Retrieval Augmented Generation - MRIWA

CITS3200 Team 13

August 2023

1 Acronyms and Abbreviations

- RAG: Retrieval Augmented Generation
- LLM: Large Language Model
- UWA: University of Western Australia
- MRIWA: Minerals Research Institute of Western Australia

2 Aim

The aim of this project is to make it easier for members of the MRIWA to search through and find the information they need from all of the PDFs and documents that are stored in their SharePoint. The PDFs currently stored can be hundreds of pages long and trying to find just a small amount of useful information can be quite challenging. We intend to create a Chatbot-like Webpage where users can query the database and have it retrieve the relevant documents for the user and give useful references as to which documents and where inside those documents the information can be found.

The main challenge that must be overcome is the security and assurance that the documents will not be made available to outside users or attackers. We must limit the use of sending any documents over the internet, specifically when sending them to the LLM to answer the queries.

3 Base Case

If a member of the MRIWA wants to find a piece of information that they know is inside a report or document that they store, they will first have to find which document they are looking for. This can either be done by searching the documents for certain keywords, which they can use to narrow down the number of documents which they need to look through, though it is still up to the user to look through the document and find which section holds the information that they need. The alternative and much more tedious approach would be to simply check each document which they believe might hold the relevant information and pray that the naming scheme and filing of the documents is sufficient.

This current solution has some very obvious problems and makes it very difficult to find any specific piece of information which a user might require. It may take a very long time for them to be able to search through all the long documents available, and there is no guarantee that what they are looking for can even be found in the documents.

4 Proposed Solution

The proposed solution is to create an online Webpage that MRIWA users can use to efficiently find any piece of specific information that they require from the available documents. This Webpage will ideally directly integrate with the SharePoint from which all the documents are stored, and will allow users to interact with the RAG using a Chatbot like format. The users will query the Chatbot, likely in the form of a question, about what information they intend to find, and the Chatbot will return an answer to the users question, plus which documents and where inside the documents it used to find its answer.

5 Functionality

Our Website will be split up into 2 main pages: the login page and the main Chatbot page. Proposed screenshots of these pages can be found here.

5.1 Login

Users will log in to their account using an email and password. There will also be a section where users can register an account.

5.2 Chatbot

This will be the main page that users use. Here they will be able to enter their queries in a dialog box and see the conversation in a big chat window. If users wish to see their previous conversations with the Chatbot they can use the menu bar on the left side. This will provide all their conversations in the form of a list which users can then choose to continue the conversations or delete them.

6 System Design

There are a number of factors to consider when deciding a stack, shown in the following table:

Factor	Priority	Explanation
Project requirements	Highest	The stack must be capable of fulfilling all requirements.
Continued development	High	The stack must use familiar technologies that are well-known and industry-standard.
Skills and backgrounds	High	Due to the short time frame, picking technologies where all developers are comfortable is important.
Languages' ecosystems	High	The language must have libraries capable of fulfilling the requirements.
Guides and documentation	Medium	The existence of detailed guides and documentation is important, and directly relates to using familiar technology.
Library ease-of-use	Medium	The libraries chosen should have a good developer experience and interface well with one another.
Learning curve	Low	Whilst desirable, this is unavoidable and hard to evaluate as members do not have prior experience which addresses all aspects of the project's requirements.

The project will be built as a web application. It will be organised into a separate frontend and backend, as it will allow for greater portability if they decide to integrate it with their existing services, or build alternative frontends. In summary, the application will use the following technologies.

- Backend: Django
  - Postgres and FAISS
- Frontend: React (Next.js)
- Deployment: DigitalOcean (to be confirmed)

6.1 Backend — Django

The project requires interfacing with Large Language Models (LLMs) to receive responses, which is why it is most important to consider. The most prominent of these frameworks are **Llamaindex** and **Langchain**. A number of observations were made during research, summarised below:

1. The vast majority of machine learning work is done in Python
2. Most libraries are primarily written in Python, and sometimes have ports or bindings
3. Most tutorials and guides use their Python variants (if they exist)

Therefore, it is proposed to build the backend in Python. Of the options, Django and Flask are the most popular and industry-standard, which was most desired as the client expressed interest in continued development after our involvement. This has the additional benefit of giving the client the freedom to change to one of the many LLMs present in the Python ecosystem without rewrites.

6.1.1 Database

This project will interact with up to two different databases. The first is for basic user logins and chat history, and an optional one for storing and interfacing with the LLM's embeddings.

