
4	TestPdfSQA.pdf
5	TestScannedPdf.pdf
6	TestShortPdf.pdf
7	Testws.pdf
8	Topic_6 (1).pdf

Figure 4.14: The file upload interface



Upload your pdf document here

Upload PDF files

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

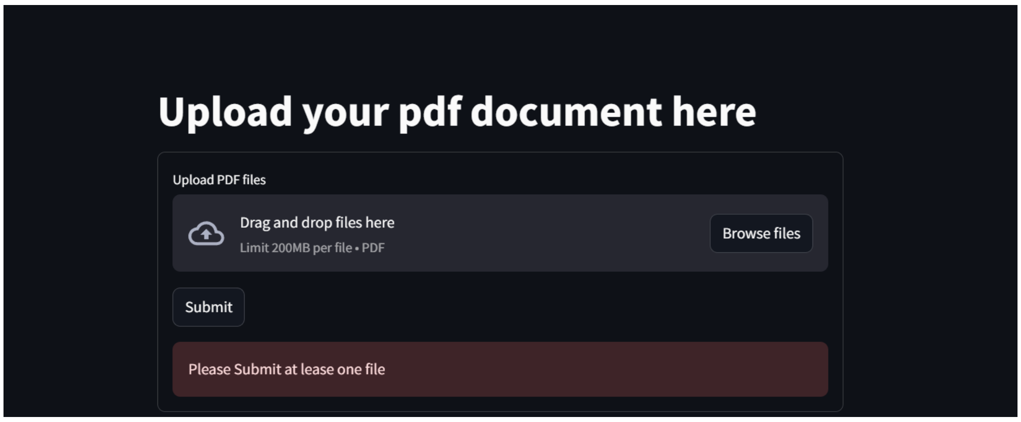
Browse files

test.drawio files are not allowed.

Submit

Show uploaded PDFs

Figure 4.15: The file upload interface with error



Upload your pdf document here

Upload PDF files

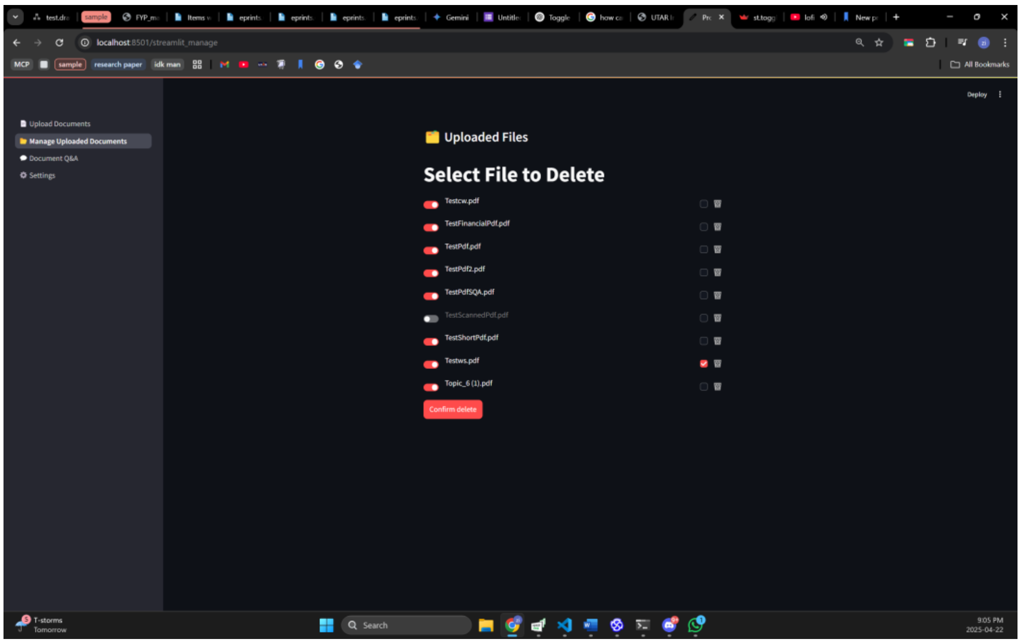
Drag and drop files here
Limit 200MB per file • PDF

Browse files

Submit

Please Submit at lease one file

Figure 4.16: The file upload interface with error

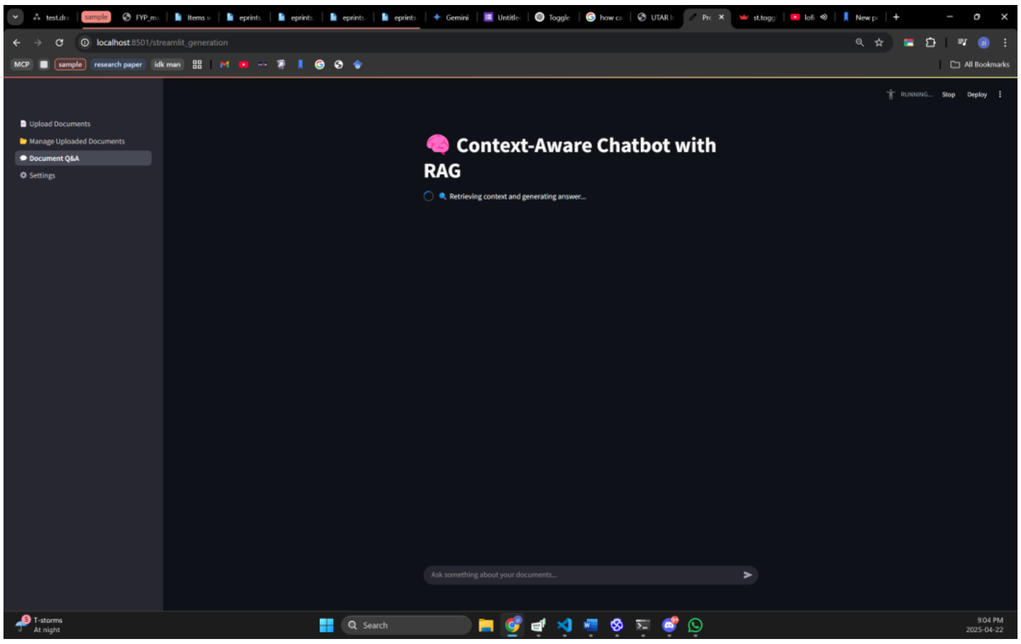


Uploaded Files

Select File to Delete

Testcw.pdf
TestFinancialPdf.pdf
TestPdf.pdf
TestPdf2.pdf
TestPdfSQA.pdf
TestScannedPdf.pdf
TestShortPdf.pdf
Testws.pdf
Topic_6 (1).pdf

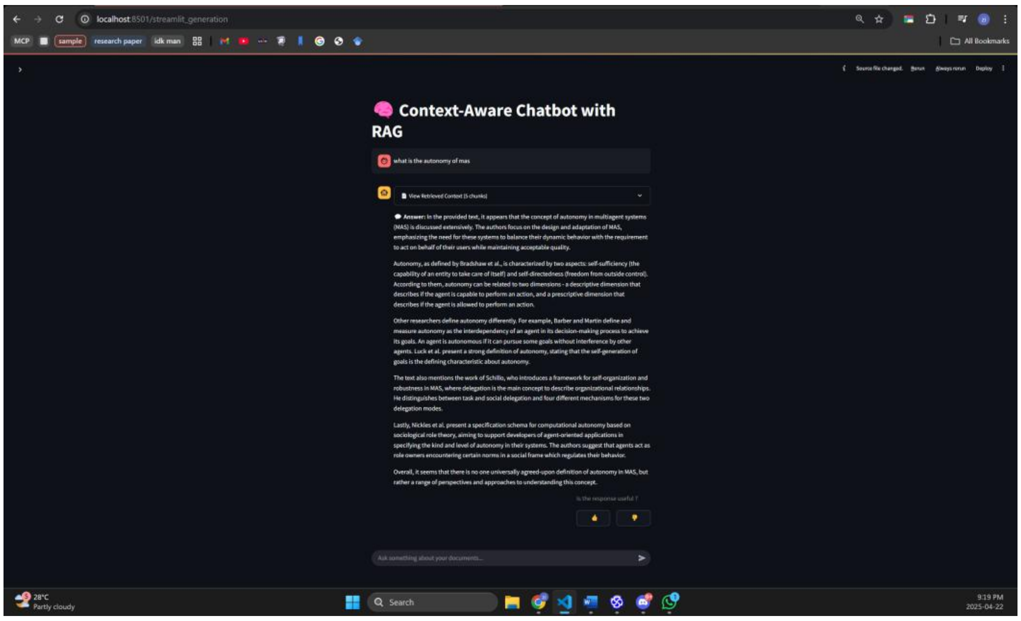
Figure 4.17: The file management interface


Upload Documents
Manage Uploaded Documents
Document Q&A
Settings
RUNNING...
Stop
Deploy
Ask something about your documents...

Context-Aware Chatbot with RAG

Retrieving context and generating answer...

Figure 4.18: The document chat interface



Context-Aware Chatbot with RAG

what is the autonomy of mas

View Retrieved Context (5 chunks)

Answer: In the provided text, it appears that the concept of autonomy in multiagent systems (MAS) is discussed extensively. The authors focus on the design and adaptation of MAS, emphasizing the need for these systems to balance their dynamic behavior with the requirement to act on behalf of their users while maintaining acceptable quality.

Autonomy, as defined by Bradshaw et al., is characterized by two aspects: self-sufficiency (the capability of an entity to take care of itself) and self-directedness (freedom from outside control). According to them, autonomy can be related to two dimensions - a descriptive dimension that describes if the agent is capable to perform an action, and a prescriptive dimension that describes if the agent is allowed to perform an action.

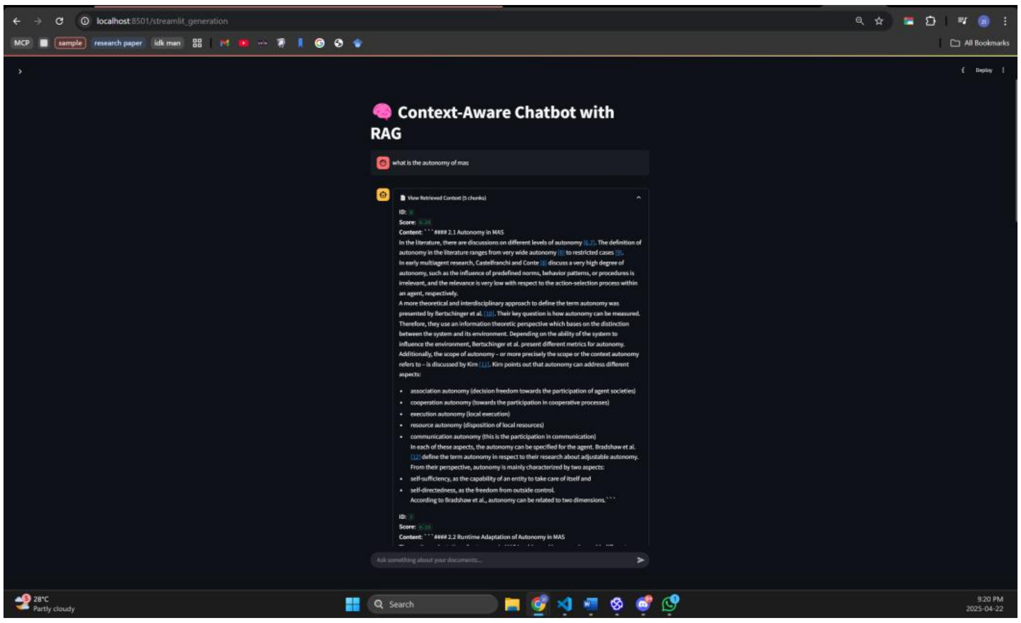
Other researchers define autonomy differently. For example, Barber and Martin define and measure autonomy as the interdependency of an agent in its decision-making process to achieve its goals. An agent is autonomous if it can pursue some goals without interference by other agents. Luck et al. present a strong definition of autonomy, stating that the self-generation of goals is the defining characteristic about autonomy.

The text also mentions the work of Schillo, who introduces a framework for self-organization and robustness in MAS, where delegation is the main concept to describe organizational relationships. He distinguishes between task and social delegation and four different mechanisms for these two delegation modes.

Lastly, Nickles et al. present a specification schema for computational autonomy based on sociological role theory, aiming to support developers of agent-oriented applications in specifying the kind and level of autonomy in their systems. The authors suggest that agents act as role owners encountering certain norms in a social frame which regulates their behavior.

Overall, it seems that there is no one universally agreed-upon definition of autonomy in MAS, but rather a range of perspectives and approaches to understanding this concept.

Figure 4.19: The document chat interface (response)



Context-Aware Chatbot with RAG

what is the autonomy of mas

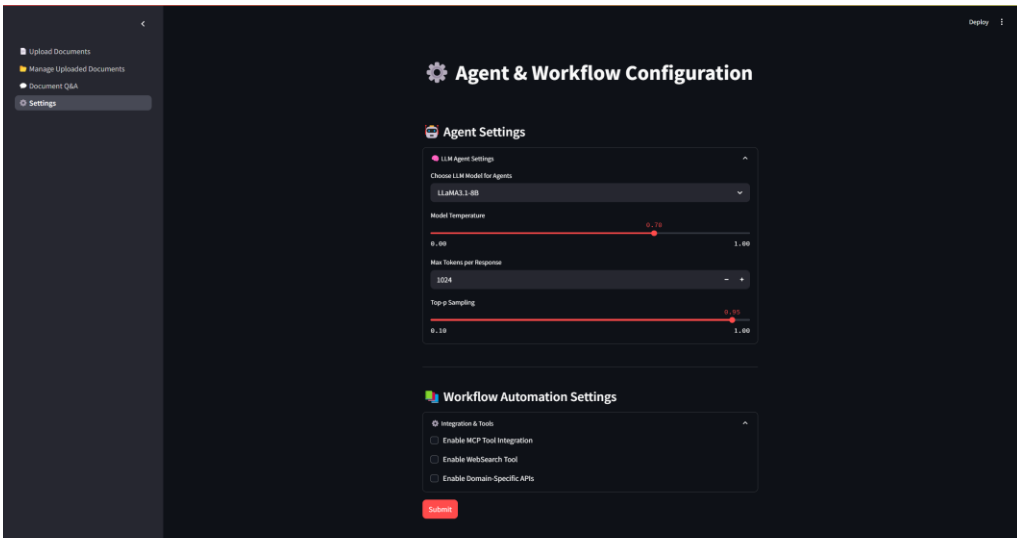
View Retrieved Context (5 chunks)

ID: [illegible]
Score: [illegible]
Content: ```#### 2.1 Autonomy in MAS
In the literature, there are discussions on different levels of autonomy [6,7]. The definition of autonomy in the literature ranges from very wide autonomy [8] to restricted cases [9].
In early multiagent research, Castelfranchi and Conte [3] discuss a very high degree of autonomy, such as the influence of predefined norms, behavior patterns, or procedures is irrelevant, and the relevance is very low with respect to the action-selection process within an agent, respectively.
A more theoretical and interdisciplinary approach to define the term autonomy was presented by Bertschinger et al. [10]. Their key question is how autonomy can be measured. Therefore, they use an information theoretic perspective which bases on the distinction between the system and its environment. Depending on the ability of the system to influence the environment, Bertschinger et al. present different metrics for autonomy. Additionally, the scope of autonomy – or more precisely the scope or the context autonomy refers to – is discussed by Kim [11]. Kim points out that autonomy can address different aspects:

- association autonomy (decision freedom towards the participation of agent societies)
- cooperation autonomy (towards the participation in cooperative processes)
- execution autonomy (local execution)
- resource autonomy (disposition of local resources)
- communication autonomy (this is the participation in communication)
  In each of these aspects, the autonomy can be specified for the agent. Bradshaw et al. [12] define the term autonomy in respect to their research about adjustable autonomy. From their perspective, autonomy is mainly characterized by two aspects:
- self-sufficiency, as the capability of an entity to take care of itself and
- self-directedness, as the freedom from outside control.
  According to Bradshaw et al., autonomy can be related to two dimensions.```

ID: [illegible]
Score: [illegible]
Content: ```#### 2.2 Runtime Adaptation of Autonomy in MAS

Figure 4.20: The document chat interface (relevant chunks)



Agent & Workflow Configuration

Agent Settings

LLM Agent Settings

Choose LLM Model for Agents: LLaMA3.1-8B

Model Temperature: 0.70 (0.00 – 1.00)

Max Tokens per Response: 1024

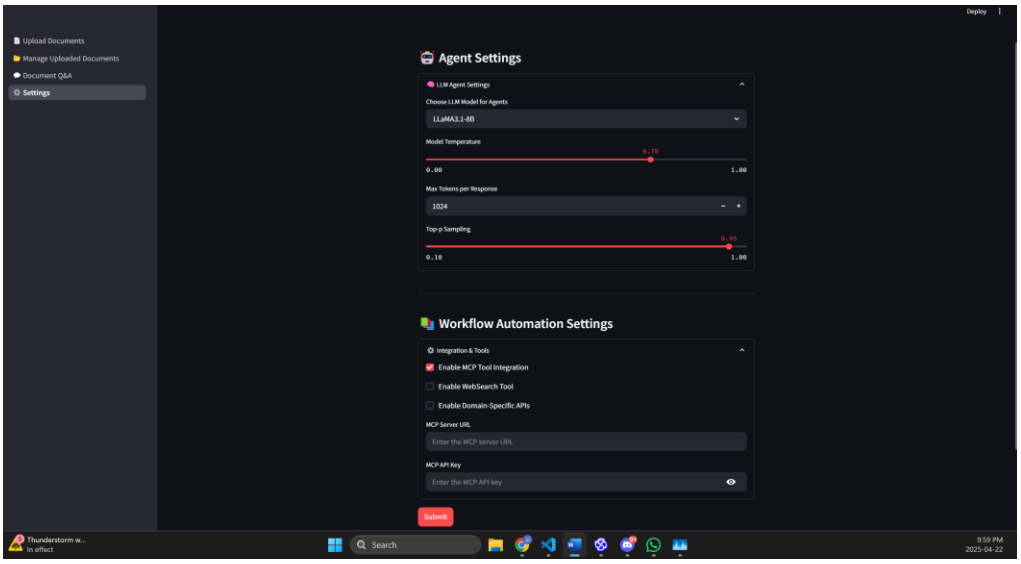
Top-p Sampling: 0.95 (0.10 – 1.00)

