SE 4485: Software Engineering Projects

Fall 2024

Detailed Design Documentation

|  |  |
| --- | --- |
| Group Number | 3 |
| Project Title | Knowledge Management Assistant (Team B) |
| Sponsoring Company | The Fellows Consulting Group (FCG) |
| Sponsor(s) | Jeff Buchmiller |
| Students | 1. Blythe Williams 2. Nidhi Prakuzhy 3. Roj Pawig 4. Ashley Primrose 5. Humayl Sheryar 6. Dalton Brua |

## ABSTRACT

This document outlines the detailed design of the Knowledge Management Assistant Application. It includes GUI designs, static and dynamic models, and traceability mapping requirements to the design elements. The document also provides evidence of configuration management. By documenting these aspects, the application aims to ensure clarity and traceability to best practices in software engineering.

## TABLE OF CONTENTS

[ABSTRACT 2](#_Toc181388821)

[TABLE OF CONTENTS 2](#_Toc181388822)

[LIST OF FIGURES 2](#_Toc181388823)

[LIST OF TABLES 2](#_Toc181388824)

[INTRODUCTION 2](#_Toc181388825)

[GUI (Graphical User Interface) Design 3](#_Toc181388826)

[STATIC MODEL - CLASS DIAGRAMS 4](#_Toc181388827)

[DYNAMIC MODEL - SEQUENCE DIAGRAMS 4](#_Toc181388828)

[RATIONALE FOR YOUR DETAILED DESIGN MODEL 7](#_Toc181388829)

[TRACEABILITY FROM REQUIREMENTS TO DETAILED DESIGN MODEL 8](#_Toc181388830)

[EVIDENCE THE DESIGN MODEL HAS BEEN PLACED UNDER CONFIGURATION MANAGEMENT 10](#_Toc181388831)

[ENGINEERING STANDARDS AND MULTIPLE CONSTRAINTS 10](#_Toc181388832)

[ADDITIONAL REFERENCES 10](#_Toc181388833)

## LIST OF FIGURES

Figures 1.1-1.3 GUI Design

Figure 1.4 Class Diagram

Figure 1.5 Query Submission Sequence Diagram

Figure 1.6 Admin Stored Query Deletion Sequence Diagram

Figure 1.7 Goes through history Sequence Diagram

Figure 1.8 User option to change password Sequence Diagram

Figure 1.9 User chooses to save response Sequence Diagram

Figure 2.0 User chooses to discard response Sequence Diagram

## LIST OF TABLES

Table 1.1 Traceability of Requirements

## INTRODUCTION

This document aims to offer a comprehensive overview of the design components, including GUI interface, static and dynamic models, traceability from requirements to the design model, and evidence of configuration management. The document is structured to guide developers and stakeholders through the technical design choices to ensure the project aligns with engineering standards and multiple constraints.

The purpose of this document is to outline the detailed design for the application. Focusing on how each requirement has been addressed and implemented in the design. The scope includes detailed descriptions of the GUI, class diagrams, sequence diagrams, and the traceability from requirements to the design model.

The structure of the documents is as follows: The introduction, which will provide an overview of the document, including purpose, scope and structure. The GUI design, which will provide screen designs created using tools such as Figma. The static model class diagrams, which will provide the structure of the system. The dynamic model sequence diagrams, which will provide sequence diagrams representing the dynamic behavior of the system in various use cases. The traceability, which will map each requirement to the corresponding design element. The evidence of configuration management, which will demonstrate that the design model has been placed under configuration management. Lastly, the standards and additional references, which will ensure that the design adheres to IEEE standards and lists the relevant literature and standards that are referenced in the design process.