For the main database, there are two types suitable:

1. Relational (Postgres, MySQL)
2. Document (MongoDB)

We propose to use **Postgres**, for its structural rigidity to maximise data security over the flexibility and ease-of-development of MongoDB.

Though a dedicated embeddings database is strictly not required, considering the documents can be several hundred pages, queries will have significantly reduced processing times at a fraction of the workload compared to a traditional database. Currently, this cannot be accurately estimated due to the many different factors that go into calculating performance. The most popular options include:

- Chroma (local) MIT Licensed
- FAISS (local) Apache 2.0 Licensed
- Pinecone (cloud-hosted)

From these options, will be using FAISS for the database. There is little difference between **Chroma** and **FAISS**. **Facebook AI Similarity Search (Faiss)** is a library for efficient similarity search and clustering of dense vectors. It contains algorithms that search in sets of vectors of any size, up to ones that possibly do not fit in RAM. It also contains supporting code for evaluation and parameter tuning. **Pinecone**, however, is a cloud-hosted service which carries additional cost and privacy risks, which should be mitigated as much as possible in the solution.

6.2 Frontend — React (Next.js)

We propose to use **React** – specifically, the **Next.js** framework. Despite Django being a full-stack framework, the JavaScript and React ecosystem is quite vast and offers many advantages over Django. Additionally, one of our group members has extensive experience working with modern framework, and the rest of the team has expressed interest in learning these technologies for future career prospects. Next.js offers many features over vanilla React, including Server-side Rending (SSR), Search Engine Optimisation (SEO) and file-based routing. Whilst Typescript would be desirable, we believe that the time lost from overcoming the learning curve is too great for the given time-frame.

The frontend will also utilise additional libraries which will greatly aid in development, which included but are not limited to:

- Styling: **TailwindCSS**
- State Management: **Zustand**

6.3 Deployment

The app will be deployed on a DigitalOcean instance, but this is subject to change based on the final model used.

6.4 LLM Application Framework

In selecting the ideal open-source framework for our LLM application, we propose a fusion of two powerful options: **LangChain** and **LlamaIndex**. Both frameworks offer user-friendly interfaces and thorough documentation, streamlining the learning process. LangChain excels in language processing, while **LlamaIndex** builds upon **LangChain**'s foundation to deliver advanced functionality. This combined approach ensures an efficient and capable framework for our application's success. Both **LangChain** and **LlamaIndex** are released under the permissive MIT License, ensuring both personal and commercial use as *MIT license only requires the inclusion of the original copyright notice and disclaims any liability from the software's creators.*

6.5 Models

6.5.1 Large Language Model (LLM)

The main challenge of choosing an LLM are the security risks. It is a clear risk that if an online LLM is used then they must not be able to access, store or share the documents that are sent along with each query. To undermine these threats, it is preferred that a local LLM is run in tandem with the Backend web server through which all the queries are passed, thus eliminating the need to send any documents over the web. For this reason, we will be conducting extensive research into the options for available, downloadable LLMs, and reach a conclusion as to which will be the most viable. Many factors will need to be taken into account such as the (file) size of the model, the hardware that is required to run it, the speed at which it runs, and the correctness of the answers it returns. If there are any severe issues found with running an LLM locally, then there may need to be a discussion about the viability of using an online LLM with very careful consideration as to how the before mentioned risks can be best mitigated.

6.5.2 Embedding / Transformer-Based Models

Embedding models excel in simplifying data representation by mapping items to continuous vectors. On the other hand, transformer-based models, with their contextual understanding, are adept at capturing intricate relationships within the data.

In the context of the LangChain and LlamaIndex frameworks, we recommend leveraging the provided or supported models for vector search. Opting for the models offered by these frameworks ensures compatibility and optimization within the ecosystem. It's essential that these models are open source and adhere to licenses such as MIT or Apache 2.0, allowing for seamless integration and confident use within our project.

7 Requirements

7.1 Functional Requirements

7.1.1 User Authentication

Identifier	Name	Description
FR1.1	Register	Users must be able to register an account with email verification.
FR1.2	Login	Users must be able to log in using their registered email and password.
FR1.3	Forgotten Password	Forgotten password functionality should be provided, allowing users to reset their password through email verification.