Workflow Automation Settings

Integration & Tools

- [ ] Enable MCP Tool Integration
- [ ] Enable WebSearch Tool
- [ ] Enable Domain-Specific APIs

Figure 4.21: The system setting interface



Agent Settings

LLM Agent Settings

Choose LLM Model for Agents: LLaMA3.1-8B

Model Temperature: 0.70 (0.00 – 1.00)

Max Tokens per Response: 1024

Top-p Sampling: 0.95 (0.10 – 1.00)

Workflow Automation Settings

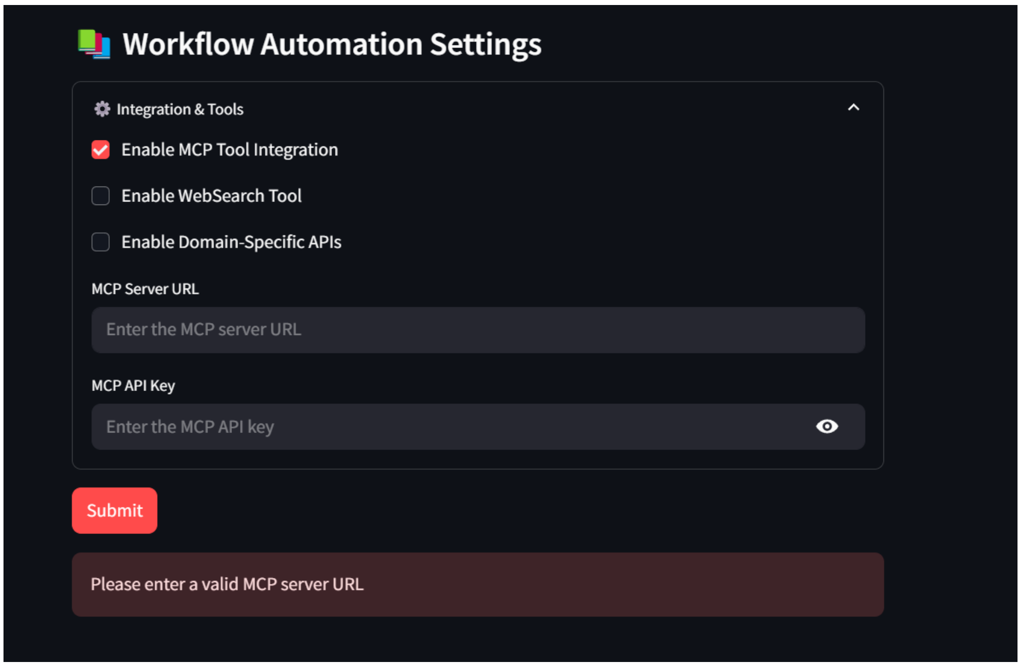
Integration & Tools

- [x] Enable MCP Tool Integration
- [ ] Enable WebSearch Tool
- [ ] Enable Domain-Specific APIs

MCP Server URL: Enter the MCP server URL

MCP API Key: Enter the MCP API key

Figure 4.22:The system setting interface (toggle MCP services)



Workflow Automation Settings

Integration & Tools

- [x] Enable MCP Tool Integration
- [ ] Enable WebSearch Tool
- [ ] Enable Domain-Specific APIs

MCP Server URL

Enter the MCP server URL

MCP API Key

Enter the MCP API key

Please enter a valid MCP server URL

Figure 4.23: Error message when invalid MCP server detail input

CHAPTER 5

SYSTEM DESIGN

5.1 Introduction

This section will discuss the overall system design, including system flow from user query to the backend database and how the user query is processed and transported to each node. Moreover, this section will also discuss different technical components and their roles in supporting the workflow, as well as the database schema, to provide a comprehensive overview of the system architecture.

5.2 System flow

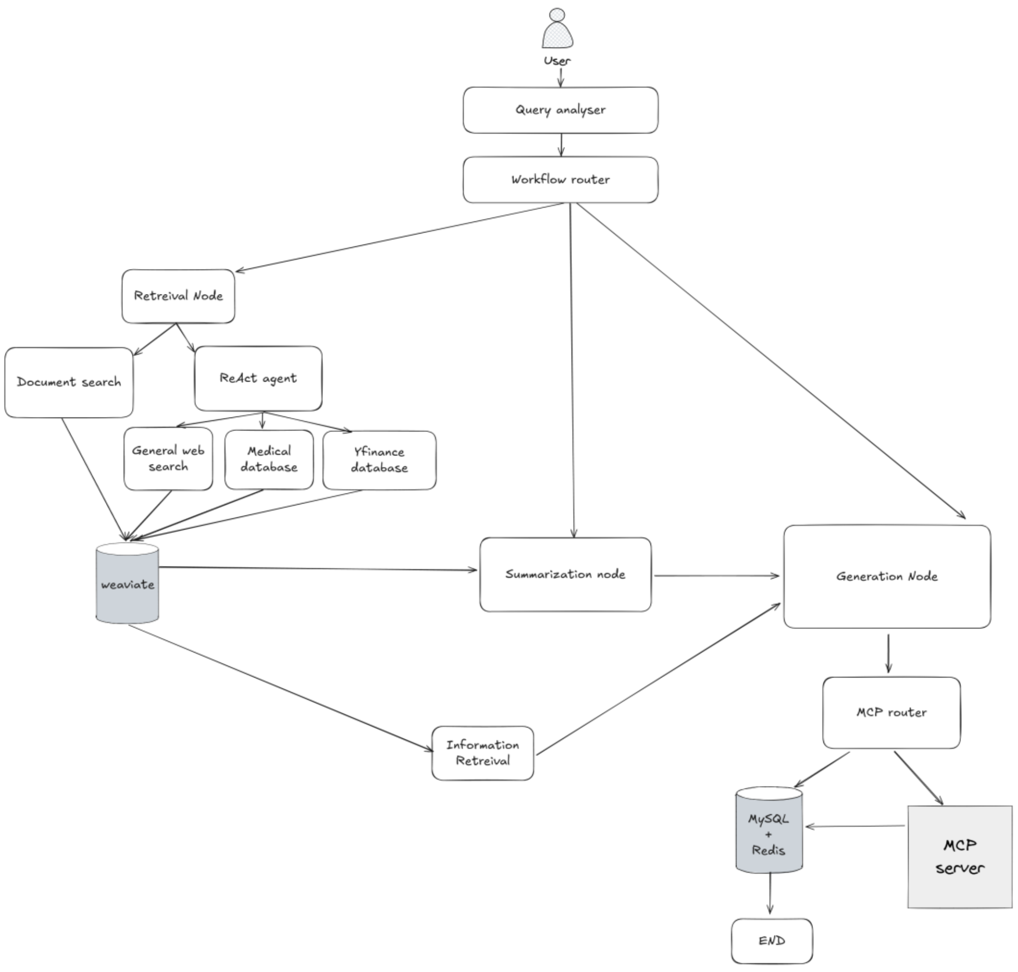


Figure 5.1: System flow

The system flow diagram illustrates the backend flow of the system, beginning with the user sending a request to the backend. The first component that processes the user request is the query analyser. The query analyser will analyse if the query is a follow-up question, if the query is a follow-up question or contains some ambiguous wording like “that”, “last-message”, or “it” that is referring to the last message, the query will be rewritten based on the message history to be more specific and searchable. For example, for a query like “what does that mean,” the query analyser will transform the query into “more detail about machine learning” if the history is related to a machine learning topic. Next, the query analyser will pass the process query to the workflow router, which will identify the most suitable task for the user query for a request that can be simply answered by the LLM's knowledge. The system will route the request to the generation node to generate a response, skipping the retrieval part for simple requests. For the document summarization request, the router will pass the request to the summarization node, which will detect the relevant document inside the database to extract the 5 most relevant chunks that can represent the document the most, then summarize the extracted chunks before passing the summary of the chunks to the generation node for the final explanation. For request that needs the latest information to support or need support information from the uploaded document that cannot be answered accurately with the knowledge base of LLM, the router will route the request to the retrieval node, which will retrieve relevant information from various sources, such as documents and external sources. Then, the relevant information will be passed into the generation node to achieve the RAG core functionality and to ensure the response is based on grounded information.

For the external search, the system employs a ReAct agent to dynamically route the retrieval request to the most suitable external search path that is most likely to provide relevant information. Three external search sources will be wrapped into tools that will pass into the ReAct agent for dynamic tool selection, including a general web search that will pass the query to the web search engine to scrape the top 5 most relevant web pages’ content, then perform preprocessing and hybrid retrieval to get the most relevant information from the general web search engine. Next, the external search will connect to a medical database source, where it will scrape the article content from the PubMed database for requests related to medical knowledge. Lastly, the external search will connect to yfinance new website, which will allow the system to fetch the latest finance or business news from Yahoo Finance page. This approach can ensure the system not only can retrieve relevant content from user-uploaded documents but also allows the system to fill the knowledge gap with the latest information from different internet sources.

After generating the response, the system will route the request query to the MCP, which the system will check if the user query contains any request about document processing, such as "create a document about machine learning" and "add content into the document machine learning.docx". If the intent is detected, the system will start to execute the document process operation by creating a ReAct agent with all of the tools provided by the MCP server. Then the system will start to execute all of the needed tools to fulfil the user document processing requirement with the support of the generated response from the generation node.

5.3 Operation mode

This system offers two operation modes for two different workflows, which are personal and organizational workflows. First, the personal workflow will allow the user to upload and manage their document as well as change the system configuration that will only affect the system configuration of that particular user. The document storage for personal mode is fully separate among different users so that users are unable to retrieve information from documents that is not uploaded by themselves. This operation mode aims to maximize the flexibility of the system for individual users who just wish to use this system in a standalone machine.

For the organization mode, the user will be separated into two roles, which are admin and user. In this mode, only the admin will be allowed to upload, manage the document in the database, and change the system configuration used in the current flow. The uploaded document will be a shared document, which every user who has the role of user will be allowed to retrieve. Admin also has a function to manage users via CRUD operation, which will be the only way for the system in organization mode to create new users. As for the user, they are only allowed to do two things, which are to view the document currently available in the database and perform document Q&A via the main chat interface that will perform the system flow. This approach is aimed to suit the server-client workflow where the backend server will act as a centralized knowledge base that will store all of the necessary files, allowing the system to essentially become an organization's document enquiry that can reduce the time taken for information searching and document processing.

5.4 System Architecture Components

5.4.1 FastAPI

FastAPI is a modern, high-performance web framework for building APIs with Python. It is designed with speed and scalability, which FastAPI is built to handle large numbers of requests effectively. In this system, FastAPI will be the backbone of this project, which it will be used to create api endpoint for all of the backend functionality of the system, such as preprocessing, graph workflow and user authentication. FastAPI is also crucial in making the system responsive, as it provides WebSocket functionality that will be used for real-time processing feedback and also to create a "streaming" effect for the generation node responses, making the system more responsive to user input.

5.4.2 React

React is a widely used, open-source JavaScript library developed by Facebook, designed to build dynamic and interactive user interfaces. React provided a platform that allows developers to build a user interface with modular components that can be used to develop a complex frontend system. Additionally, React also supports a wide range of UI frameworks and libraries that allow for rapid prototyping for a user-friendly interface. In this project, React is used to develop the frontend component, which is the main entry point for the user to interact with the system, ensuring a seamless and interactive experience for the user.

5.4.3 Weaviate

Weaviate is an open-source vector database that is designed for the storage of a high volume of vector embedding. Weaviate provides Docker deployment options that can be accessed without needing to be online. Additionally, the embedding can also be stored along with extra metadata, which can allow filtering operations. In this system, Weaviate will be used to store the vector embedding data generated from the nomic embedding model that converts text chunks into vector representations. Moreover, Weaviate also provides a built-in hybrid retrieval method that uses BM25 for keyword-based search and a dense search algorithm, which will be fed into a fusion algorithm that will calculate the relevancy and combine results from both search algorithms, making the retrieval component more flexible for different types of queries.

5.4.4 MySQL

MySQL is a lightweight, open-source relational database management system (RDBMS) that is widely adopted for web applications and enterprise systems due to its stability, speed, and reliability. It offers a strong, quick, and dependable method for handling significant amounts of data in an organized way. In this system, a MySQL database will be used as the secondary database, while the vector database is specialized for storing and retrieving high-dimensional text embeddings. MySQL is responsible for managing the broader system data that is essential for application functionality. For example, MySQL will also be used to store the user information for the authentication module, document information that is uploaded by the user, and the chat history for the user for persistent usage. Thus, MySQL's robust database features and ability to scale well make it a perfect choice as the secondary database in this project

5.4.5 Redis

Redis is an open source, in-memory cache system that is popular because of its reliability and lightning-fast data operation speed, while having the ability to scale to millions of messages, making it a very popular choice among different types of systems, such as web applications and speed-sensitive applications. In this system, Redis will be used to store two data, which are user authentication tokens and chat history. For user authentication, this can ensure the user token can be fetched with minimum delay, with a given time-to-live that will eliminate the need for manual cleanup. For the chat history, this can ensure chat history can be fetched very quickly to pass into system operation to let the generation module refer to previous history, ensuring the generation module can get all the context it needs. Thus, Redis will be used in this system as the third database to store data that is time sensitive.

5.5 Database schema design

5.5.1 MySQL database schema

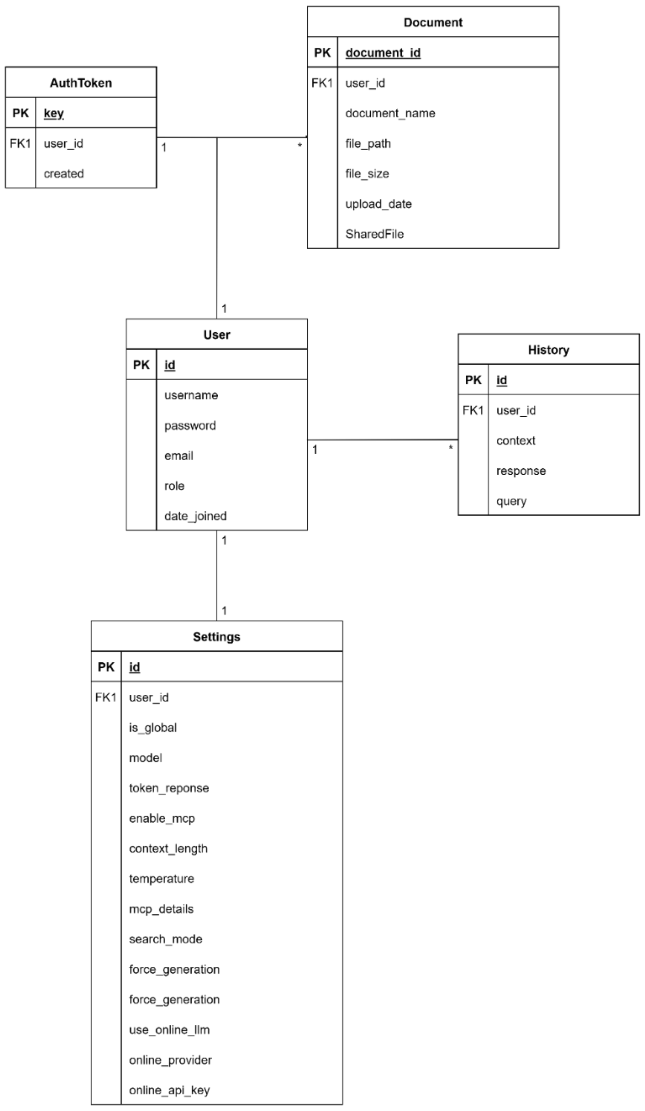


Figure 5.2: MySQL Entity Relationship Diagram

Table 5.1: MySQL database schema

Table	Description
User	This table is used to store all user information, including username and password, that will be used for authentication, and the role that will be utilized in access control
AuthToken	This table is used to store a unique JWT-based token based on different users and sessions to increase the security of the system
Settings	This table is used to store users' settings based on different user IDs, including the model used, the maximum token for response, and enable MCP component. If the deployment mode is set to organization, the only user setting that will be used is based on the value of is_global, which will be used if the value is True
History	This table is used to store users' chat history, including the context, generated context and query.
Document	This table is used to store all of the information about the document store uploaded and the related user for the document