## GUI (Graphical User Interface) Design

* screen designs (coded or using drawing tool) ~ Figma

A login screen with blue and white lines and dots

Description automatically generatedA screenshot of a chat

Description automatically generated

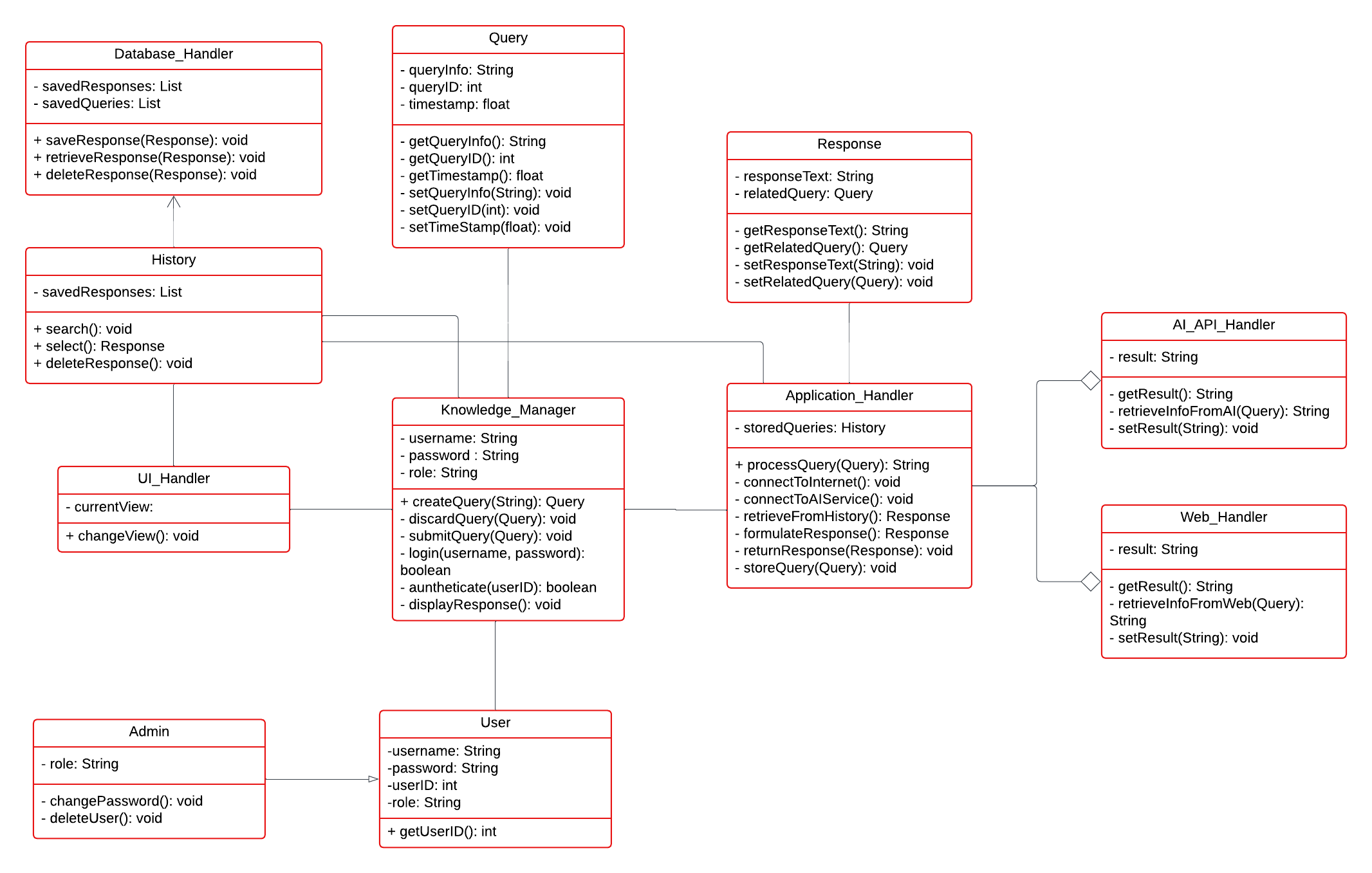
A screenshot of a search box

Description automatically generated

Figure 1.1-1.3 GUI Design

## STATIC MODEL - CLASS DIAGRAMS

* captured in Rose (other tools are also allowed): LucidChart

 Figure 1.4 Class Diagram

## DYNAMIC MODEL - SEQUENCE DIAGRAMS

* captured in Rose (other tools are also allowed): LucidChart

Query Submission Sequence Diagram:

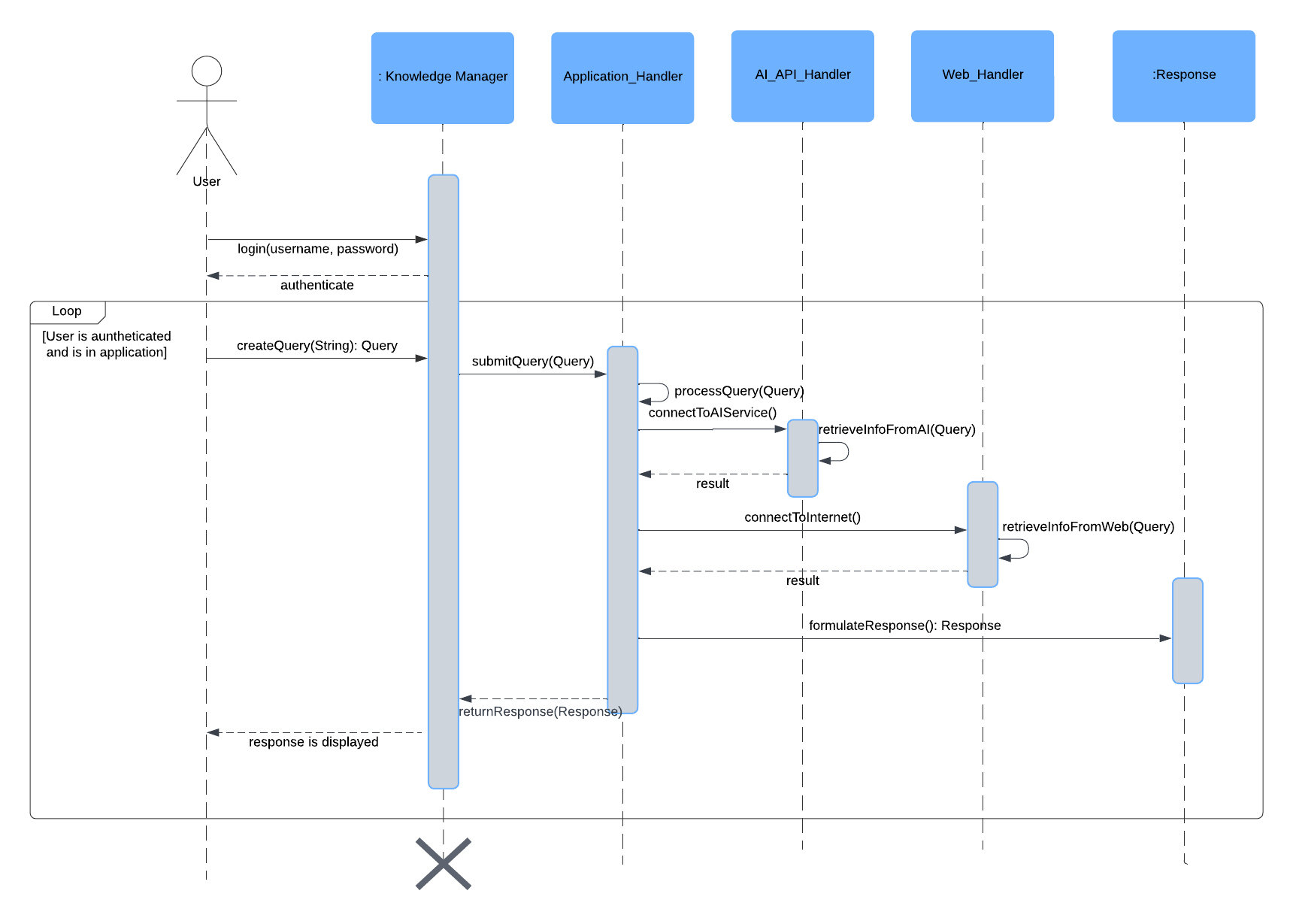


Figure 1.5 Query Submission Sequence Diagram

Admin Stored Query Deletion Sequence Diagram:

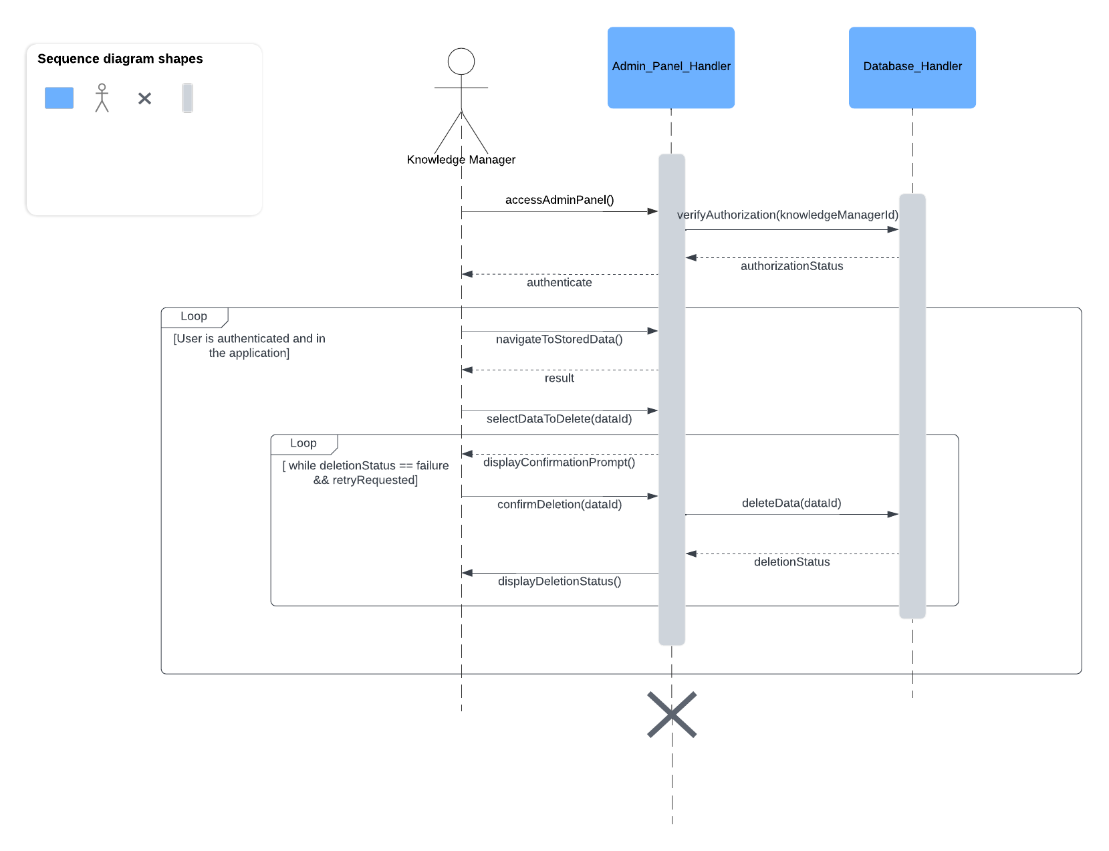


Figure 1.6 Admin Stored Query Deletion Sequence Diagram

Goes through history sequence diagram:

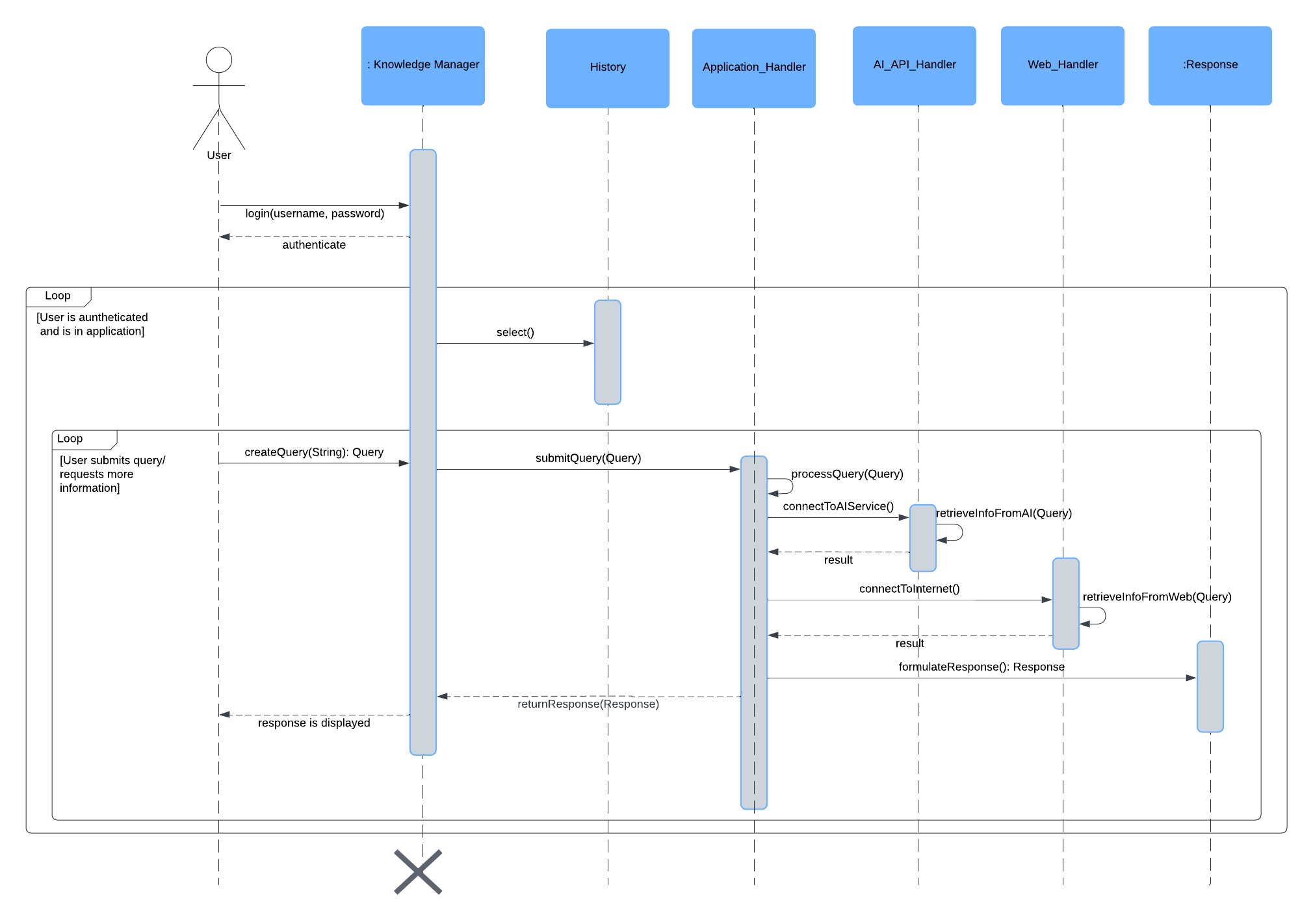


Figure 1.7 Goes through history Sequence Diagram

Users option to change password sequence diagram:

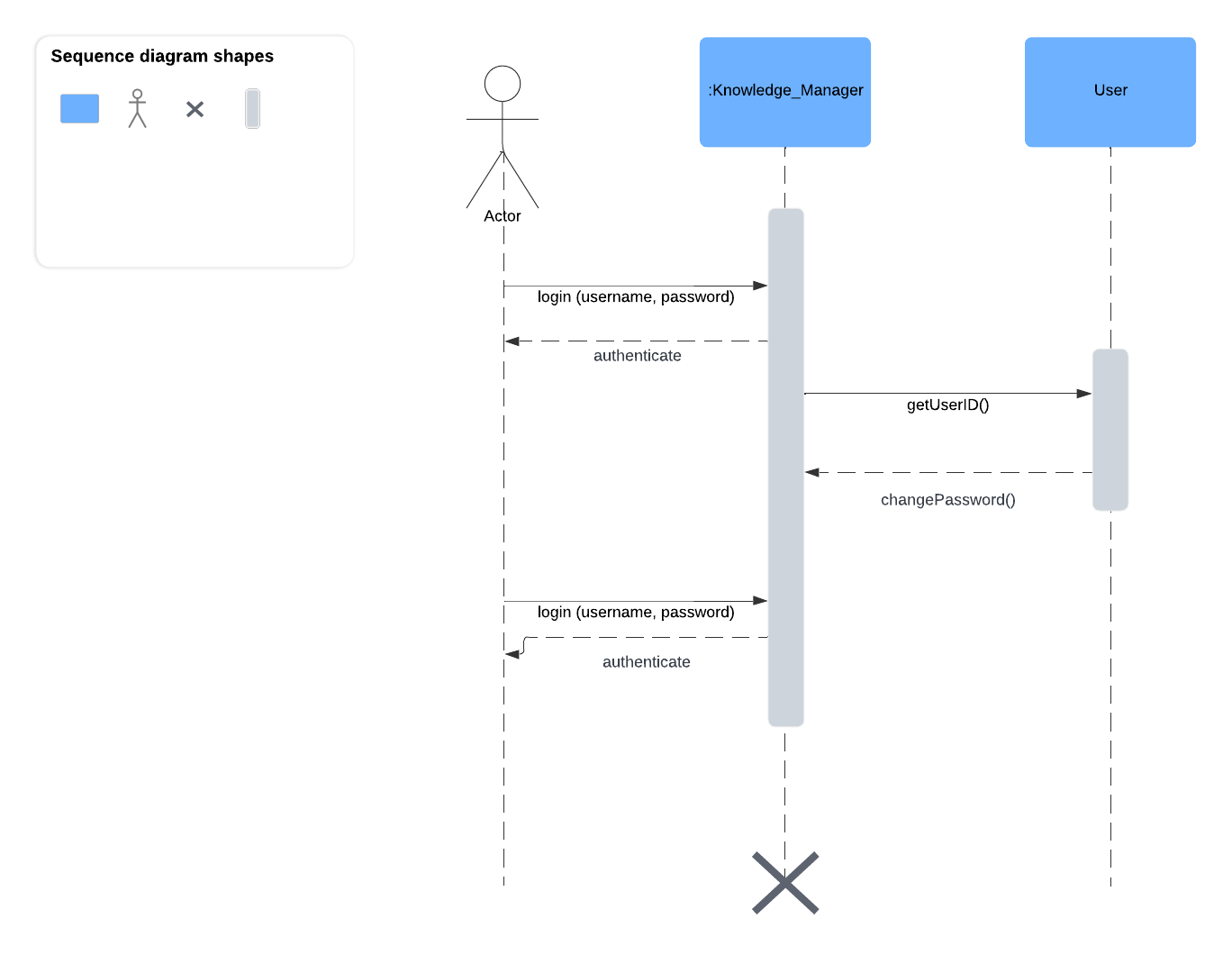


Figure 1.8 User option to change password Sequence Diagram

User chooses to save result sequence diagram:

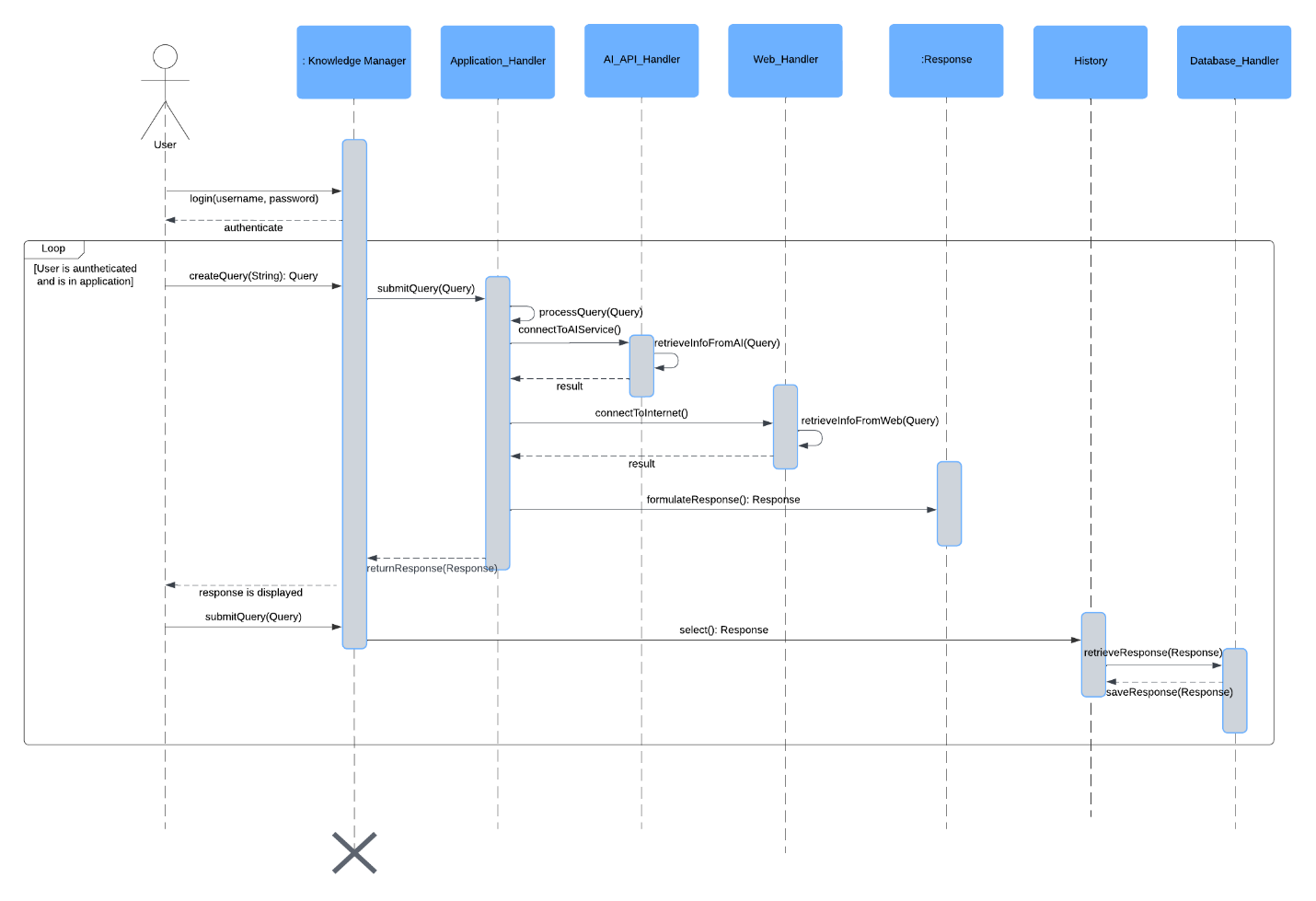


Figure 1.9 User chooses to save response Sequence Diagram

User chooses to discard result sequence diagram:

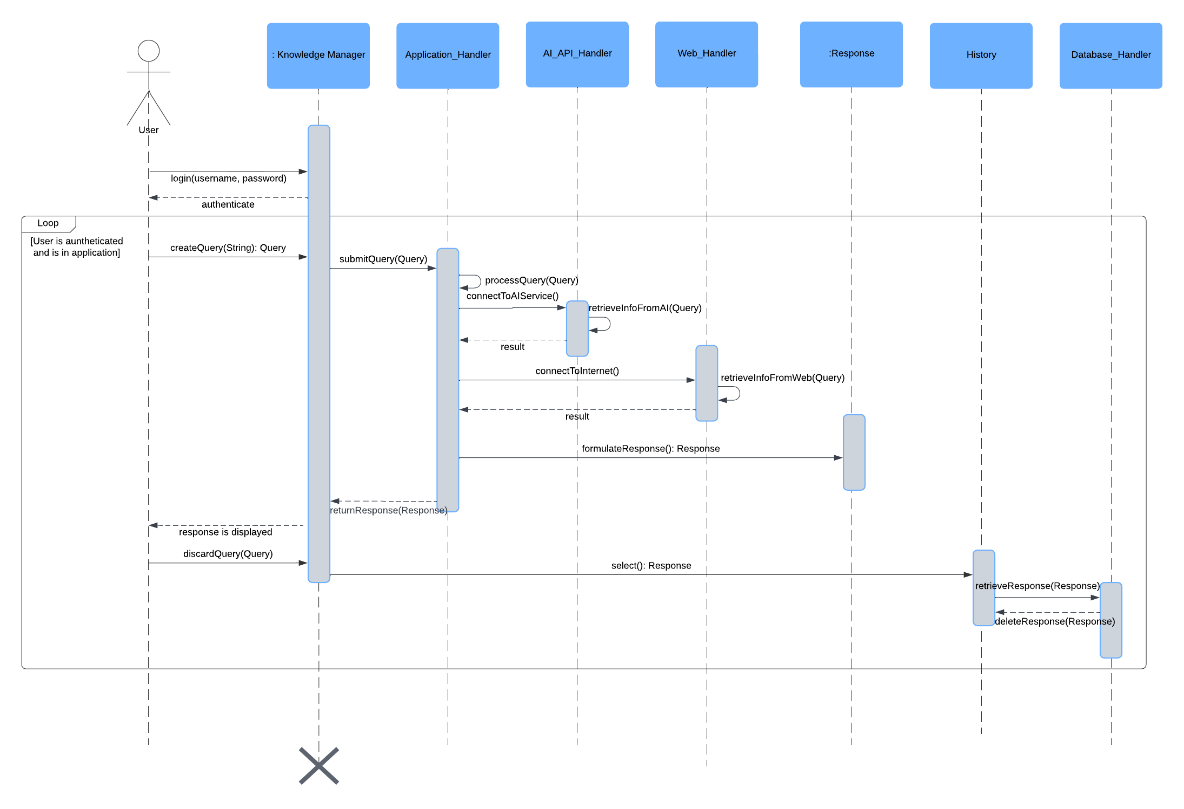


Figure 2.0 User chooses to discard response Sequence Diagram

## RATIONALE FOR YOUR DETAILED DESIGN MODEL

The detailed design model of the application has been developed to ensure a robust, scalable, and maintainable solution. The design decisions are driven by requirements, project constraints, and best practices in software engineering. Here are the key considerations that shaped the detailed design model:

1. **Alignment with Requirements:** The design model is closely aligned with the requirements specified in the requirements documentation. Each design element, whether it is a class in the static model, or an interaction in the dynamic model, is crafted to fulfill specific requirements. This alignment ensures that all functionalities promised to the stakeholders are implemented effectively.
2. **Modularity and Reusability:** The design promotes modularity by dividing the system into distinct classes and components with well defined responsibilities. This modular approach facilitates easier maintenance, testing, and future enhancement. Reusability is also a key focus, with components designed to be reused across different parts of the system. Which leads to a reduction in redundancy and development efforts.
3. **Scalability:** The design considers scalability to handle increasing loads and growing data sets. Components such as the Database\_Handler and Web\_Handler are designed to efficiently manage large volumes of queries and data. The use of robust architectural patterns ensures that the system can scale horizontally and vertically as needed.
4. **Performance and Efficiency:** The design emphasizes performance and efficiency by optimizing key operations and data flows. Sequence diagrams detail the interactions between components, ensuring that time-critical operations are streamlined. For example, the query submission sequence diagram illustrates efficient processing from user input to response generation.
5. **User Experience:** The GUI design is centered around user experience, ensuring that the interface is intuitive and easy to navigate. Screen designs created using tools such as Figma focus on usability, accessibility, and visual appeal. The UI\_Handler class is responsible for managing user interactions, providing a seamless experience.
6. **Security:** Security is a paramount consideration in the design, particularly for user authentication and data handling. The Knowledge Manager class includes mechanisms for secure login, password management, and user authentication checks. These security measures protect user data and ensure compliance with privacy standards.
7. **Traceability:** The design model includes a traceability matrix mapping requirements to design elements. This traceability ensures that each requirement is addressed and provides a clear reference for developers and stakeholders. It also facilitates easier validation and verification of the system against the specified requirements.
8. **Adherence to standards:** The design adheres to relevant engineering standards, including IEEE std 1016-1998 for software design documentation. This adherence ensures that the design follows industry best practices and meets quality benchmarks.