7.1.2 Chatbot Interaction

Identifier	Name	Description
FR2.1	Question	Users should be able to input queries in natural language.
FR2.2	Answer	The Chatbot should return relevant information from the PDF documents.
FR2.3	Citation	The Chatbot should cite the specific PDF documents, and the location within the document, from which the information was retrieved.

7.1.3 Document Retrieval

Identifier	Name	Description
FR3.1	Integration	Direct integration with SharePoint for PDF document retrieval.
FR3.2	Retrieve Documents	The system should return relevant documents/snippets based on user queries.

7.1.4 User History

Identifier	Name	Description
FR4.1	History Query	Users should be able to view their previous queries.
FR4.2	Manage Query	Users should have the option to resume, archive, or delete past conversations.

7.2 Non-Functional Requirements

7.2.1 Performance

Identifier	Name	Description
NFR1.1	Defined Time	The system should return Chatbot responses in a defined time frame (TBC).
NFR1.2	Simultaneous Usage	The system should support simultaneous user interactions without degradation in performance.

7.2.2 Security

Identifier	Name	Description
NFR2.1	Encrypted User Data	All user data, including login credentials and chat histories, should be encrypted.
NFR2.2	Secure Environment	Documents should never be transmitted outside the secure environment, especially to a public LLM.

7.2.3 Usability

Identifier	Name	Description
NFR3.1	User-friendly	The interface should be intuitive and user-friendly.
NFR3.2	User guidance	Provide tooltips or help sections for user guidance.

7.2.4 Maintainability

Identifier	Name	Description
NFR4.1	Clear Code	The codebase should be well-documented, formatted and linted to facilitate future updates and maintenance.

7.2.5 Compatibility

Identifier	Name	Description
NFR5.1	Integration	It can interface with MRIWA's SharePoint, and use the PDF files as context.
NFR5.2	Responsive Design	The application design supports both desktop and mobile views.

7.2.6 Data Integrity

Identifier	Name	Description
NFR6.1	Accuracy	The information retrieved from the documents must remain unaltered and accurate.

7.2.7 Training and Documentation

Identifier	Name	Description
NFR7.1	Accessibility	User manuals (if any) should be easily accessible and distributed internally via PDF.

8 Risk Register

The risk register aims to include as many plausible dangers as possible. With that, it includes a variety of mitigation strategies for each risk. As a team, we have endeavoured to include as many of these mitigation strategies in the form of either functional or non-functional requirements (previously stated). Strategies that are not included in our list of requirements shall be viewed as helpful strategies for MRIWA and future development teams on this project. An example is having training sessions for employees utilising the systems – something that our group cannot satisfy, but a suggestion nonetheless, and one that can be informed by user manuals/resources which our team can provide.

RISK RATING					
Likelihood	Consequence				
	Low	Minor	Moderate	Major	Extreme
Rare	Low (L1)	Low (L4)	Minor (Mi5)	Moderate (Mo4)	Major (Maj3)
Unlikely	Low (L2)	Low (L5)	Minor (Mi6)	Moderate (Mo5)	Major (Maj4)
Possible	Low (L3)	Minor (Mi3)	Moderate (Mo2)	Major (Maj2)	Extreme (E3)
Likely	Minor (Mi1)	Minor (Mi4)	Moderate (Mo3)	Extreme (E1)	Extreme (E4)
Almost Certain	Minor (Mi2)	Moderate (Mo1)	Major (Maj1)	Extreme (E2)	Extreme (E5)

The E2, L1, Maj4 etc are there to differentiate degrees of a certain risk rating. eg, E5 is a 'worse' Extreme rating that E1.