5.5.2 Weaviate database schema

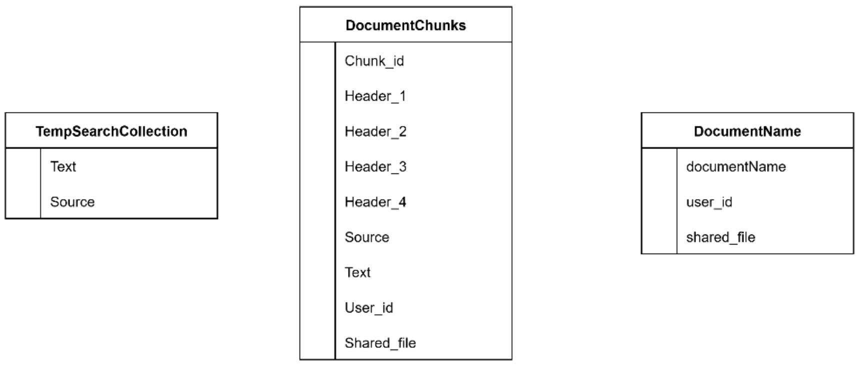
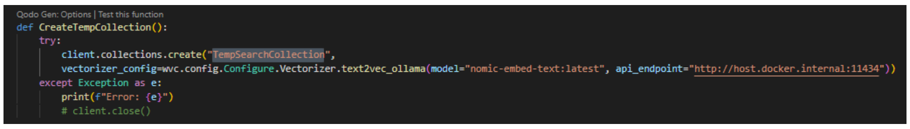
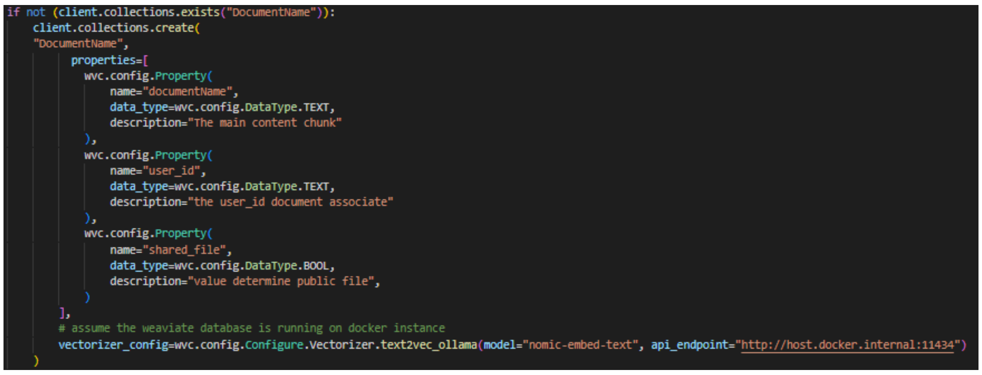


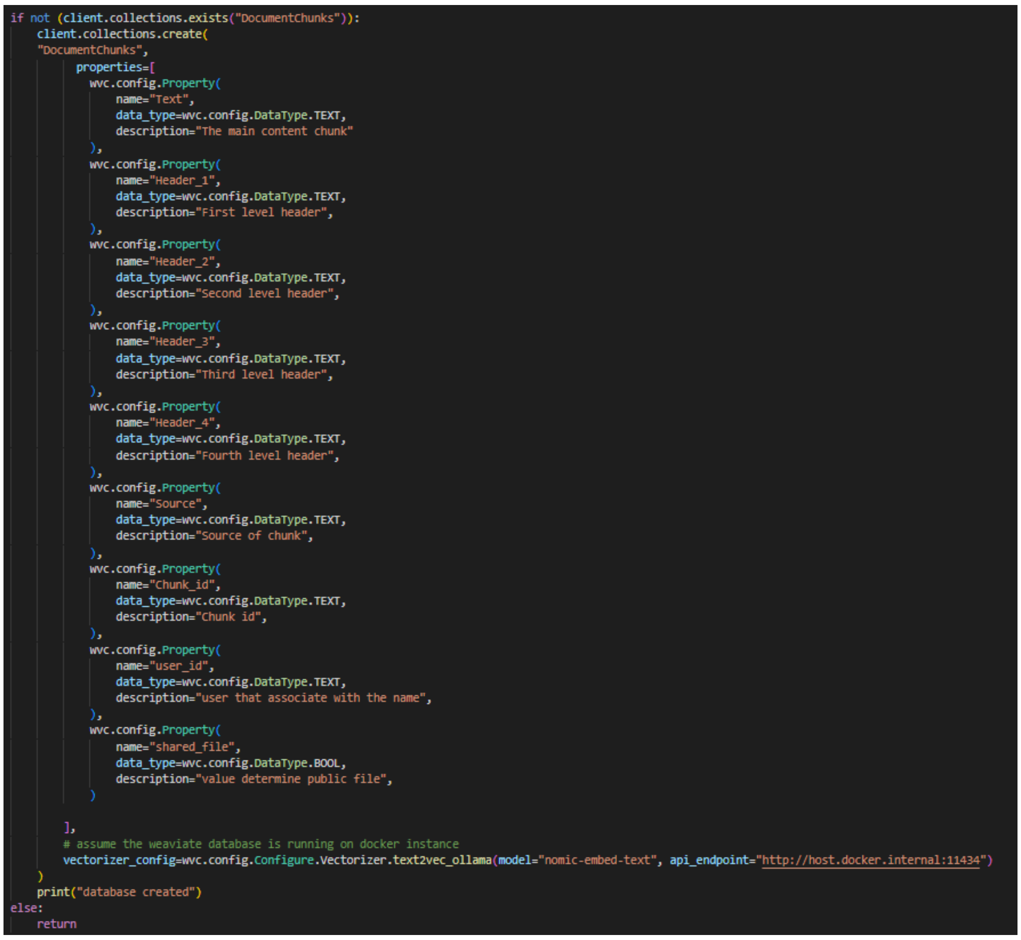
Figure 5.3: Weaviate Entity Relationship Diagram



```
Qodo Gen: Options | Test this function
def CreateTempCollection():
    try:
        client.collections.create("TempSearchCollection",
        vectorizer_config=wvc.config.Configure.Vectorizer.text2vec_ollama(model="nomic-embed-text:latest", api_endpoint="http://host.docker.internal:11434"))
    except Exception as e:
        print(f"Error: {e}")
        # client.close()
```



```
if not (client.collections.exists("DocumentName")):
    client.collections.create(
    "DocumentName",
          properties=[
            wvc.config.Property(
                name="documentName",
                data_type=wvc.config.DataType.TEXT,
                description="The main content chunk"
            ),
            wvc.config.Property(
                name="user_id",
                data_type=wvc.config.DataType.TEXT,
                description="the user_id document associate"
            ),
            wvc.config.Property(
                name="shared_file",
                data_type=wvc.config.DataType.BOOL,
                description="value determine public file",
            )
        ],
        # assume the weaviate database is running on docker instance
        vectorizer_config=wvc.config.Configure.Vectorizer.text2vec_ollama(model="nomic-embed-text", api_endpoint="http://host.docker.internal:11434")
    )
```



```
if not (client.collections.exists("DocumentChunks")):
    client.collections.create(
    "DocumentChunks",
          properties=[
            wvc.config.Property(
                name="Text",
                data_type=wvc.config.DataType.TEXT,
                description="The main content chunk"
            ),
            wvc.config.Property(
                name="Header_1",
                data_type=wvc.config.DataType.TEXT,
                description="First level header",
            ),
            wvc.config.Property(
                name="Header_2",
                data_type=wvc.config.DataType.TEXT,
                description="Second level header",
            ),
            wvc.config.Property(
                name="Header_3",
                data_type=wvc.config.DataType.TEXT,
                description="Third level header",
            ),
            wvc.config.Property(
                name="Header_4",
                data_type=wvc.config.DataType.TEXT,
                description="Fourth level header",
            ),
            wvc.config.Property(
                name="Source",
                data_type=wvc.config.DataType.TEXT,
                description="Source of chunk",
            ),
            wvc.config.Property(
                name="Chunk_id",
                data_type=wvc.config.DataType.TEXT,
                description="Chunk id",
            ),
            wvc.config.Property(
                name="user_id",
                data_type=wvc.config.DataType.TEXT,
                description="user that associate with the name",
            ),
            wvc.config.Property(
                name="shared_file",
                data_type=wvc.config.DataType.BOOL,
                description="value determine public file",
            )

        ],
        # assume the weaviate database is running on docker instance
        vectorizer_config=wvc.config.Configure.Vectorizer.text2vec_ollama(model="nomic-embed-text", api_endpoint="http://host.docker.internal:11434")
    )
    print("database created")
else:
    return
```

Figure 5.4: Weaviate database scripts

Table 5.2: Weaviate collection schema

Collection	Description
DocumentChunks	This collection is used to store all of the chunks from the user-uploaded document, with some

	additional information that will be used in the retrieval module, such as header information and source of the chunks
DocumentName	This collection is used to store all of the documents uploaded to the vector database
TempSearchCollection	This collection is used by all external search tools, including GeneralWebSearch, MedicalDatabase, and FinanceNews modules. This collection will be initialized every time the external search is invoked and will be deleted after the processing is finished.

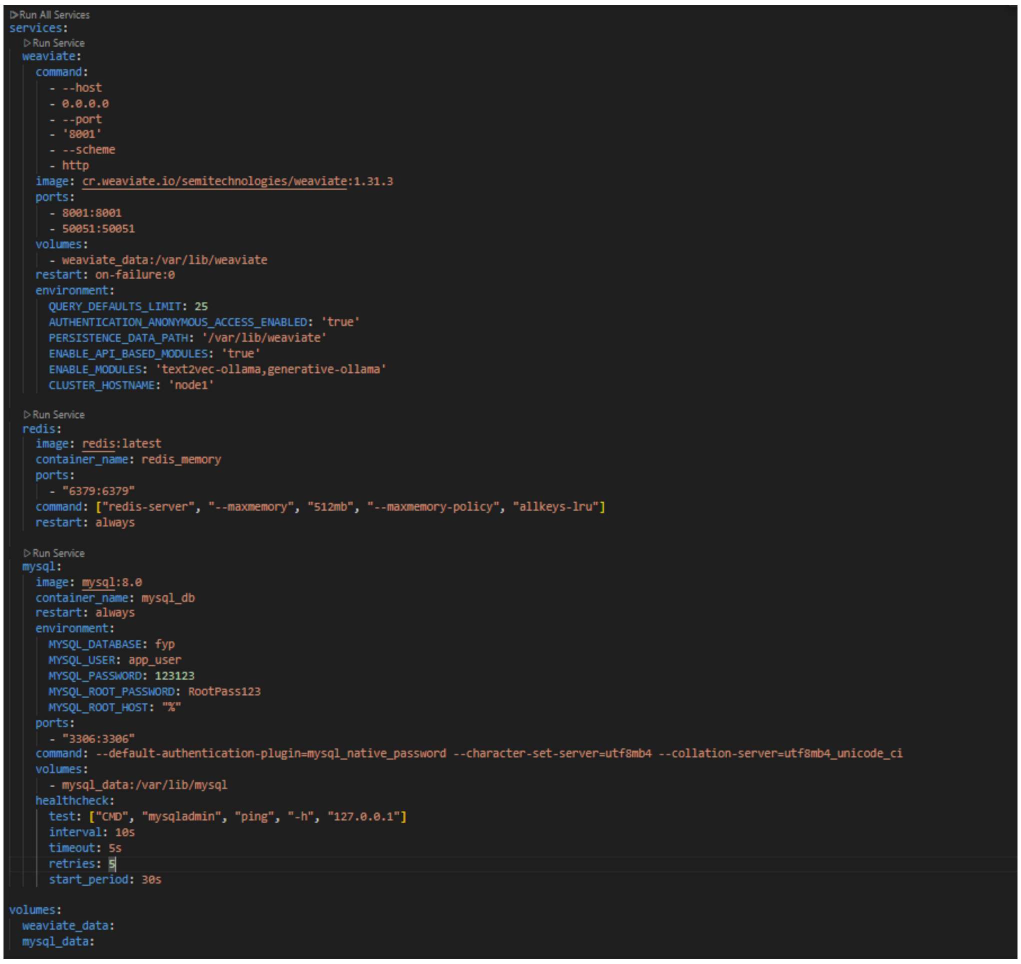
CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 Introduction

In this section, a snippet of the actual working product will be provided and explain which a detailed description of all system components, such as user authentication, file management, and chat function. setting the interface and user management interface to provide a clear view of how the system operates as a working product.

6.2 Software setup



```
▷Run All Services
services:
  ▷Run Service
  weaviate:
    command:
      - --host
      - 0.0.0.0
      - --port
      - '8001'
      - --scheme
      - http
    image: cr.weaviate.io/semitechnologies/weaviate:1.31.3
    ports:
      - 8001:8001
      - 50051:50051
    volumes:
      - weaviate_data:/var/lib/weaviate
    restart: on-failure:0
    environment:
      QUERY_DEFAULTS_LIMIT: 25
      AUTHENTICATION_ANONYMOUS_ACCESS_ENABLED: 'true'
      PERSISTENCE_DATA_PATH: '/var/lib/weaviate'
      ENABLE_API_BASED_MODULES: 'true'
      ENABLE_MODULES: 'text2vec-ollama,generative-ollama'
      CLUSTER_HOSTNAME: 'node1'

  ▷Run Service
  redis:
    image: redis:latest
    container_name: redis_memory
    ports:
      - "6379:6379"
    command: ["redis-server", "--maxmemory", "512mb", "--maxmemory-policy", "allkeys-lru"]
    restart: always

  ▷Run Service
  mysql:
    image: mysql:8.0
    container_name: mysql_db
    restart: always
    environment:
      MYSQL_DATABASE: fyp
      MYSQL_USER: app_user
      MYSQL_PASSWORD: 123123
      MYSQL_ROOT_PASSWORD: RootPass123
      MYSQL_ROOT_HOST: "%"
    ports:
      - "3306:3306"
    command: --default-authentication-plugin=mysql_native_password --character-set-server=utf8mb4 --collation-server=utf8mb4_unicode_ci
    volumes:
      - mysql_data:/var/lib/mysql
    healthcheck:
      test: ["CMD", "mysqladmin", "ping", "-h", "127.0.0.1"]
      interval: 10s
      timeout: 5s
      retries: 5
      start_period: 30s

volumes:
  weaviate_data:
  mysql_data:
```

Figure 6.1: docker-compoose scripts

The system requires three external dependencies for the databases used in this system, including Redis, MySQL, and Weaviate. To utilize these services, this system uses docker compose, which simplifies deployment and ensures consistency across development and production environments. A docker-compose file is created with the scripts above that specify the service's port, environment variable and the image version used.

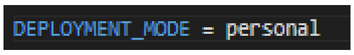
The Weaviate is deployed as a vector database service using the cr.weaviate.io/semitechnologies/weaviate:1.31.3 image, which is the official Docker image release by the development team of Weaviate. It is configured to listen on port 8001 for API requests and port 50051 for gRPC communication. The setup enables API-based modules used in this system to fetch and write into the Weaviate database.

The Redis service is built with the latest official Redis Docker image that will be deployed on port 6379 and run with specify command that will set the memory settings use for the Redis service

Lastly, MySQL is deployed using the mysql:8.0 image that is configured to listen on port 3306 along with standard application user (app_user) and password authentication. Additionally, the character encoding is set to utf8mb4 with collation utf8mb4_unicode_ci to support multilingual data processing.

6.3 Log in & Register module

This system uses a standard access control where every user needs to register an account to prevent any unauthorized access to the document chat information. The only difference is that this system has two types of deployment mode, which is personal and organization deployment modes. Where the personal workflow is for individual who deploy this system on their personal computer, and the organization deployment mode is for client-server deployments



```
DEPLOYMENT_MODE = personal
```

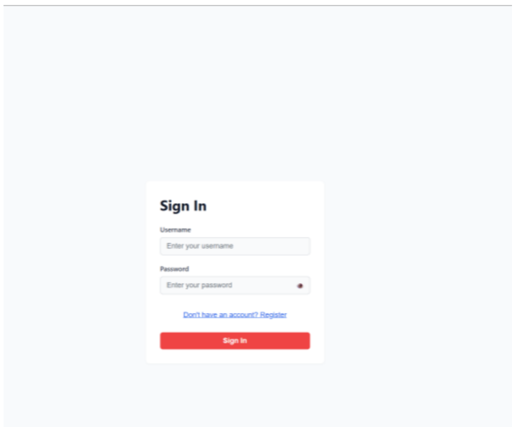


Figure 6.2: Sign in interface for personal deployment

The sign-in interface is the first interface users see when accessing the system. In this interface, user need to enter their username and the password to sign in to the actual system main interface. This interface also has a hyperlink that will take the user to the register page if their does not have an existing account.