By considering these factors, the detailed design model aims to provide a comprehensive and effective solution for the Knowledge Management Assistant application. The model not only meets the specified requirements but also ensures scalability, maintainability, and user satisfaction.

## TRACEABILITY FROM REQUIREMENTS TO DETAILED DESIGN MODEL

* provide a mapping between requirements and detailed design model
* clearly describe how each requirement in the *Requirements Documentation* is captured in the design

|  |  |  |
| --- | --- | --- |
| REQUIREMENT | DESIGN | DESCRIPTION |
| Users must be able to input text. (UC01) | Knowledge Manager & UI\_Handler Classes | Users will be able to input text through the graphical user interface carried out by the UI\_Handler and Knowledge Manager Class. |
| K.M.A. must be able to process text. (UC01) | Knowledge Manager Class | The knowledge manager class will be responsible for processing text received from the user in order to create and submit a query. |
| K.M.A. must be able to gather information from the Web. (UC01) | Application\_Handler and Web\_Handler Classes | The Application\_Handler and Web\_Handler classes will connect to the Web and gather information from the web based on the query submitted by the Knowledge Manager. |
| K.M.A. must be able to analyze information gathered. (UC01) | Application\_Handler Class | The Application\_Handler class will analyze the data collected from the Web\_Handler and AI\_API\_Handler to provide relevant information to the Knowledge Manager. |
| K.M.A. must be able to output its response to users. (UC01) | UI\_Handler and Knowledge Manager Class | The Knowledge Manager class will compile and display the analyzed response to the user through the UI. |
| K.M.A. must be able to save user queries and responses to a database. (UC01) | Application\_Handler and Database\_Handler Classes | The Application\_Handler will manage queries and responses calling upon the Database\_Handler handle the storage of user queries and responses in a database for future reference. |
| Users must be able to access and view a history of queries and responses. (UC02) | UI\_Handler and History Classes | The UI\_Handler will allow users to access the History class to view previously saved queries and responses. |
| Users must be able to select previous queries and modify or re-submit to the application. (UC02) | Application\_Handler and History Classes | The Application\_Handler will allow modification and resubmission of past queries retrieved from the History class. |
| Users must be able to search through previous queries. (UC02) | UI\_Handler and History Classes | The UI\_Handler will provide a search functionality for users to look up previous queries stored in the History. |
| Users must be able to login. (UC03) | Knowledge Manager Class | The Knowledge Manager will manage user authentication and session handling to enable secure login. |
| Users must be able to change passwords. (UC04) | UI\_Handler and User Class | The UI\_Handler will facilitate the password change process, interacting with the User class to update credentials. |
| Users must be able to save queries. (UC05) | Application\_Handler Classes | The queries submitted by the user through the Knowledge\_Manager will be automatically saved by the Application\_Handler class upon formulating a response. |
| Users must be able to discard queries. (UC05) | Knowledge\_Manager and UI\_Handler Classes | The Knowledge Manager will manage the discarding of queries if users decide not to retain them. |
| Users must be able to delete stored results. (UC06) | UI\_Handler and History Classes | The UI\_Handler will allow users to delete past results stored within the History class. |
| K.M.A. must be able to authorize users. (UC06) | Knowledge\_Manager Class | The Knowledge Manager will handle user authorization checks to ensure access control for specific features, e.g. administrative privileges. |
| K.M.A. must be able to gather information from AI services. (UC07) | Application\_Handler and AI\_API\_Handler Classes | The Application\_Handler will interface with the AI\_API\_Handler to gather and process information from external AI services. |

Table 1.1 Traceability of Requirements

EVIDENCE THE DESIGN MODEL HAS BEEN PLACED UNDER CONFIGURATION MANAGEMENT

## ENGINEERING STANDARDS AND MULTIPLE CONSTRAINTS

* IEEE Std 1016-1998-(Revision-2009): Software Design

## ADDITIONAL REFERENCES

* Larman, C., 2012. Applying UML and Patterns: *An Introduction to Object Oriented Analysis and Design and Iterative Development. Pearson Education*
* Hyman, B., 1998*. Fundamentals of Engineering Design. New Jersey: Prentice Hall*
* Simon, H.A., 2014. *A Student's Introduction to Engineering Design: Pergamon Unified Engineering Series* (Vol. 21). Elsevier