	Uncontrolled				Controlled		
Potential Danger (Provide detail)	**Likelihood**	**Consequence**	**Risk Rating**	**Controls = Actions which can lower the Risk**	**Likelihood**	**Consequence**	**New Risk Rating**
Data Breach (Unintended release or access to client's internal PDFs)	Unlikely	Extreme	Major (Maj4)	- Encrypted connection for data transfers - Only utilize vetted open-source tools - Periodically review security updates	Rare	Moderate	Minor (Mi5)
Inaccurate Retrieval (System not fetching relevant data from PDFs)	Possible	Major	Major (Maj2)	- Regular system tests using competency questions - Devise further testing criterion - Research	Unlikely	Moderate	Minor (Mi6)
Loss of Data during Processing	Rare	Extreme	Major (Maj3)	- Regular SharePoint back-ups - Data integrity checks when moving/processing data	Rare	Moderate	Minor (Mi5)
Incompatibility Issues with Open-Source Tool(s)	Unlikely	Moderate	Minor (Mi6)	- Test tools for compatibility with client systems before full deployment - Have alternatives in place	Rare	Moderate	Minor (Mi5)
Poor System Performance (slow response times)	Possible	Moderate	Moderate (Mo2)	- Utilise metrics and regularly measure - Optimise data processing and retrieval processes	Unlikely	Minor	Low (L5)
Inadequate Access Control (unauthorised personnel accessing the system)	Possible	Major	Major (Maj2)	- Role-based access control - Authentication system - Access logging/tracking activity (added benefit for later debugging)	Unlikely	Major	Moderate (Mo5)
Over-reliance on Open-Source Tools (tools might become deprecated or unsupported)	Possible	Moderate	Moderate (Mo2)	- Regularly review and update open-source tools - Have plans/backup tools in place	Rare	Moderate	Minor (Mi5)
Insufficient Logging and Monitoring (hinders tracing issues or user behaviour)	Unlikely	Moderate	Minor (Mi6)	- Implement comprehensive logic and monitoring from the outset - Possibly utilise tools that offer real-time monitoring	Rare	Moderate	Minor (Mi5)
Scalability Concerns (system cannot handle more users/queries than anticipated)	Possible	Major	Major (Maj2)	Design system with scalability in mind, and well documented to make future scaling smoother	Possible	Minor	Minor (Mi3)
Inadequate User Training (users unfamiliar with querying the system)	Likely	Moderate	Moderate (Mo3)	-User manuals and help resources -Training sessions	Rare	Minor	Low (L4)
Inefficient Error Handling (leads to crashes or unclear messages)	Possible	Moderate	Moderate (Mo2)	- Implement comprehensive error handling - Test and refine based on feedback and competency tests	Rare	Minor	Low (L4)
Unanticipated Costs	Possible	Moderate	Moderate (Mo2)	- Budget reviews - Allocate a contingency budget	Unlikely	Minor	Low (L5)
Data Storage Limitations	Possible	Major	Major (Maj2)	- Monitor storage usage - Use scalable storage solutions	Unlikely	Major	Moderate (Mo5)
Security Vulnerabilities (in third-party tools)	Unlikely	Extreme	Major (Maj4)	- Regularly update libraries - Vulnerability scanners - Security audits	Rare	Major	Moderate (Mo4)

9 User Acceptance Tests

The client has provided 7 documents, each spanning several hundred pages for development and a series of competency questions for testing. A compliant solution will be capable of answering all competency questions with reasonable accuracy, performed through the app without visual bugs. A fully-complete solution will be capable of fulfilling all functional (FR) and non-functional (NFR) requirements, but is unlikely in the given the time frame. The following are the provided competency questions:

1. Confirm an acknowledgment to MRIWA (or its predecessors) funding is given in the reports:
   (a) identify which reports reference MERIWA or MRIWA?
   (b) extract all references to MERIWA and MRIWA from the reports
2. Search by periodic table elements or full name:
   (a) identify any references to nickel or Ni in the reports?
   (b) Which elements are considered in the reports?
3. Search for organisations by variations in name:
   (a) which reports has Commonwealth Scientific Industrial Research Organisation been involved with:
      i. in any capacity (including being listed in references)?
      ii. as researcher?
      iii. as a sponsor?
4. Ability to search on geographic locations:
   (a) Which report is relates to the East Kimberley region?
   (b) Which regions of Western Australia are referenced in the reports?
5. Ability to differentiate authors:
   (a) Which author has been involved in more than one project?
6. Ability to aggregate/integrate information:
   (a) What is the average number of references in each report?
   (b) Which reports relate to leaching?
   (c) Which reports relate to exploration?
   (d) Which reports relate to mining extraction?
   (e) Which reports relate to mineral processing?
   (f) Which elements are considered in the reports?

User Stories

10 User Management

10.1 User Login

User Story(General User)	Acceptance Criteria
As a general user, I want to be able to log in to the application using my credentials, ensuring secure access to my account.	• The login page provides input fields for username and password. • Upon successful login, the user is directed to their personalized dashboard.
User Story(Administrator)	Acceptance Criteria
As a general user, I want to be able to log in to the application using my credentials, ensuring secure access to my account.	• The login page includes a designated section for administrators to input their credentials. • After logging in, administrators are directed to an admin dashboard with additional functionalities.