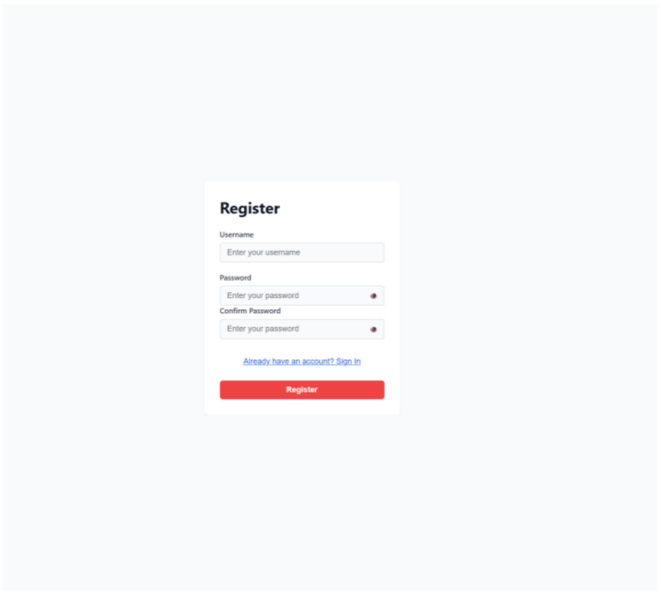
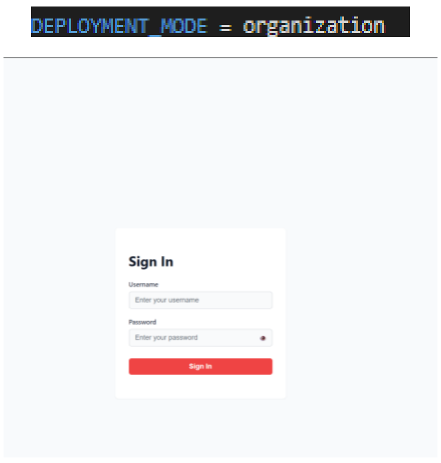


Figure 6.3: Register interface for personal deployment

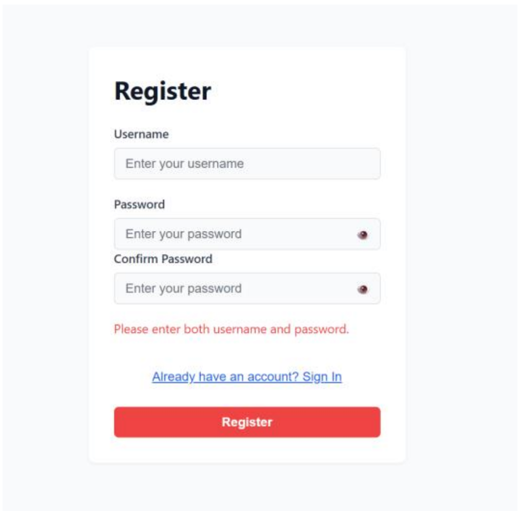
The register interface is shown when the user presses the hyperlink. In this interface, the user also needs to enter their username and password that will be stored as a new user instance in the database, but in this interface user is required to enter the confirm password field to prevent any password misinput



```
DEPLOYMENT_MODE = organization
```

Figure 6.4: Sign in interface for organization deployment

As for the organization deployment, the user is only given the option to sign in and not register because in this workflow, the system will have two roles, which are admin and user, where only the admin can create new user accounts. Moreover, the system will create a default admin user account that allows the admin user to access the system.



Please enter both username and password.

Figure 6.5: Error message

If any invalid credentials or format, such as an empty password or username, or an unmatched password, are entered into the system, the error message will be displayed under the input field in both the register and login interface

6.4 File upload & File management module

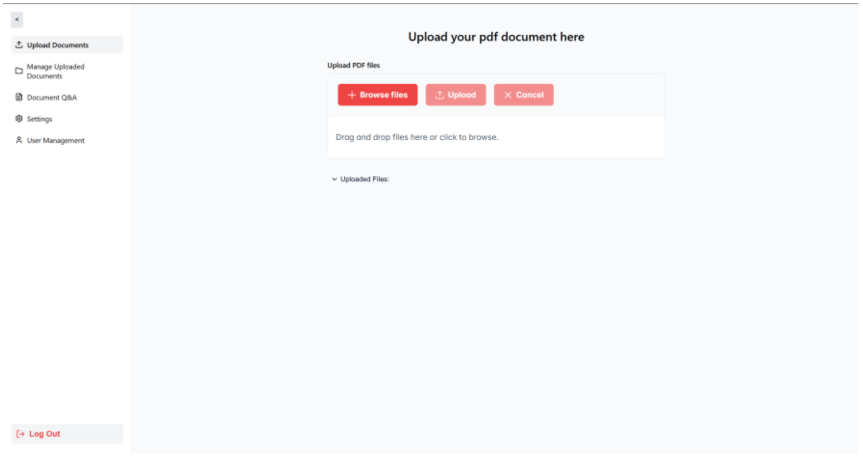
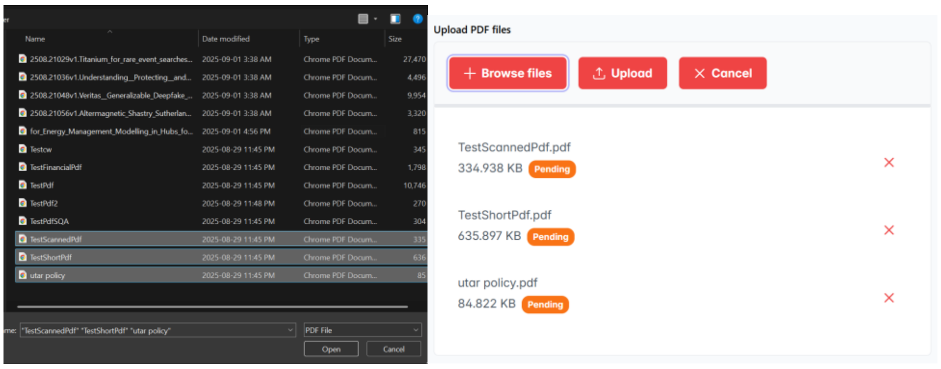


Figure 6.6: File upload interface

For every subsequent interface, a sidebar will be used to navigate to different interfaces and log back in to the sign-in interface. The File upload interface will be the first interface users see after signing in or registering, and users can upload the PDF document they want with the browse file component



Name	Date modified	Type	Size
2508.21029v1.Titanium_for_rare_event_searches...	2025-09-01 3:38 AM	Chrome PDF Docum...	27,470
2508.21036v1.Understanding__Protecting__and...	2025-09-01 3:38 AM	Chrome PDF Docum...	4,496
2508.21048v1.Veritas__Generalizable_Deepfake_...	2025-09-01 3:38 AM	Chrome PDF Docum...	9,954
2508.21056v1.Altermagnetic_Shastry_Sutherlan...	2025-09-01 3:38 AM	Chrome PDF Docum...	3,320
for_Energy_Management_Modelling_in_Hubs_fo...	2025-09-01 4:56 PM	Chrome PDF Docum...	815
Testcw	2025-08-29 11:45 PM	Chrome PDF Docum...	345
TestFinancialPdf	2025-08-29 11:45 PM	Chrome PDF Docum...	1,798
TestPdf	2025-08-29 11:45 PM	Chrome PDF Docum...	10,746
TestPdf2	2025-08-29 11:48 PM	Chrome PDF Docum...	270
TestPdfSQA	2025-08-29 11:45 PM	Chrome PDF Docum...	304
TestScannedPdf	2025-08-29 11:45 PM	Chrome PDF Docum...	335
TestShortPdf	2025-08-29 11:45 PM	Chrome PDF Docum...	636
utar policy	2025-08-29 11:45 PM	Chrome PDF Docum...	85

"TestScannedPdf" "TestShortPdf" "utar policy"
PDF File

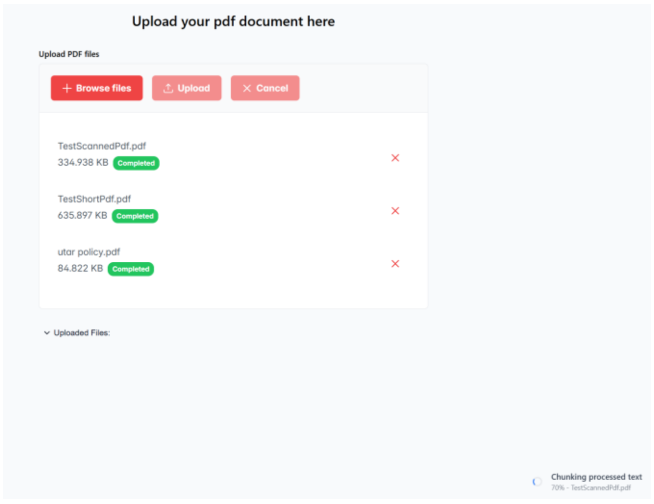
TestScannedPdf.pdf
334.938 KB Pending

TestShortPdf.pdf
635.897 KB Pending

utar policy.pdf
84.822 KB Pending

Figure 6.7: Select document to upload

After the user selects the document user, the file explorer will pop up, and the user is given the option to start or cancel the document upload process


Upload your pdf document here

Upload PDF files

TestScannedPdf.pdf
334.938 KB Completed

TestShortPdf.pdf
635.897 KB Completed

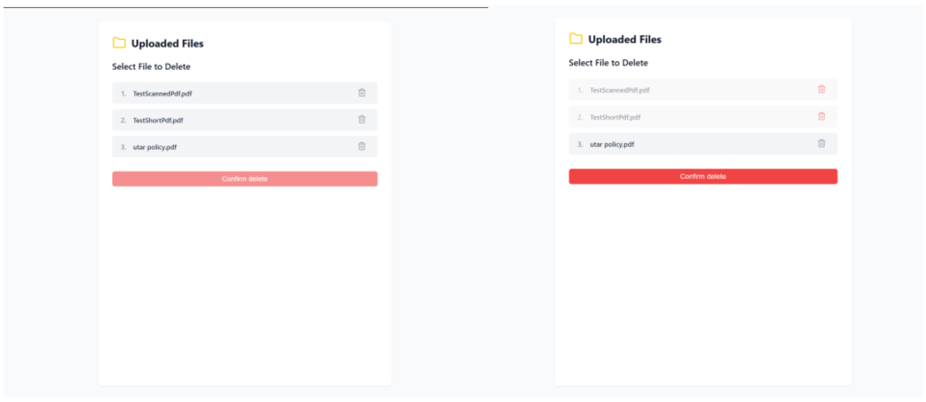
utar policy.pdf
84.822 KB Completed

Uploaded Files:

Chunking processed text
70% - TestScannedPdf.pdf

Figure 6.8: After the upload button is pressed

After the upload button is pressed, the system will initialize the upload pipeline. First, the system will store the uploaded file into the destination file system and the MySQL and Weaviate databases, and the interface will show upload completed if the upload is successful. Then, the system will start the preprocessing pipeline, where the document will be converted to a markdown file format, chunked into multiple pieces, and stored in the vector database to ensure the subsequent process can extract relevant information from the document. The preprocessing will be handled in the background, and the interface will show the progress of the document processing at the bottom right corner for a more responsive user experience.


Uploaded Files

Select File to Delete

1. TestScannedPdf.pdf
2. TestShortPdf.pdf
3. utar policy.pdf

Uploaded Files

Select File to Delete

1. TestScannedPdf.pdf
2. TestShortPdf.pdf
3. utar policy.pdf

Figure 6.9: File management interface

After the user uploads their document, all uploaded and processed files will show in the file management interface. In this interface, the user can only perform one operation: deleting the document. The user can check all the files they want to delete and press the "Confirm Delete" button to remove the document and all its content from the MySQL and Weaviate databases. For the organization workflow, the uploaded file is shared among all users with the role “user” or “admin” where “users” can fetch relevant information from the file uploaded by the admin.

6.5 Chat interface module

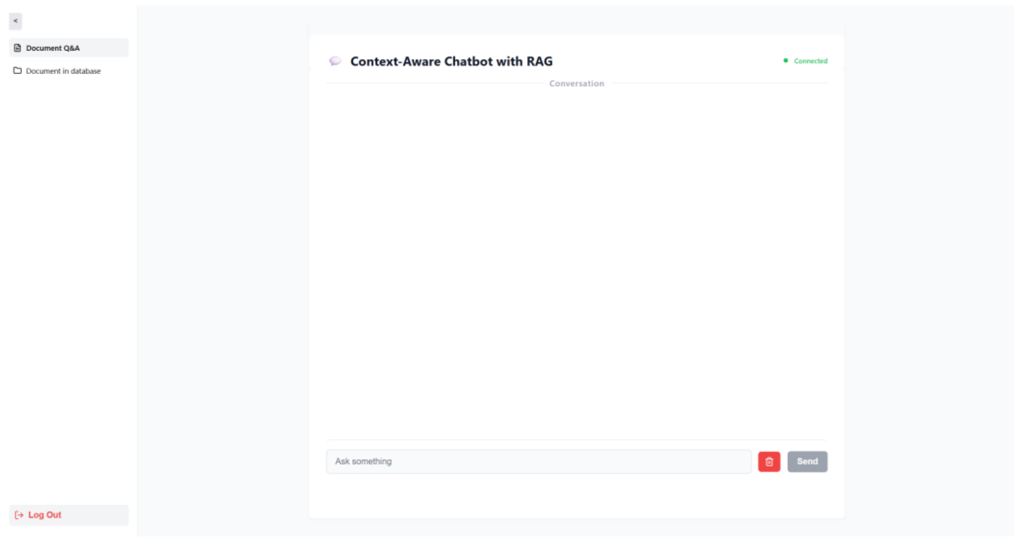


Figure 6.10: Chat interface for organization deployment

As for the account with the role user in organization deployment mode, the chat interface will be their first interface where they are only allowed to perform two operations in this system, including asking a question and viewing the available document in the database.

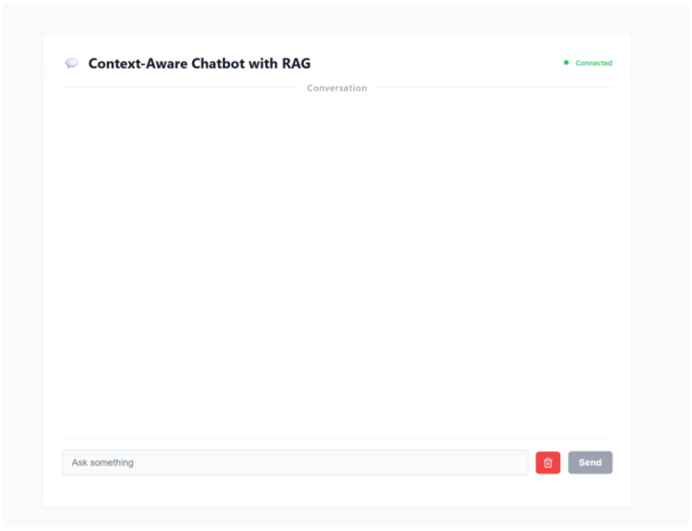
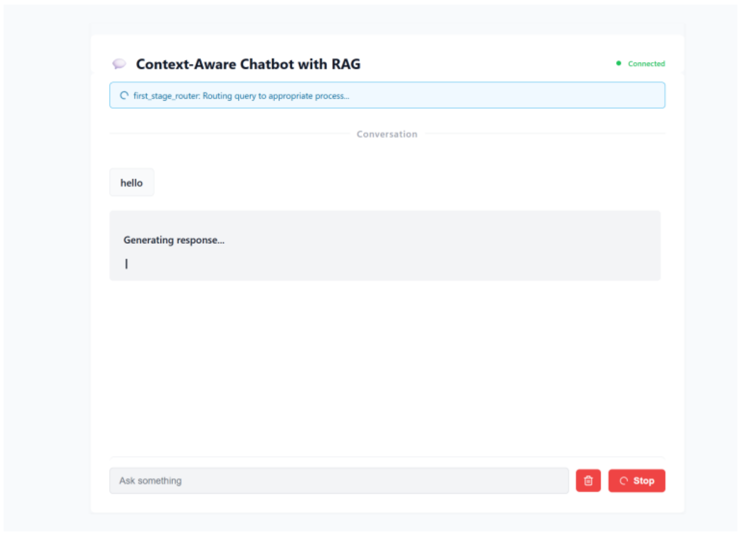


Figure 6.11: Chat interface

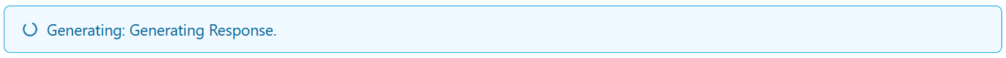
The Chat interface will be the main interaction point for the user and this system is designed to be as simple as possible for the user to work. In this interface, user can ask their questions, where the system will determine the best workflow to resolve the question with different processes such as retrieval, generation, and document automation, as shown in Figure 5.1.



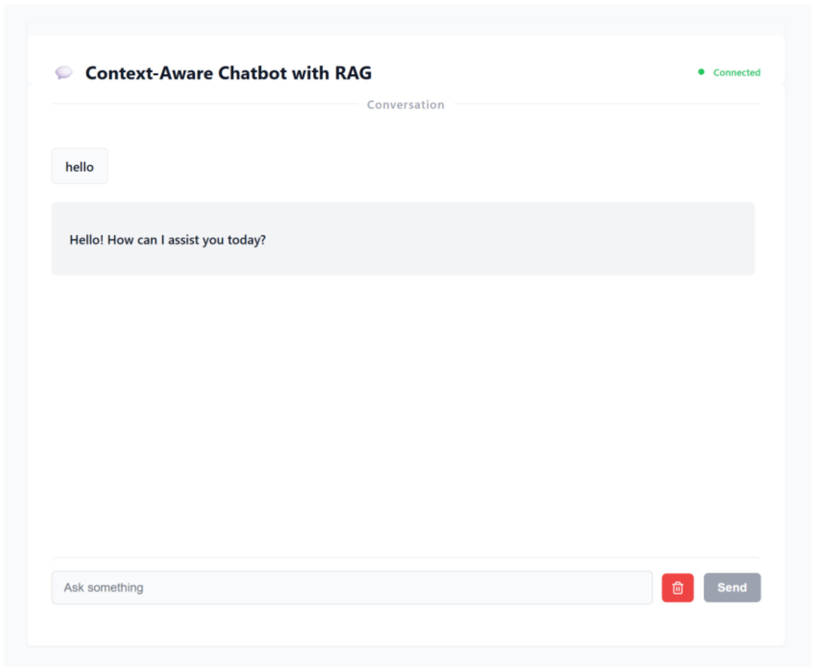
first_stage_router: Routing query to appropriate process...

hello

Generating response...



Generating: Generating Response.

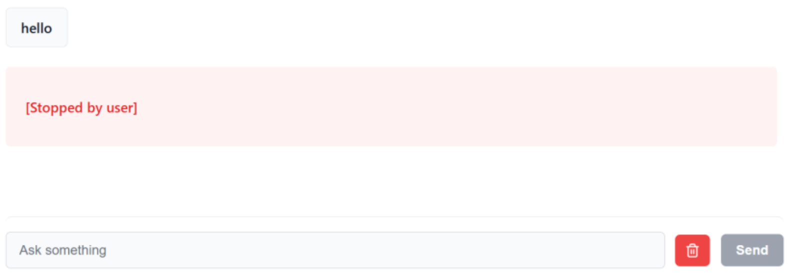


hello

Hello! How can I assist you today?

Figure 6.12: Processing user query

After the user enters the query into the system, the system will determine the most suitable workflow according to the user's query. For example, the system will route to the generation process for simple requests like a greeting, and the system will show the current progress stage at the top of the chat interface, such as the first stage router: routing the query to the appropriate workflow. After the process finishes, the system will stream the response token by token, similar to OpenAI ChatGPT. Therefore, the user does not need to wait after the full response is generated; they can see the output in real time, giving a more responsive user experience.

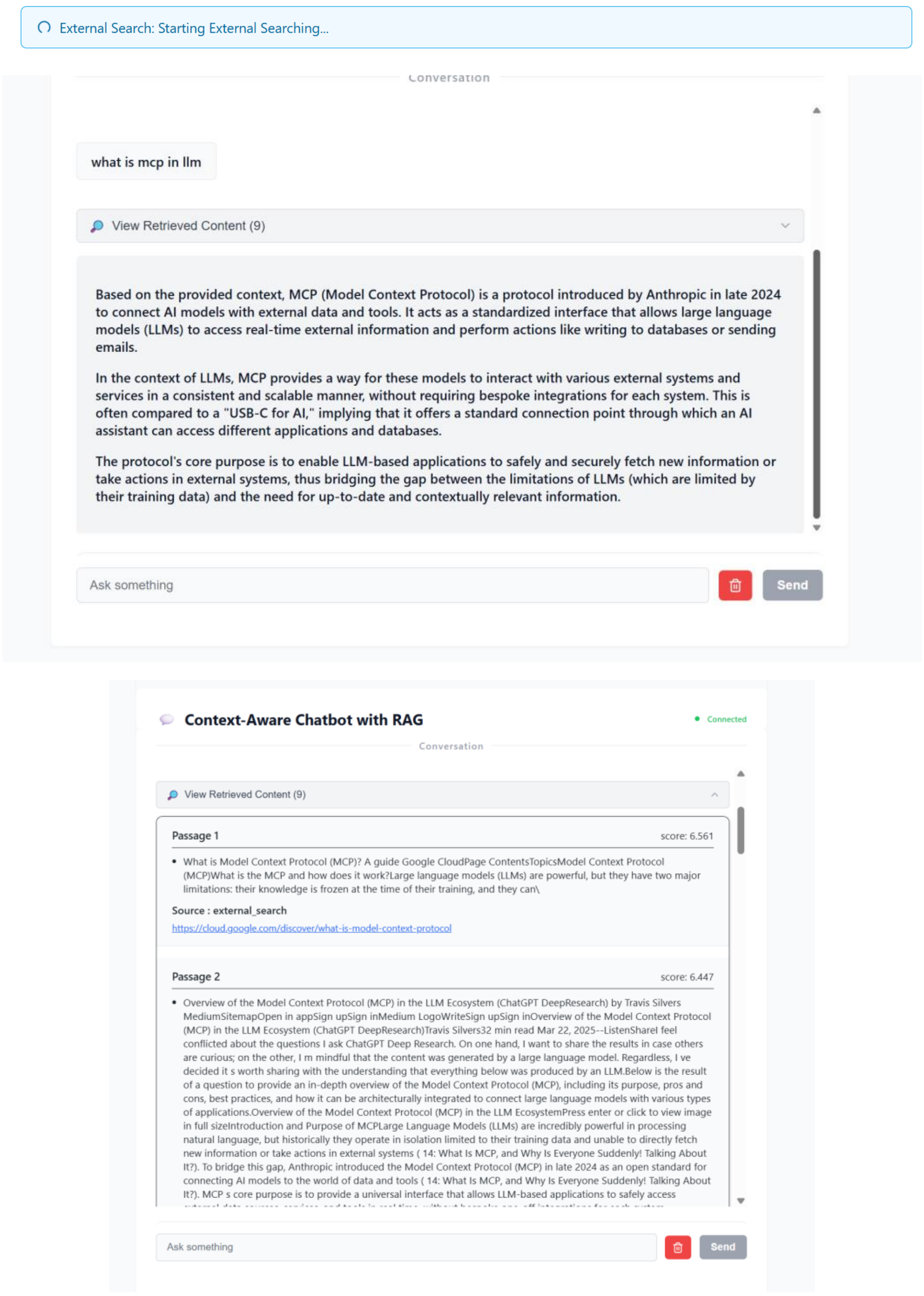


hello

[Stopped by user]

Figure 6.13: Force stop

During any stage before the full response is generated, users are allowed to stop the system process by pressing the stop button, which will stop all processes of the system and allow the user to enter a new query.


External Search: Starting External Searching...
Conversation
View Retrieved Content (9)
Ask something
Send
Context-Aware Chatbot with RAG
Connected
Conversation
View Retrieved Content (9)
Ask something
Send

what is mcp in llm

Based on the provided context, MCP (Model Context Protocol) is a protocol introduced by Anthropic in late 2024 to connect AI models with external data and tools. It acts as a standardized interface that allows large language models (LLMs) to access real-time external information and perform actions like writing to databases or sending emails.

In the context of LLMs, MCP provides a way for these models to interact with various external systems and services in a consistent and scalable manner, without requiring bespoke integrations for each system. This is often compared to a "USB-C for AI," implying that it offers a standard connection point through which an AI assistant can access different applications and databases.

The protocol's core purpose is to enable LLM-based applications to safely and securely fetch new information or take actions in external systems, thus bridging the gap between the limitations of LLMs (which are limited by their training data) and the need for up-to-date and contextually relevant information.

Passage 1 score: 6.561

- What is Model Context Protocol (MCP)? A guide Google CloudPage ContentsTopicsModel Context Protocol (MCP)What is the MCP and how does it work?Large language models (LLMs) are powerful, but they have two major limitations: their knowledge is frozen at the time of their training, and they can\

Source : external_search

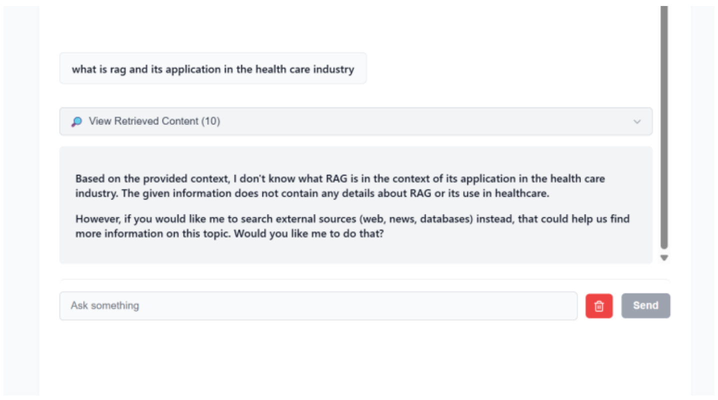
https://cloud.google.com/discover/what-is-model-context-protocol

Passage 2 score: 6.447

- Overview of the Model Context Protocol (MCP) in the LLM Ecosystem (ChatGPT DeepResearch) by Travis Silvers MediumSitemapOpen in appSign upSign inMedium LogoWriteSign upSign inOverview of the Model Context Protocol (MCP) in the LLM Ecosystem (ChatGPT DeepResearch)Travis Silvers32 min read Mar 22, 2025--ListenShareI feel conflicted about the questions I ask ChatGPT Deep Research. On one hand, I want to share the results in case others are curious; on the other, I m mindful that the content was generated by a large language model. Regardless, I ve decided it s worth sharing with the understanding that everything below was produced by an LLM.Below is the result of a question to provide an in-depth overview of the Model Context Protocol (MCP), including its purpose, pros and cons, best practices, and how it can be architecturally integrated to connect large language models with various types of applications.Overview of the Model Context Protocol (MCP) in the LLM EcosystemPress enter or click to view image in full sizeIntroduction and Purpose of MCPLarge Language Models (LLMs) are incredibly powerful in processing natural language, but historically they operate in isolation limited to their training data and unable to directly fetch new information or take actions in external systems (14: What Is MCP, and Why Is Everyone Suddenly! Talking About It?). To bridge this gap, Anthropic introduced the Model Context Protocol (MCP) in late 2024 as an open standard for connecting AI models to the world of data and tools (14: What Is MCP, and Why Is Everyone Suddenly! Talking About It?). MCP s core purpose is to provide a universal interface that allows LLM-based applications to safely access

Figure 6.14: Retrieval workflow

When the user query requires additional information to support the generation, the system will initiate the retrieval process depending on the user's settings. If the user chooses to retrieve from the internet in the setting, the system will scrape the information from the relevant websites or database and use it for response generation and return the relevant information to the frontend interface. Otherwise, the system will initialize the document search process that will fetch all relevant chunks or information from the deviate database with hybrid retrieval. Thus, the user can see the relevant information in the dropdown menu in the interface while the response is being generated

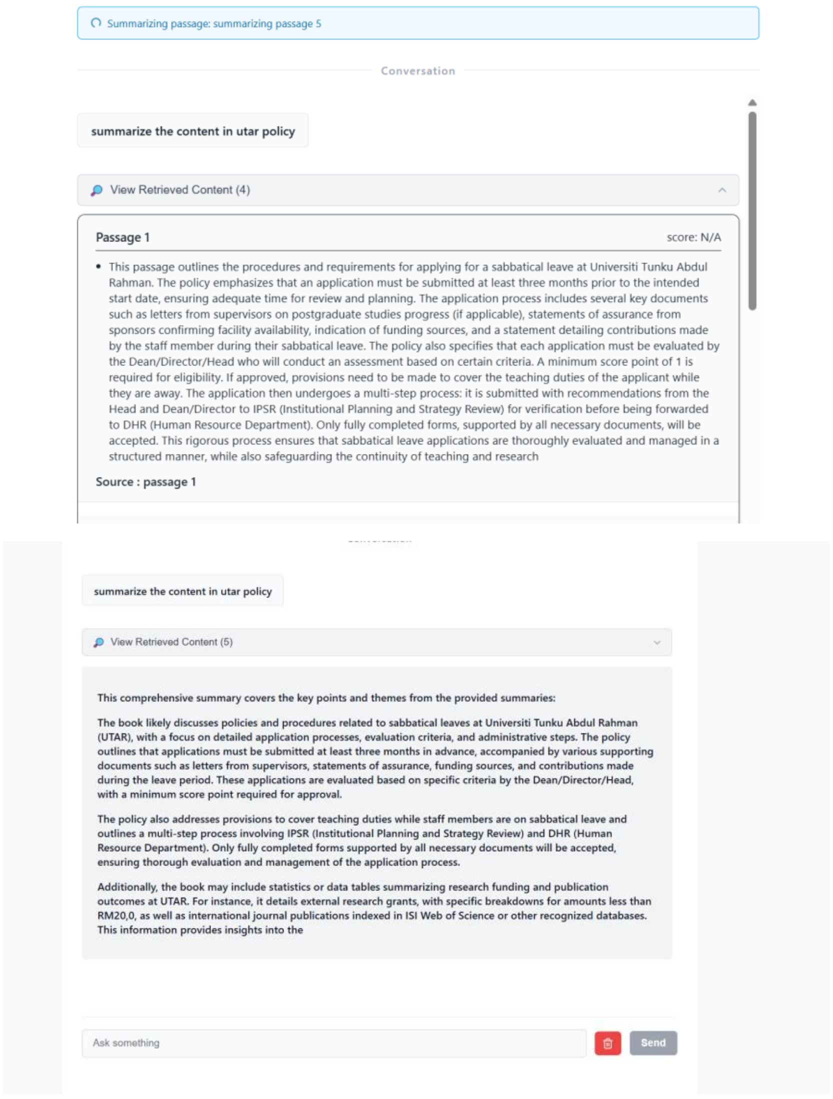

what is rag and its application in the health care industry

Based on the provided context, I don't know what RAG is in the context of its application in the health care industry. The given information does not contain any details about RAG or its use in healthcare.

However, if you would like me to search external sources (web, news, databases) instead, that could help us find more information on this topic. Would you like me to do that?

Figure 6.15: Unrelated query inserted

If the system fails to retrieve relevant information from the vector database. The system will not elaborate on the wrong response, and it will suggest that the user switch the retrieval source, ensuring the response generated is fully based on grounded information.


Summarizing passage: summarizing passage 5
Conversation
View Retrieved Content (4)
Ask something
Send

summarize the content in utar policy

Passage 1 score: N/A

- This passage outlines the procedures and requirements for applying for a sabbatical leave at Universiti Tunku Abdul Rahman. The policy emphasizes that an application must be submitted at least three months prior to the intended start date, ensuring adequate time for review and planning. The application process includes several key documents such as letters from supervisors on postgraduate studies progress (if applicable), statements of assurance from sponsors confirming facility availability, indication of funding sources, and a statement detailing contributions made by the staff member during their sabbatical leave. The policy also specifies that each application must be evaluated by the Dean/Director/Head who will conduct an assessment based on certain criteria. A minimum score point of 1 is required for eligibility. If approved, provisions need to be made to cover the teaching duties of the applicant while they are away. The application then undergoes a multi-step process: it is submitted with recommendations from the Head and Dean/Director to IPSR (Institutional Planning and Strategy Review) for verification before being forwarded to DHR (Human Resource Department). Only fully completed forms, supported by all necessary documents, will be accepted. This rigorous process ensures that sabbatical leave applications are thoroughly evaluated and managed in a structured manner, while also safeguarding the continuity of teaching and research

Source : passage 1

summarize the content in utar policy

View Retrieved Content (5)

This comprehensive summary covers the key points and themes from the provided summaries:

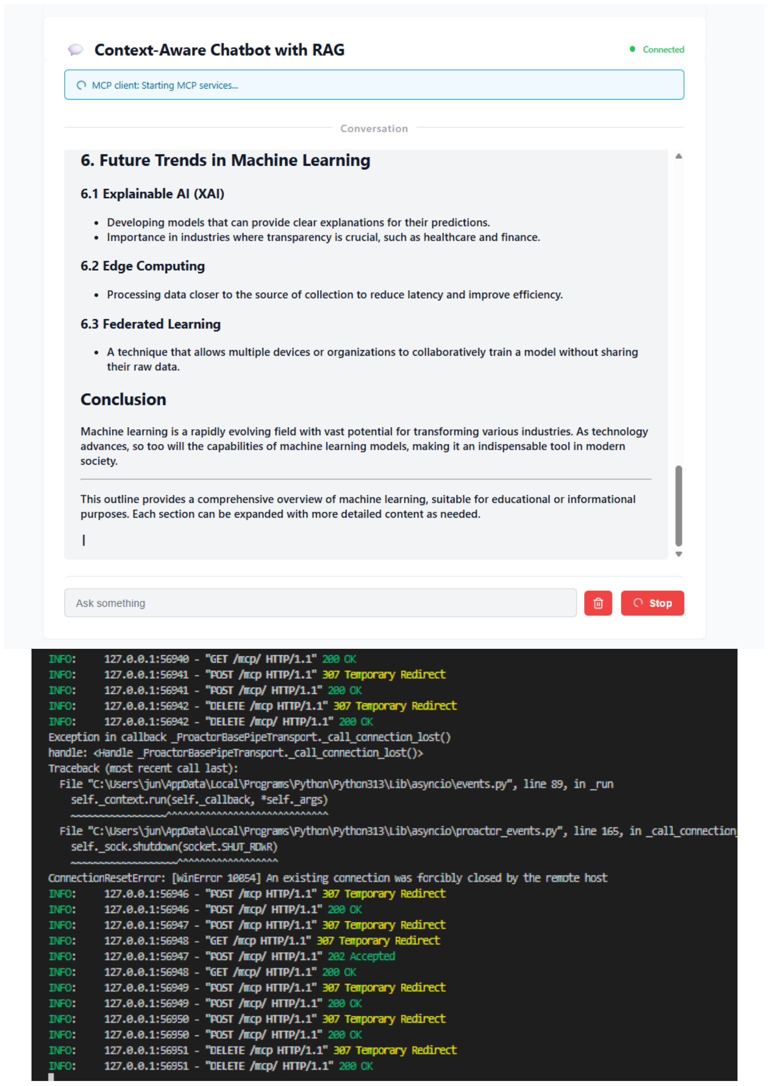
The book likely discusses policies and procedures related to sabbatical leaves at Universiti Tunku Abdul Rahman (UTAR), with a focus on detailed application processes, evaluation criteria, and administrative steps. The policy outlines that applications must be submitted at least three months in advance, accompanied by various supporting documents such as letters from supervisors, statements of assurance, funding sources, and contributions made during the leave period. These applications are evaluated based on specific criteria by the Dean/Director/Head, with a minimum score point required for approval.