10.2 Remember Me Option

User Story(General User)	Acceptance Criteria
As a general user, I want the option to have the application remember my login credentials for convenient access.	• A "Remember Me" checkbox is present on the login page. • When checked, the application stores the user's login credentials securely for future sessions.

10.3 User Registration

User Story(General User)	Acceptance Criteria
As a general user, I want to be able to create a new account by registering with a valid email address and password.	• The registration page provides fields for entering an email address and password. • Users receive a verification email upon successful registration.
User Story(Administrator)	Acceptance Criteria
As an administrator, I want a separate registration process that allows me to create an administrator account with extra privileges.	• The administrator registration page includes additional fields to gather information relevant to administrators. • Upon successful registration, administrators gain access to the admin dashboard.

10.4 Password Strength Indicator

User Story(General User)	Acceptance Criteria
As a general user, I want the registration page to include a password strength indicator to ensure the security of my account.	• The registration page features a visual indicator that rates the strength of the entered password (e.g., weak, medium, strong). • Users receive immediate feedback on the security level of their chosen password.

11 User Interface

11.1 General - User stories

11.1.1 Dashboard Overview

User Story	Acceptance Criteria
As a user, I want to view a user-friendly dashboard upon logging in, providing a clear overview of available features and options.	• The dashboard presents an organized layout with clear labels for available features. • Users can quickly grasp the app's capabilities upon login. • Different features will be separated by containers, padding, margins, changes in colour and other aesthetic choices.

11.1.2 Efficient Navigation

User Story	Acceptance Criteria
As a user, I want the application to have an efficient navigation menu for easy movement between sections.	• A persistent navigation menu or sidebar enables smooth 'Single Page App'-like navigation between different sections. • Users switch between features effortlessly.

11.1.3 Clear Error Messages

User Story	Acceptance Criteria
As a user, I want the application to display clear and informative error messages.	• Error messages are clear, concise, and provide guidance for resolution. • Users can troubleshoot issues effectively. • Clean error widget appears up the top.

11.2 Query Q&A Section

11.2.1 Competency Question Submission

User Story	Acceptance Criteria
As a user, I want to have an easy and intuitive interface for submitting competency questions, guiding me through the process seamlessly.	• The submission process is intuitive, guiding users through question input. • Users experience a seamless interaction without confusion. • The user interface includes a text input field for entering competency questions.

11.2.2 Immediate Feedback

User Story	Acceptance Criteria
As a user, I want to receive immediate feedback upon submitting a competency question, ensuring that my query has been processed.	• After submitting a question, the interface promptly displays a confirmation message. • Users receive assurance that their query has been successfully received.

11.2.3 Clickable Hyperlinks

User Story	Acceptance Criteria
As a user, I want the ability to access the source document(s) directly from the application's answers.	• Each answer includes a clickable hyperlink leading directly to the source document(s) for further context. • Clicking on the hyperlink opens the corresponding document in a new browser window or tab. • The linked documents open securely, ensuring user privacy and data protection.

11.2.4 Enhanced User Experience with Spell Correction

User Story	Acceptance Criteria
As a user, I want the application to understand and interpret my questions accurately, even if they contain spelling mistakes or typos.	• The application employs a spell correction mechanism that analyses user-input questions for potential spelling errors. • When a question contains misspelled words, the application offers suggestions for corrected spellings, aiding users in clarifying their queries.

11.3 Administrator Features User Stories

11.3.1 Upload Documents (Administrator)

User Story(Administrator)	Acceptance Criteria
As an administrator, I want the ability to upload new documents to the application's database for processing.	• The administrator dashboard includes an "Upload Documents" section. • Administrators can select and upload PDF documents from their local devices. • Uploaded documents are processed using the search and extraction frameworks. • Successful document processing notifications are displayed to administrators.

11.3.2 Document Management (Administrator)

User Story(Administrator)	Acceptance Criteria
As an administrator, I want to manage the documents uploaded to the database.	• The administrator dashboard includes a "Manage Documents" section. • Administrators can view a list of uploaded documents and their status. • Options to delete or update documents are available to administrators.

11.3.3 Document Processing (Administrator)

User Story(Administrator)	Acceptance Criteria
As an Administrator, I want the ability to upload new documents for processing. This will ensure that the application remains up-to-date with relevant information.	• Add an "Upload Documents" option accessible from the user dashboard. • Allow users to select and upload PDF documents from their local devices. • Display a progress indicator during the document upload process. • Process the uploaded documents using the search and information extraction frameworks. • Notify users upon successful document processing completion. • Ensure uploaded documents are searchable and can be referenced in answers.