The policy also addresses provisions to cover teaching duties while staff members are on sabbatical leave and outlines a multi-step process involving IPSR (Institutional Planning and Strategy Review) and DHR (Human Resource Department). Only fully completed forms supported by all necessary documents will be accepted, ensuring thorough evaluation and management of the application process.

Additionally, the book may include statistics or data tables summarizing research funding and publication outcomes at UTAR. For instance, it details external research grants, with specific breakdowns for amounts less than RM20,0, as well as international journal publications indexed in ISI Web of Science or other recognized databases. This information provides insights into the

Figure 6.16: Document summarization

When the system detects a summarization keyword together with a matching document name in the document name vector database, the system will automatically initiate the summarization workflow. In this workflow, the system will fetch the 5 most relevant passages from the document and summarize the passage content one by one, and send the summarized passage content to the frontend. After summarizing all of 5 passages, the system combines the summarized content, analyses it in context, and generates a final comprehensive explanation of the overall document.


Context-Aware Chatbot with RAG
Connected
MCP client: Starting MCP services...
Conversation
Ask something
Stop

6. Future Trends in Machine Learning

6.1 Explainable AI (XAI)

- Developing models that can provide clear explanations for their predictions.
- Importance in industries where transparency is crucial, such as healthcare and finance.

6.2 Edge Computing

- Processing data closer to the source of collection to reduce latency and improve efficiency.

6.3 Federated Learning

- A technique that allows multiple devices or organizations to collaboratively train a model without sharing their raw data.

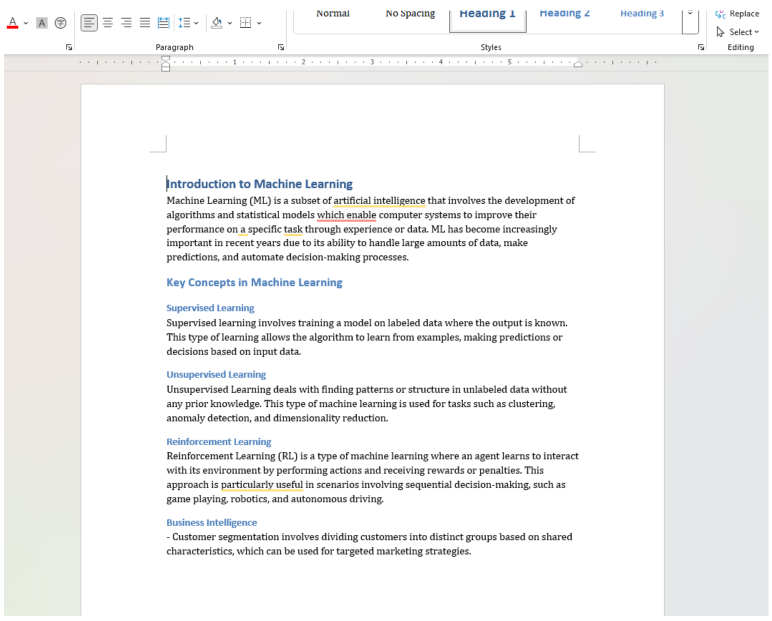
Conclusion

Machine learning is a rapidly evolving field with vast potential for transforming various industries. As technology advances, so too will the capabilities of machine learning models, making it an indispensable tool in modern society.

This outline provides a comprehensive overview of machine learning, suitable for educational or informational purposes. Each section can be expanded with more detailed content as needed.

```
INFO:     127.0.0.1:56940 - "GET /mcp/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:56941 - "POST /mcp HTTP/1.1" 307 Temporary Redirect
INFO:     127.0.0.1:56941 - "POST /mcp/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:56942 - "DELETE /mcp HTTP/1.1" 307 Temporary Redirect
INFO:     127.0.0.1:56942 - "DELETE /mcp/ HTTP/1.1" 200 OK
Exception in callback _ProactorBasePipeTransport._call_connection_lost()
handle: <Handle _ProactorBasePipeTransport._call_connection_lost()>
Traceback (most recent call last):
  File "C:\Users\jun\AppData\Local\Programs\Python\Python313\Lib\asyncio\events.py", line 89, in _run
    self._context.run(self._callback, *self._args)
    ~~~~~~~~~~~~~~~~~^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\jun\AppData\Local\Programs\Python\Python313\Lib\asyncio\proactor_events.py", line 165, in _call_connection
    self._sock.shutdown(socket.SHUT_RDWR)
    ~~~~~~~~~~~~~~~~~~~^^^^^^^^^^^^^^^^^^
ConnectionResetError: [WinError 10054] An existing connection was forcibly closed by the remote host
INFO:     127.0.0.1:56946 - "POST /mcp HTTP/1.1" 307 Temporary Redirect
INFO:     127.0.0.1:56946 - "POST /mcp/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:56947 - "POST /mcp HTTP/1.1" 307 Temporary Redirect
INFO:     127.0.0.1:56948 - "GET /mcp HTTP/1.1" 307 Temporary Redirect
INFO:     127.0.0.1:56947 - "POST /mcp/ HTTP/1.1" 202 Accepted
INFO:     127.0.0.1:56948 - "GET /mcp/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:56949 - "POST /mcp HTTP/1.1" 307 Temporary Redirect
INFO:     127.0.0.1:56949 - "POST /mcp/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:56950 - "POST /mcp HTTP/1.1" 307 Temporary Redirect
INFO:     127.0.0.1:56950 - "POST /mcp/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:56951 - "DELETE /mcp HTTP/1.1" 307 Temporary Redirect
INFO:     127.0.0.1:56951 - "DELETE /mcp/ HTTP/1.1" 200 OK
```

Figure 6.17: MCP server terminal



Introduction to Machine Learning

Machine Learning (ML) is a subset of artificial intelligence that involves the development of algorithms and statistical models which enable computer systems to improve their performance on a specific task through experience or data. ML has become increasingly important in recent years due to its ability to handle large amounts of data, make predictions, and automate decision-making processes.

Key Concepts in Machine Learning

Supervised Learning

Supervised learning involves training a model on labeled data where the output is known. This type of learning allows the algorithm to learn from examples, making predictions or decisions based on input data.

Unsupervised Learning

Unsupervised Learning deals with finding patterns or structure in unlabeled data without any prior knowledge. This type of machine learning is used for tasks such as clustering, anomaly detection, and dimensionality reduction.

Reinforcement Learning

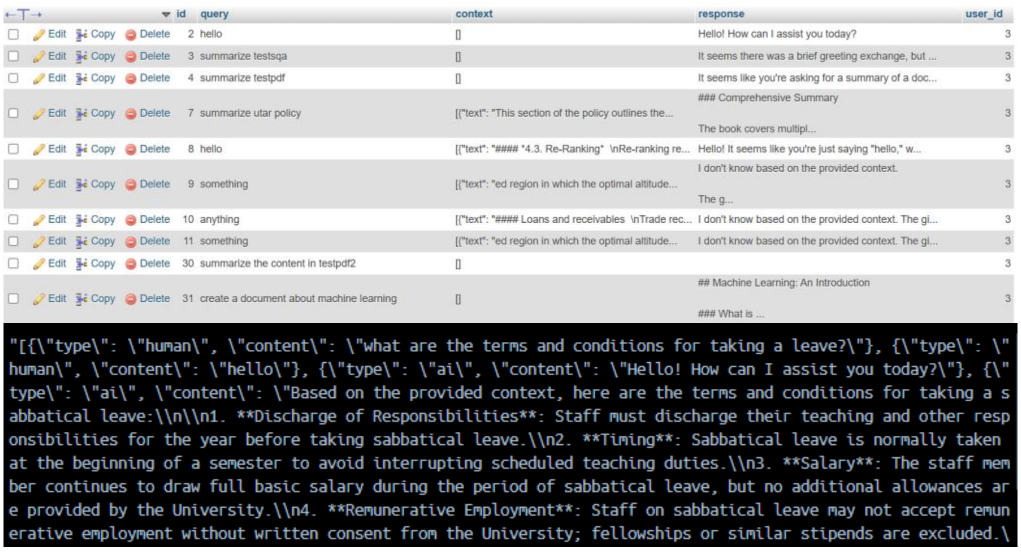
Reinforcement Learning (RL) is a type of machine learning where an agent learns to interact with its environment by performing actions and receiving rewards or penalties. This approach is particularly useful in scenarios involving sequential decision-making, such as game playing, robotics, and autonomous driving.

Business Intelligence

- Customer segmentation involves dividing customers into distinct groups based on shared characteristics, which can be used for targeted marketing strategies.

Figure 6.18: Word document generated

After the final response is generated, the system re-analyses the user query to check whether it contains a document-related operation, such as creating a new document or adding content to an existing one. If detected, the system will initialize the document automation workflow. In this workflow, the system will communicate with the external MCP server that provides access to various tools for document processing. Based on the query, the system invokes the necessary tools through the MCP server to execute the required document automation tasks.



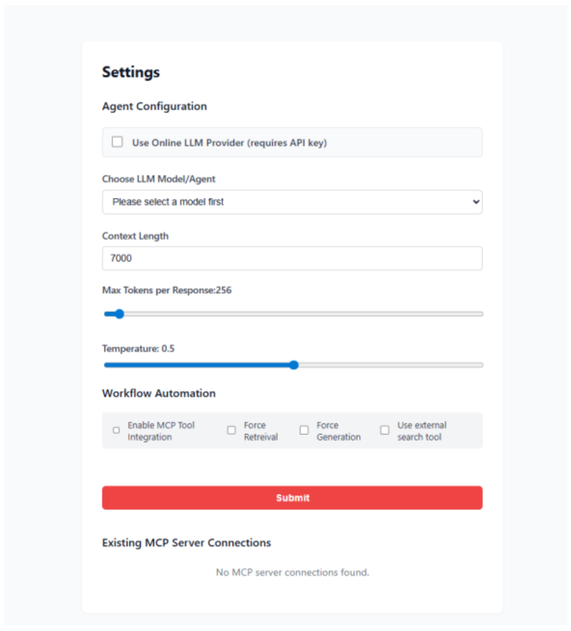
←T→			id	query	context	response	user_id
Edit	Copy	Delete	2	hello	[]	Hello! How can I assist you today?	3
Edit	Copy	Delete	3	summarize testsqa	[]	It seems there was a brief greeting exchange, but ...	3
Edit	Copy	Delete	4	summarize testpdf	[]	It seems like you're asking for a summary of a doc...	3
Edit	Copy	Delete	7	summarize utar policy	[{"text": "This section of the policy outlines the...	### Comprehensive Summary The book covers multipl...	3
Edit	Copy	Delete	8	hello	[{"text": "#### *4.3. Re-Ranking* \nRe-ranking re...	Hello! It seems like you're just saying "hello," w...	3
Edit	Copy	Delete	9	something	[{"text": "ed region in which the optimal altitude...	I don't know based on the provided context. The g...	3
Edit	Copy	Delete	10	anything	[{"text": "#### Loans and receivables \nTrade rec...	I don't know based on the provided context. The gi...	3
Edit	Copy	Delete	11	something	[{"text": "ed region in which the optimal altitude...	I don't know based on the provided context. The gi...	3
Edit	Copy	Delete	30	summarize the content in testpdf2	[]		3
Edit	Copy	Delete	31	create a document about machine learning	[]	## Machine Learning: An Introduction ### What is ...	3

```
"[{\"type\": \"human\", \"content\": \"what are the terms and conditions for taking a leave?\"}, {\"type\": \"
human\", \"content\": \"hello\"}, {\"type\": \"ai\", \"content\": \"Hello! How can I assist you today?\"}, {\"
type\": \"ai\", \"content\": \"Based on the provided context, here are the terms and conditions for taking a s
abbatical leave:\\n\\n1. **Discharge of Responsibilities**: Staff must discharge their teaching and other resp
onsibilities for the year before taking sabbatical leave.\\n2. **Timing**: Sabbatical leave is normally taken
at the beginning of a semester to avoid interrupting scheduled teaching duties.\\n3. **Salary**: The staff mem
ber continues to draw full basic salary during the period of sabbatical leave, but no additional allowances ar
e provided by the University.\\n4. **Remunerative Employment**: Staff on sabbatical leave may not accept remun
erative employment without written consent from the University; fellowships or similar stipends are excluded.\
```

Figure 6.19: Chat History Store In Databases

After finishing one chat response, the content of the chat session will be stored in two databases, which are MySQL for long-term storage and Redis for short-term storage. The history store in MySQL will be used to ensure the previous chat can be fetched even if the user logs out or goes to a different interface. Meanwhile, Redis is leveraged for its fast retrieval capabilities, allowing the generation module to access recent conversation history quickly. To prevent context overload, only the last three conversation entries are stored in Redis, ensuring that responses remain both efficient and contextually accurate.

6.6 Settings module



Settings

Agent Configuration

☐ Use Online LLM Provider (requires API key)

Choose LLM Model/Agent

Please select a model first

Context Length

7000

Max Tokens per Response:256

Temperature: 0.5

Workflow Automation

☐ Enable MCP Tool Integration ☐ Force Retreival ☐ Force Generation ☐ Use external search tool

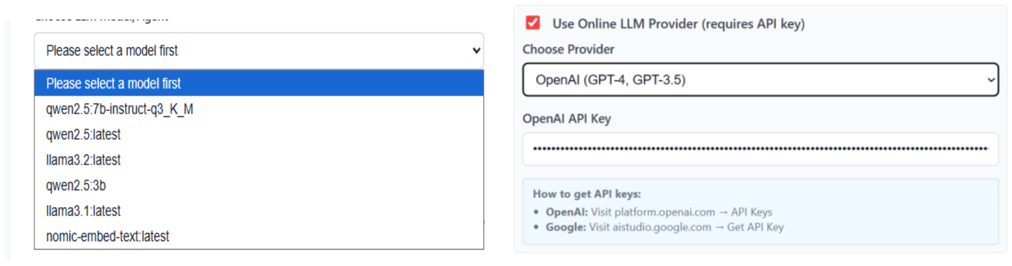
Submit

Existing MCP Server Connections

No MCP server connections found.

Figure 6.20: Setting interface

The setting module will allow the user to customize various component configurations of the system, including the LM model used, context length limit, output token limit, temperature, and workflow configuration.



Please select a model first

- Please select a model first
- qwen2.5:7b-instruct-q3_K_M
- qwen2.5:latest
- llama3.2:latest
- qwen2.5:3b
- llama3.1:latest
- nomic-embed-text:latest

☑ Use Online LLM Provider (requires API key)

Choose Provider

OpenAI (GPT-4, GPT-3.5)

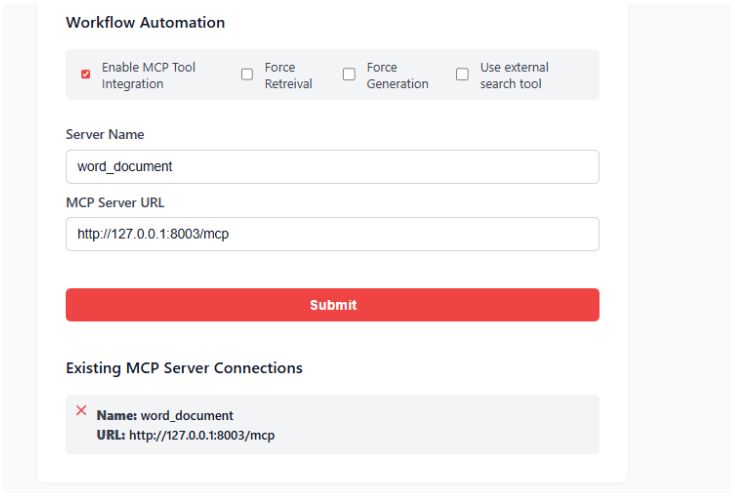
OpenAI API Key

••

How to get API keys:
- **OpenAI:** Visit platform.openai.com → API Keys
- **Google:** Visit aistudio.google.com → Get API Key

Figure 6.21: LLM configuration

For the local model, the system will detect all of the Ollama LLM models in the system, and the user can choose the LLM model to use in this system. But the system also allows users to integrate an external LLM connection via API, where users can check the use online LLM provider option, and they can choose to use either the OpenAI model or the Gemini model in this system.


Workflow Automation

- [✓ Enable MCP Tool Integration]
- Force Retreival
- Force Generation
- Use external search tool

Server Name: word_document

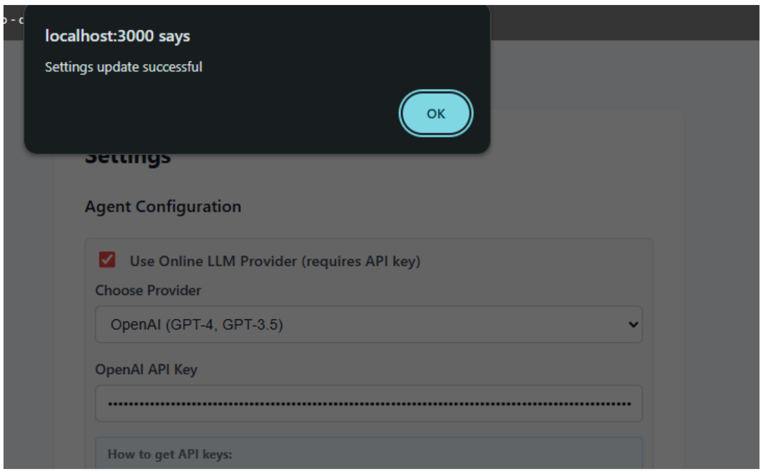
MCP Server URL: http://127.0.0.1:8003/mcp

Existing MCP Server Connections

Name: word_document
URL: http://127.0.0.1:8003/mcp

Figure 6.22: MCP Server configuration

To enable external document workflow, the user can check the Enable MCP Tool Integration option in the settings. Once enabled, the system displays an input field where the user can enter the MCP server URL to establish a connection via HttpStreamable. After creating a new server connection, the list of connected servers will show at the bottom of the settings interface, and the user can delete if needed.


localhost:3000 says

Settings update successful

Settings

Agent Configuration

[✓ Use Online LLM Provider (requires API key)]

Choose Provider: OpenAI (GPT-4, GPT-3.5)

OpenAI API Key: ••

How to get API keys:

Figure 6.23:Setting submitted

After the user completes their setting changes and submits them, the system will display a confirmation message indicating that the update has been successfully applied. This notification ensures that the user is clearly informed of the successful update.

6.7 User management module

```
DEPLOYMENT_MODE = organization
```

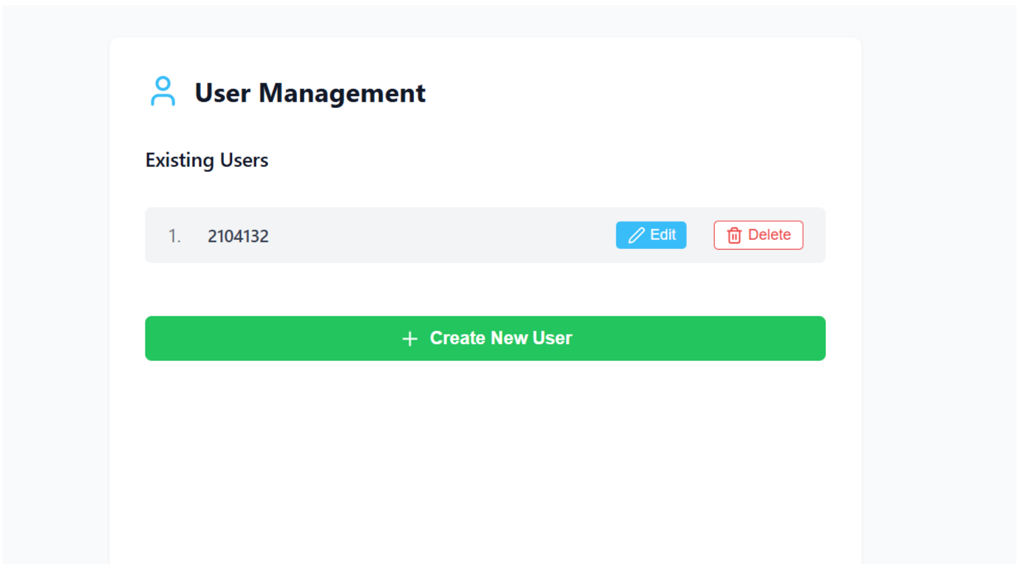


Figure 6.24: User management interface

For the organization workflow, the account with the role admin will have a new interface option, which is the user management interface. In this interface, the admin can see all of the users in this system and perform CRUD actions on the users.

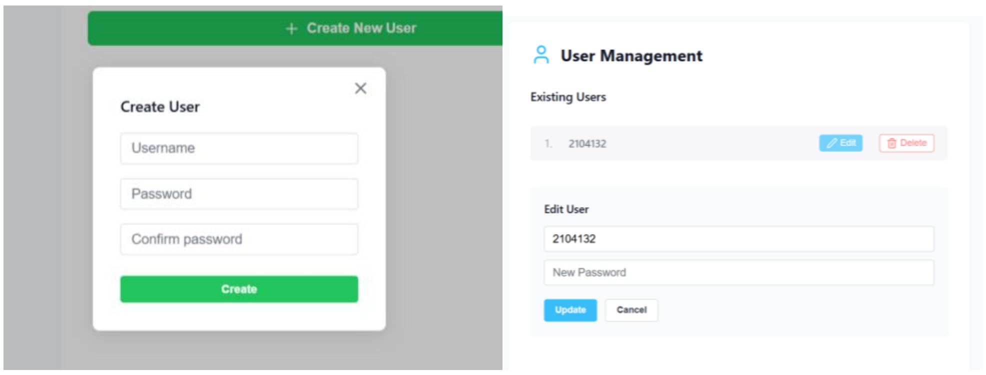


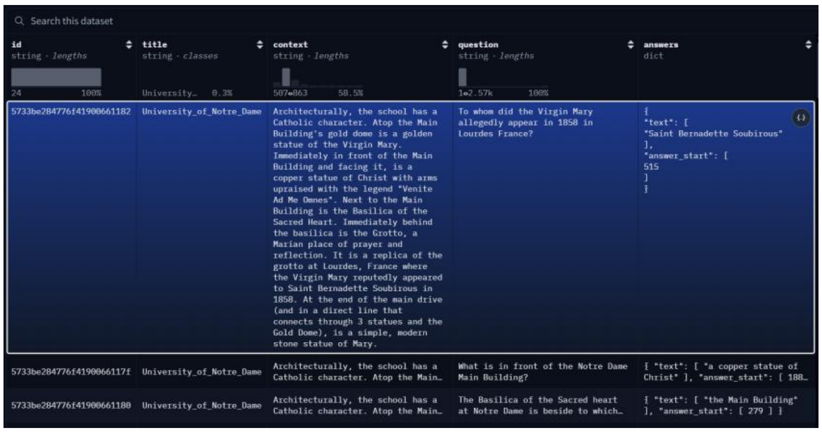
Figure 6.25: CRUD operation for user management

The admin can create a new user, update the information about an existing user and delete any user. When any of the options is chosen, the system displays an input form where the admin can enter the required details to apply the changes.

CHAPTER 7

SYSTEM TESTINGS AND DISCUSSION

7.1 Discussion



Search this dataset

id string · lengths 24 100%	title string · classes University… 0.3%	context string · lengths 507↔863 58.5%	question string · lengths 1↔2.57k 100%	answers dict
5733be284776f41900661182	University_of_Notre_Dame	Architecturally, the school has a Catholic character. Atop the Main Building's gold dome is a golden statue of the Virgin Mary. Immediately in front of the Main Building and facing it, is a copper statue of Christ with arms upraised with the legend "Venite Ad Me Omnes". Next to the Main Building is the Basilica of the Sacred Heart. Immediately behind the basilica is the Grotto, a Marian place of prayer and reflection. It is a replica of the grotto at Lourdes, France where the Virgin Mary reputedly appeared to Saint Bernadette Soubirous in 1858. At the end of the main drive (and in a direct line that connects through 3 statues and the Gold Dome), is a simple, modern stone statue of Mary.	To whom did the Virgin Mary allegedly appear in 1858 in Lourdes France?	{ "text": ["Saint Bernadette Soubirous"], "answer_start": [515] }
5733be284776f4190066117f	University_of_Notre_Dame	Architecturally, the school has a Catholic character. Atop the Main…	What is in front of the Notre Dame Main Building?	{ "text": ["a copper statue of Christ"], "answer_start": [188…
5733be284776f41900661180	University_of_Notre_Dame	Architecturally, the school has a Catholic character. Atop the Main…	The Basilica of the Sacred heart at Notre Dame is beside to which…	{ "text": ["the Main Building"], "answer_start": [279] }

Figure 7.1: Segment of SQuAD on hugging face

To evaluate the performance of the two-core component in this system, which includes the retrieval and generation components. The performance of the component is evaluated with the Stanford Question Answering Dataset (SQuAD), a popular benchmark dataset for measuring question-answering systems. The SQuAD dataset contains passages from various Wikipedia articles with provided questions and golden answers to the questions, making it suitable for the evaluation of the retrieval and the generation components in this system.

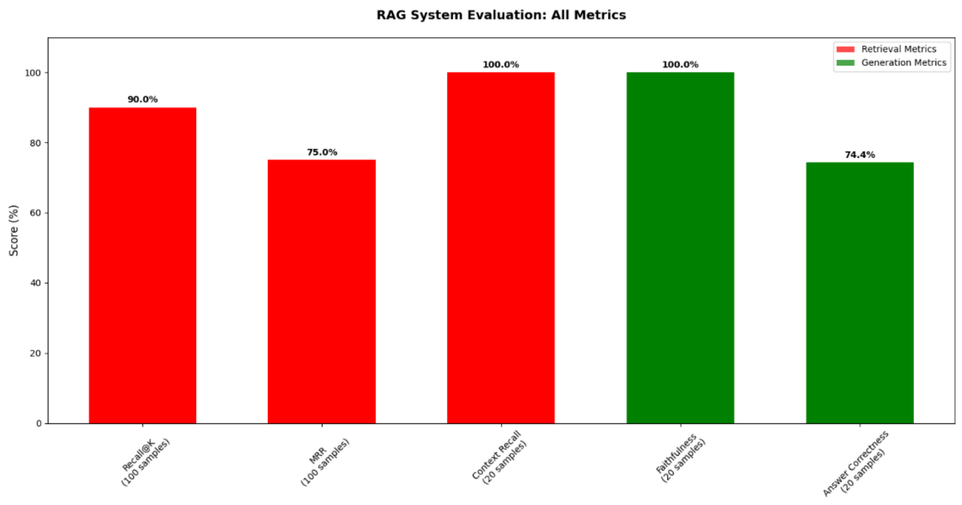


Figure 7.2:Result chart

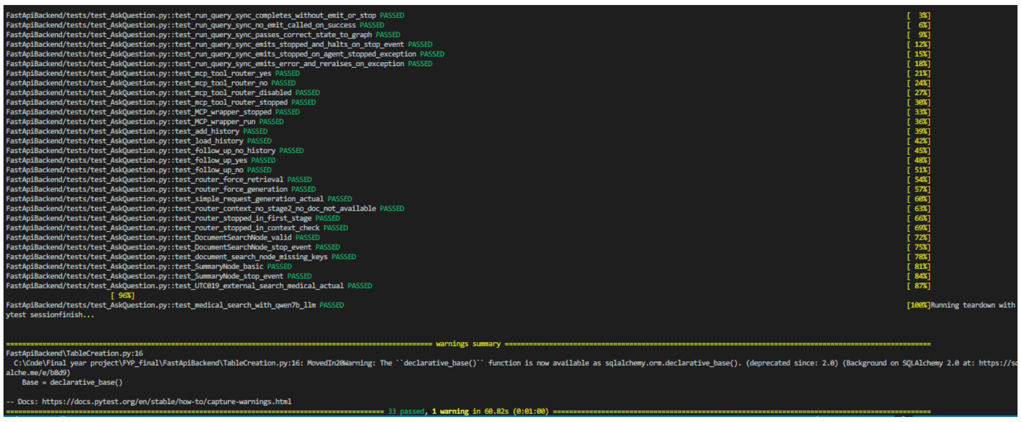
The chart presents the evaluation results of the multi-agent RAG system with SQuAD across two categories of metrics: retrieval performance and generation performance. The retrieval performance is measured with metrics such as Recall@3, Mean Reciprocal Rank (MRR) and Context Recall, while the generation performance is measured with metrics such as Faithfulness and AnswerCorrectness. Moreover, two evaluation approaches were used, with a calculation-based method applied to a sample size of 100, and an LLM-as-Judge method using the GPT-4o model, applied to a smaller sample size of 20 due to token and request limitations. The LLM-as-Judge approach was adopted because some metrics are difficult to measure with a calculation-based method and this approach can leverage the flexibility of LLM to provide more human-like evaluation for the answer quality.

For retrieval, the system's retrieval component achieves a Recall@3 of 90% which means that out of 100 samples, the most relevant context is present within the top 3 passages retrieved by the system. Moreover, the retrieval component also achieved an MRR score of 75%, meaning that the correct context is retrieved at an average rank of top 2 or 1. Lastly, the system manages to score 100% in context recall, which, as referenced in SQuAD, through reference context demonstrates the retrieval component's ability to consistently retrieve the correct context.

For generations, the system generation component performs excellently in terms of Faithfulness, which the system achieves a perfect score, meaning that the generation is consistently grounded in its responses based on the retrieved context without hallucinating. However, the system only achieves 74.4% accuracy in the AnswerCorrectness metric, meaning that although the answer is accurate most of the time but there are still times when the response generated is not accurate. It is important to note that this evaluation was conducted using the Llama 3.1:8B model, which is relatively small. Thus, this limitation can possibly contribute to the lower accuracy observed.

7.2 Unit test

Unit testing was performed to evaluate every major component to ensure the module functions as intended. To ensure proper traceability, the unit test cases were designed in alignment with the defined use cases. This approach ensure a clear link between system requirements and the corresponding test scenarios.



```
FastApiBackend/tests/test_AskQuestion.py::test_run_query_sync_completes_without_emit_or_stop PASSED                  [  3%]
FastApiBackend/tests/test_AskQuestion.py::test_run_query_sync_no_emit_called_on_success PASSED                       [  6%]
FastApiBackend/tests/test_AskQuestion.py::test_run_query_sync_passes_correct_state_to_graph PASSED                   [  9%]
FastApiBackend/tests/test_AskQuestion.py::test_run_query_sync_emits_stopped_and_halts_on_stop_event PASSED           [ 12%]
FastApiBackend/tests/test_AskQuestion.py::test_run_query_sync_emits_stopped_on_agent_stopped_exception PASSED         [ 15%]
FastApiBackend/tests/test_AskQuestion.py::test_run_query_sync_emits_error_and_reraises_on_exception PASSED           [ 18%]
FastApiBackend/tests/test_AskQuestion.py::test_mcp_tool_router_yes PASSED                                            [ 21%]
FastApiBackend/tests/test_AskQuestion.py::test_mcp_tool_router_no PASSED                                             [ 24%]
FastApiBackend/tests/test_AskQuestion.py::test_mcp_tool_router_disabled PASSED                                       [ 27%]
FastApiBackend/tests/test_AskQuestion.py::test_mcp_tool_router_stopped PASSED                                        [ 30%]
FastApiBackend/tests/test_AskQuestion.py::test_MCP_wrapper_stopped PASSED                                            [ 33%]
FastApiBackend/tests/test_AskQuestion.py::test_MCP_wrapper_run PASSED                                                [ 36%]
FastApiBackend/tests/test_AskQuestion.py::test_add_history PASSED                                                    [ 39%]
FastApiBackend/tests/test_AskQuestion.py::test_load_history PASSED                                                   [ 42%]
FastApiBackend/tests/test_AskQuestion.py::test_follow_up_no_history PASSED                                           [ 45%]
FastApiBackend/tests/test_AskQuestion.py::test_follow_up_yes PASSED                                                  [ 48%]
FastApiBackend/tests/test_AskQuestion.py::test_follow_up_no PASSED                                                   [ 51%]
FastApiBackend/tests/test_AskQuestion.py::test_router_force_retrieval PASSED                                         [ 54%]
FastApiBackend/tests/test_AskQuestion.py::test_router_force_generation PASSED                                        [ 57%]
FastApiBackend/tests/test_AskQuestion.py::test_simple_request_generation_actual PASSED                               [ 60%]
FastApiBackend/tests/test_AskQuestion.py::test_router_context_no_stage2_no_doc_not_available PASSED                  [ 63%]
FastApiBackend/tests/test_AskQuestion.py::test_router_stopped_in_first_stage PASSED                                  [ 66%]
FastApiBackend/tests/test_AskQuestion.py::test_router_stopped_in_context_check PASSED                                [ 69%]
FastApiBackend/tests/test_AskQuestion.py::test_DocumentSearchNode_valid PASSED                                       [ 72%]
FastApiBackend/tests/test_AskQuestion.py::test_DocumentSearchNode_stop_event PASSED                                  [ 75%]
FastApiBackend/tests/test_AskQuestion.py::test_document_search_node_missing_keys PASSED                              [ 78%]
FastApiBackend/tests/test_AskQuestion.py::test_SummaryNode_basic PASSED                                              [ 81%]
FastApiBackend/tests/test_AskQuestion.py::test_SummaryNode_stop_event PASSED                                         [ 84%]
FastApiBackend/tests/test_AskQuestion.py::test_UTC019_external_search_medical_actual PASSED                          [ 87%]
                               [ 96%]
FastApiBackend/tests/test_AskQuestion.py::test_medical_search_with_qwen7b_llm PASSED                                 [100%]Running teardown with
ytest sessionfinish...


================================================================ warnings summary ================================================================
FastApiBackend\TableCreation.py:16
  C:\Code\Final year project\FYP_final\FastApiBackend\TableCreation.py:16: MovedIn20Warning: The ``declarative_base()`` function is now available as sqlalchemy.orm.declarative_base(). (deprecated since: 2.0) (Background on SQLAlchemy 2.0 at: https://sq
alche.me/e/b8d9)
    Base = declarative_base()

-- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
=========================================================== 33 passed, 1 warning in 60.82s (0:01:00) ============================================================
```

Figure 7.3: Test automation script result

Table 7.1: Test cases

Test CaseID	Test Title	Expected result	Test data	Use case	Status
UTC001	Test sign in with a valid credential	User successfully accessed to the main interface	1. Name : 2104132 2. Password: 123	UC01	Pass
UTC002	Test sign in with empty username and password	System prompt username and password cannot be empty	-		Pass
UTC003	Test sign in with invalid credentials	System prompt error	1. Name: nonUser		Pass

		message: "Invalid credentials. Please verify your input."	2. Password: 123123		
UTC004	Test sign in with the existing authentication token in the database.	System skips the sign-in step and navigates to the main interface	1. Token		Pass
UTC005	Test register with a valid credential	User successfully accessed to the main interface	1. Name: 2104132 2. Password:123123 3. Confirm password: 123123	UC02	Pass
UTC006	Test register with an empty field	System prompt username and password cannot be empty	-		Pass
UTC007	Test register with an unmatched password	System error message "password does not match"	1. Name: 2104132 2. Password: 123123		Pass