12 User Experience (UX) - user stories

User Story	Acceptance Criteria
As a user, I want to see answers in comprehensive English sentences for better understanding.	• The application generates coherent and contextually relevant English sentences as answers. • The answers incorporate the extracted data from documents in a structured manner.

User Story	Acceptance Criteria
As a user, I want the ability to access the source document(s) directly from the application's answers.	• Each generated answer includes a clickable hyperlink to the specific source document(s) that the information was extracted from. • Clicking on the hyperlink opens the corresponding document in a new browser window or tab. • The hyperlink text provides context, such as the document title or relevant keywords, to indicate the content of the linked document. • The linked documents open securely, ensuring user privacy and data protection.

User Story	Acceptance Criteria
As a user, I want the application's interface to work well on both desktop and mobile devices.	• The interface adapts seamlessly to various devices, ensuring consistent usability. • Users can access and interact with the app on mobile and desktop.

13 Data Processing Data Storage

User Story	Acceptance Criteria
As a user, I want the ability to access the source document(s) directly from the application's answers.	• Each generated answer includes a clickable hyperlink to the specific source document(s) that the information was extracted from. • Clicking on the hyperlink opens the corresponding document in a new browser window or tab. • The hyperlink text provides context, such as the document title or relevant keywords, to indicate the content of the linked document. • The linked documents open securely, ensuring user privacy and data protection.
User Story(client)	**Acceptance Criteria**
As a client, I want assurance that my confidential documents and intellectual property are secure and inaccessible to any AI tool or search extraction framework like Haystack.	• The application's architecture includes robust security measures to protect confidential documents and intellectual property from unauthorized access. • Access controls are implemented to restrict access to authorised users only, preventing any unauthorised personnel from viewing or extracting data. • Documents designated as confidential or containing sensitive intellectual property are encrypted and stored securely in a way that prevents any direct exposure to external tools.

14 Skills and Resources Audit

14.1 Project Requirement Analysis

- **Backend Development:** Integration with Large Language Models, Django development, and Database management.
- **Frontend Development:** React (specifically Next.js) application development.
- **Deployment:** Setting up and managing a DigitalOcean instance.
- **LLM Application:** Integration with LangChain and LlamaIndex frameworks.
- **Model Management:** Handling of Large Language Models and Transformer-Based Models.

14.2 Skills Inventory

Task	**Required Skills**
Integration with LLMs	Python, Django, familiarity with LLM principles
Django Backend Development	Python, Django, Database design (Postgres)
React Development	JavaScript, React, Next.js
DigitalOcean Deployment	Server setup, Deployment strategies
Frameworks Integration	Knowledge of LangChain & LlamaIndex frameworks
Model Management	Deep understanding of LLMs, Transformer-based models

14.3 Resources Inventory

Task	**Required Resources**
Backend Database	PostgreSQL, optional dedicated embeddings database
Frontend Development	Node.js, npm, Next.js, TailwindCSS, Zustand
Deployment	DigitalOcean instance
LLM Frameworks	LangChain, LlamaIndex

14.4 Team's Current Capabilities

Skill/Resource	**Status**
DigitalOcean Deployment	Red
LangChain & LlamaIndex knowhow	Dark orange
LLM principles understanding	Dark orange
TailwindCSS & Zustand	Orange
Django (Python) proficiency	Yellow
React & Next.js familiarity	Green
Postgres	Green

14.5 Action Plan

Action Item	Action
React & Django	Develop a comprehensive to-do list application. Dylan will provide guidance and instruction on React framework implementation.
LLM	Conduct in-depth research on relevant topics. Refer to guides, tutorials, and scholarly resources to gain a comprehensive understanding.
Deployment	Engage in further discussions with the client to finalize deployment plans and ensure alignment with project requirements.
Langchain & Llamaindex	Monitor the current status of Langchain and Llamaindex components. Verify that they are functioning optimally and troubleshoot any issues as needed.
Postgres	Conduct thorough research on Postgres database management. Study relevant documentation and resources to enhance database proficiency.
Tailwind & Zustand	Carefully review the documentation for Tailwind CSS and Zustand state management. Seek assistance from Dylan to understand and utilize these tools effectively.