			3. Confirm password: 123		
UTC008	Test upload a document with a valid file type and size	System successfully initialized the preprocessing process	1. Document: utarpolicy.pdf (85KB)	UC03	Pass
UTC009	Test upload with an invalid document type	System prompt error message: "Only PDF files are allowed!”	1. Document: test.png		Pass
UTC010	Test upload with an invalid size	System prompt error message “Invalid file size”	1. Document: Testpdf.pdf (25MB)		Pass
UTC011	Test manage file interface with the existing file	The system displays all the uploaded file lists	-	UC04	Pass
UTC012	Test file deletion in the manage file interface	The system successfully deleted the document	-		Pass

UTC013	Test manage file interface with no existing file	The system displays "No files uploaded."	-		Pass
UTC014	Test system routing with force generation setting on	The system routes the user request to the generation node	1. Query: "what is an apple" 2. force_generation: True	UC05	Pass
UTC015	Test system routing with force retrieval setting on	The system routes the user request to the retrieval decision node	1. Query: "what are the different types of rag?" 2. force_retrieval: True		Pass
UTC016	Test system routing with a simple request	The system routes user requests to the generation node	1. Query: "What is the capital of France?"		Pass
UTC017	Test system routing with a query that requires the latest information	The system routes the user request to the retrieval decision node	1. Query: "What is MCP, and what are the applications?"		Pass

UTC018	Test system feature with follow-up question	The system analyse and enhances the user query based on the conversation history	1. Query: “Explain more on that.”		Pass
UTC019	Test external search routing with general question	The system routes user request to general web search	1. Query: “What is MCP, and what is the applications?”		Pass
UTC020	Test external search routing with medical medical-specific question	The system route user requests to the medical database search tool	1. Query: “What is the effect of anabolic steroids?”		Pass
UTC021	Test external search routing with the finance new specific question	The system routes user requests to the finance new search tool	1. Query: “Why is Tesla stock down today?”		Pass
UTC022	Test system routing to	The system routes user	1. Query: “why are the		Pass

	document search with no document available	requests to the generation node	different types of rag?"		
UTC023	Test stop feature system processing	The system stops the process and returns the response "stopped by user"	-		Pass
UTC024	Test system response with a question not in the document	The system responds with "The context provided can not answer your question, want to switch to external search?"	1. Query: "who is the winner of cs austin major"		Pass
UTC025	Test system response with a question cannot be found on the internet	The system responds with "The context provided can not answer	1. Query: "what is the term and condition for utar staff?"		Pass

		your question. want to switch to document search?”			
UTC026	Test system response with error occurs during processing	The system displays error message at the chat interface	-		Pass
UTC027	Test system setting changed with a different LLM model applied	The system successfully saved and applied the updated LLM model	1. Model: qwen2.5:7b	UC06	Pass
UTC028	Test system setting changed, external LLM model applied	The system successfully saved and applied the updated LLM model	1. use_online_llm : True 2. online_provider: openai 3. online_api_key		Pass
UTC029	Test system setting changes with LLM configuration updated	The system successfully saved and applied the updated LLM	1. token_response: 6000 2. context_length:7000 3. temperature: 0.4		Pass

		configuration			
UTC030	Test system setting changes with workflow configuration updated	The system successfully saved and applied the updated workflow configuration	1. force_retrieval: True 2. search_mode:external search		Pass
UTC031	Test system setting changes with the MCP workflow configuration updated	The system successfully saved and applied the updated MCP server details	1. enable_mcp: True 2. mcp_details: ”[{\"name\": \"word_document\", \"url\": \"http://127.0.0.1:8003/mcp\"}]”	UC07	Pass
UTC032	Test system setting change in the organization deployment mode	The system successfully saves the updated settings. The is_global flag is set to True, and accounts with the role “user”	1. Deployment_mode: organization		Pass

		can use the shared settings.			
UTC033	Test the document's automated workflow with a valid server connection	The system successfully routes to McpAgent node and communicates with the MCP server to create a document	1. Query: “Create a document about machine learning.” 2. enable_mcp: True 3. mcp_details: ”[{\"name\": \"word_document\", \"url\": \"http://127.0.0.1:8003/mcp\"}]”		Pass
UTC034	Test the document automated workflow with no MCP detail entered	The system skips the document automation workflow	1. Query: “Create a document about machine learning.” 2. Enable_mcp: True 3. Mcp_details: []		Pass
UTC035	Test the document automated workflow	The system displays an alert error message	1. Query: “Create a document about		Pass

	with invalid MCP details	“Error: No MCP servers are reachable. Skipping MCP step.”	machine learning.” 2. Enable_mcp: True 3. Mcp_details: "[{\"name\": \"word_document\", \"url\": \"dummy\"}]		
UTC036	Test the document summarization workflow with a valid summarization request	The system successfully routes the request to the summarization node and summarizes the document	1. Query: “summarize the content in utar policy”	UC08	Pass
UTC037	Test the document summarization workflow with no document available	The system skips the summarization step	1. Query: “summarize content in utar policy”		Pass
UTC038	Test the document summarization workflow with failed	The system stops the system process and	-		Pass

	summarization	displays an error message			

CHAPTER 8

CONCLUSION AND RECOMMENDATIONS

8.1 Conclusion

This is a seven-month project that lasted two semesters from February to September. This project set out to design and develop a Self-Hosted Multi-Agent Retrieval-Augmented Generation (RAG) System for Contextual Document Processing, which follows a set of objectives to address problems identified in common LLM solutions in document processing tasks, including hallucinations, privacy concerns, and lack of modularity. During the planning of this project, the project scope and initial proposed solution were also outlined to provide clearer direction for development. Additionally, a literature review on different existing RAG solutions, MAS design patterns, types of RAG, MAS frameworks, and types of retrieval methods in RAG systems is also carried out to provide a complete study on different aspects of the RAG model and determine which approach and component to be implemented into this system.

After the literature review, a prototype-based development methodology is carefully chosen to ensure an iterative development that allows for continuous refinement and testing throughout the project because of the rapid advancement of different AI tools. Each stage of the development was structured to validate the functionality of the system component before proceeding to develop the full multi-agent workflow. Additionally, various useful tools are chosen, such as marker-pdf, Ollama, and LangGraph, that are used to speed up the development process by providing a ready-to-use component that can provide a solid foundation for the development of this system. During this stage, a clear work plan and timeline are outlined with a work breakdown structure to ensure that the deliverable can be provided on the appropriate deadline. This ensures that every deliverable is aligned with the tight seven-month deadline, which encompasses all phases from initial planning to final project testing.

Moving on to the requirement gathering stage, the system requirements are identified from all of the prior planning phases. Use case and activity diagrams are created to help with the visualization of all of the required functionality of this system. All of the functional and non-functional requirements are identified and recorded to provide a clear guideline for development and testing. The first single-agent RAG system prototype is developed in this stage to provide an overall demonstration of which functionality and design the system will follow to the stakeholders and serve as the foundation for subsequent development of a more complete and robust solution.

The system design for the final solution is developed with an outline of communication between system components and agents. This will be used to support the iteration from the single-agent RAG prototype to the intended final solution. The major frameworks and external services are also identified to help in producing a robust solution.

After the second planning phase, the final system was developed and delivered in line with all defined objectives and requirements by integrating the multi-agent workflow along with the necessary functionalities. The behaviour of the final product was thoroughly recorded to maintain traceability and confirm the completeness of the project.

Lastly, the final system testing is carried out via performance and unit testing, in which the system performances are evaluated with different metrics such as accuracy, ReCall@3, MRR, and faithfulness. This can demonstrate that the system performs effectively in reducing hallucination rate and producing reliable responses. Although some limitations remain in terms of correctness due to smaller-scale LLMs and computational constraints, the system proves that a modular and extensible multi-agent RAG pipeline can function effectively in a self-hosted environment. After that, unit testing is performed based on the use case determined to ensure all major functionality was implemented correctly and that the project objectives were successfully achieved.

In conclusion, this project successfully achieved its objectives by delivering a working solution for a privacy-preserving, modular, and context-aware document processing system. The outcomes highlight the feasibility of integrating multi-agent architectures with RAG technology to produce accurate and reliable results while safeguarding data privacy.

8.2 Limitations & Future Development

Although all of the objectives are successfully achieved in this project, several limitations remain. The system currently only supports English only document processing and is restricted to external search in only three databases, including general web search, finance news database, and medical database. Moreover, the system currently only focuses on textual content processing and does not include any figures and visual data processing. Due to the limitation of the hardware, the response precision is also heavily limited by the scale of the local LLM used and the lack of domain-specific LLM fine-tuning, which the hardware limitation also constrains the real-time capabilities of this system.

Despite the constraints mentioned, the project provides a strong foundation for future development. Potential areas of enhancement include extending the system to support multilingual document processing, improving accessibility and usability across different contexts. Additionally, the integration of multimodal document analysis, including figures, tables, and diagrams, could enrich the scope of content processing. Optimizing the multi-agent workflow for scalability, such as through distributed processing and hardware acceleration, would enable deployment in large-scale enterprise environments. Expanding automation capabilities beyond note generation, incorporating adaptive retrieval strategies, and advancing toward real-time processing would further increase the system's versatility and impact.

By following these directions, this project can evolve into a robust document processing system that can provide immense value to individuals and organizations, positioning itself as a robust and adaptable solution for future enterprise and research applications.

REFERENCES

Amazon Web Services, Inc., *What is LangChain? - LangChain Explained - AWS* [Online]. Available at: https://aws.amazon.com/what-is/langchain/.

Banerjee, S., Agarwal, A. and Singla, S., 2024, *LLMs Will Always Hallucinate, and We Need to Live With This* [Online]. Available at: https://arxiv.org/abs/2409.05746.

Bo, X. et al., 2024. Reflective Multi-Agent Collaboration based on Large Language Models. *Advances in Neural Information Processing Systems*, 37, pp.138595–138631. Available at: https://proceedings.neurips.cc/paper_files/paper/2024/hash/fa54b0edce5eef0bb07654e8ee800cb4-Abstract-Conference.html [Accessed: 15 March 2025].

Cardenas, E. and Monigatti, L., 2024, *What is Agentic RAG* [Online]. Available at: https://weaviate.io/blog/what-is-agentic-rag.

Chauhan, C., 2024. The Impact of Generative Artificial Intelligence in Scientific Content Synthesis for Authors. *American Journal Of Pathology*, 194(8).

Cheng, H. et al., 2024. Have AI-Generated Texts from LLM Infiltrated the Realm of Scientific Writing? A Large-Scale Analysis of Preprint Platforms. *bioRxiv (Cold Spring Harbor Laboratory)*.

Clark, B., 2025, *What is LangGraph?* [Online]. Available at: https://www.ibm.com/think/topics/langgraph.

Cuconasu, F. et al., 2024. The Power of Noise: Redefining Retrieval for RAG Sys-tems. Available at: https://arxiv.org/pdf/2401.14887.

Duan, H., Yang, Y. and Tam, K.Y., 2024, *Do LLMs Know about Hallucination? An Empirical Investigation of LLM's Hidden States* [Online]. Available at: https://arxiv.org/abs/2402.09733 [Accessed: 5 March 2025].

Duan, Z. and Wang, J., 2024, *Exploration of LLM Multi-Agent Application Implementation Based on LangGraph+CrewAI* [Online]. Available at: https://arxiv.org/abs/2411.18241.

Feng, S. et al., 2024, *Don't Hallucinate, Abstain: Identifying LLM Knowledge Gaps via Multi-LLM Collaboration* [Online]. Available at: https://arxiv.org/abs/2402.00367.

Guo, T. et al., 2024. *Large Language Model based Multi-Agents: A Survey of Progress and Challenges*,

Gustavo et al., 2025. From RAG to Multi-Agent Systems: A Survey of Modern Approaches in LLM Development. *From RAG to Multi-Agent Systems: A Survey of Modern Approaches in LLM Development*.

Homayoun S., 2025, *6 Types of Retrieval-Augmented Generation (RAG) Techniques You Should Know* [Online]. Available at: https://homayounsrp.medium.com/6-types-of-retrieval-augmented-generation-rag-techniques-you-should-know-b45de9071c79#bypass [Accessed: 26 April 2025].

IBM, 2023, *LangChain* [Online]. Available at: https://www.ibm.com/think/topics/langchain.

Jiménez, A., García-Díaz, V. and Bolaños, S., 2018. A Decentralized Framework for Multi-Agent Robotic Systems. *Sensors*, 18(2), p.417.

Khan, N., 2024, *Introduction Information retrieval is the task of finding documents that satisfy an information need from a large collection of documents. Given the vast amount of data, efficient information retrieval techniques are essential to a number of applications from web search to recommendations to convers* [Online]. Available at: https://www.linkedin.com/pulse/retrieval-techniques-sparse-dense-hybrid-najeeb-khan-ph-d--wmtpc/ [Accessed: 26 April 2025].

Knight, M., 2025, *What Is a Knowledge Graph? - DATAVERSITY* [Online]. Available at: https://www.dataversity.net/what-is-a-knowledge-graph/ [Accessed: 26 April 2025].

Kumar, S., 2023, *Generative Retrieval for End-to-End Search Systems* [Online]. Available at: https://blog.reachsumit.com/posts/2023/09/generative-retrieval/ [Accessed: 26 April 2025].

Li, M., Miao, S. and Li, P., 2024, *Simple is Effective: The Roles of Graphs and Large Language Models in Knowledge-Graph-Based Retrieval-Augmented Generation* [Online]. Available at: https://arxiv.org/abs/2410.20724.

Li, X. et al., 2024. A survey on LLM-based multi-agent systems: workflow, infrastructure, and challenges. , 1(1).

Liu, Y. et al., 2024, *Agent Design Pattern Catalogue: A Collection of Architectural Patterns for Foundation Model based Agents* [Online]. Available at: https://arxiv.org/abs/2405.10467.

Mahadevkar, S.V. et al., 2024. Exploring AI-driven approaches for unstructured document analysis and future horizons. *Journal of big data*, 11(1).

Milvus, 2025, *What is the difference between sparse and dense retrieval?* [Online]. Available at: https://milvus.io/ai-quick-reference/what-is-the-difference-between-sparse-and-dense-retrieval [Accessed: 12 April 2025].

Oguzhan Topsakal and Tahir Cetin Akinci, 2023. Creating Large Language Model Applications Utilizing LangChain: A Primer on Developing LLM Apps Fast. *International Conference on Applied Engineering and Natural Sciences*, 1(1), pp.1050–1056.

Peng, C., Xia, F., Naseriparsa, M. and Osborne, F., 2023, *Knowledge Graphs: Opportunities and Challenges* [Online]. Available at: https://arxiv.org/abs/2303.13948 [Accessed: 25 April 2025].

Ramalingam, S., 2023. *RAG in Action: Building the Future of AI-Driven Applications*, Libertatem Media Private Limited.

Ravuru, C., Sakhinana, Sagar Srinivas and Runkana, V., 2024, *Agentic Retrieval-Augmented Generation for Time Series Analysis* [Online]. Available at: https://arxiv.org/abs/2408.14484.

Reinsel, D., Gantz, J. and Rydning, J., 2018. *The Digitization of the World From Edge to Core*,

Sanmartin, D., 2024, *KG-RAG: Bridging the Gap Between Knowledge and Creativity* [Online]. Available at: https://arxiv.org/abs/2405.12035.

Singh, A., Ehtesham, A., Kumar, S. and Khoei, Tala Talaei, 2025, *Agentic Retrieval-Augmented Generation: A Survey on Agentic RAG* [Online]. Available at: https://arxiv.org/abs/2501.09136.

Tran, K.-T. et al., 2025, *Multi-Agent Collaboration Mechanisms: A Survey of LLMs* [Online]. Available at: https://arxiv.org/abs/2501.06322#page=5.32 [Accessed: 15 March 2025].

Wang, J. and Duan, Z., 2024, *Agent AI with LangGraph: A Modular Framework for Enhancing Machine Translation Using Large Language Models* [Online]. Available at: https://arxiv.org/abs/2412.03801 [Accessed: 26 January 2025].

Winland, V., Syed, M. and Gutowska, A., 2024, *crewAI* [Online]. Available at: https://www.ibm.com/think/topics/crew-ai.

Wu, Q. et al., 2023, *AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation* [Online]. Available at: https://arxiv.org/abs/2308.08155.

Wu, X., Duan, R. and Ni, J., 2023. Unveiling Security, Privacy, and Ethical Concerns of ChatGPT. *Journal of Information and Intelligence*, 2(2). Available at: https://www.sciencedirect.com/science/article/pii/S2949715923000707.

Yan, S.-Q., Gu, J.-C., Zhu, Y. and Ling, Z.-H., 2024, *Corrective Retrieval Augmented Generation* [Online]. Available at: https://arxiv.org/abs/2401.15884 [Accessed: 25 March 2025].

Yu, C., Yan, J. and Cai, N., 2024. ChatGPT in higher education: factors influencing ChatGPT user satisfaction and continued use intention. *Frontiers in education*, 9(1).

Zeeshan, T., Kumar, A., Pirttikangas, S. and Tarkoma, S., 2025, *Large Language Model Based Multi-Agent System Augmented Complex Event Processing Pipeline for Internet of Multimedia Things* [Online]. Available at: https://arxiv.org/abs/2501.00906 [Accessed: 16 March 2025].

Zhang, M. et al., 2023. *How Language Model Hallucinations Can Snowball*,

APPENDIXES