**Software Requirements Specification**

**for**

**Book Managing Web Application**

**Version 1.0 approved**

**Prepared by:**

1. **Phạm Bá Nhật Khang - 2211460**
2. **Nguyễn Ngọc Duy - 2210522**
3. **Dương Quang Minh - 2212021**
4. **Nguyễn Minh Khánh - 2211523**
5. **Nguyễn Trương Thái Bảo - 2210251**
6. **Đinh Xuân Quyết - 2212854**
7. **Đặng Hoàng Khang - 2211422**

**Department of Software Engineering**

**Faculty of Computer Science and Engineering**

**Ho Chi Minh City University of Technology – VNU-HCM**

**03/10/2024**

**Table of Contents**

[**1. Task 1: Requirement elicitation (1.1, 1.2) 2**](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.o02ypkqv95wu)

[1.1 Domain Context 2](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.db4lxa1mcyfn)

[1.2 Stakeholders and Needs 2](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.7yrs7x6lpuv0)

[1.3 Benefits of the System 2](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.9q2rds5ai1qu)

[1.4 Functional Requirements 3](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.pnua2um90gfe)

[1.5 Non- Functional Requirements 4](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.8qd323lt3qdj)

[**2. Use-case Diagrams (1.3) 6**](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.5umbvplae92x)

[2.1 Use-case Diagram for the Whole System 6](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.plo5apn3mfcq)

[2.2 Use-case Diagram for Printing Module 7](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.8ysgvqz39dx0)

[2.3 The Details of Usecases in Printing Document Module 7](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.ei6uiuddhc25)

[1. Usecase Upload Document File 7](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.gj09c0a8hgfn)

[2. Usecase Re-upload 8](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.rabcja1cqpg1)

[3. Usecase Verify File Type 9](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.u8peiuovr6hw)

[4. Usecase Search For Printer 10](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.9h78uts82puo)

[5. Usecase Choose Printer 11](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.z8zvib2d0y5)

[6. Usecase Check Printer Status 12](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.erox23txdx4e)

[7. Usecase Modify Printing 13](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.bx5sraby8t3e)

[8. Usecase View Print Preview 14](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.4q9t8g5lyc1q)

[9. Usecase Buy Printing Pages 15](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.u2yaao4zxvuc)

[10. Usecase Submit Print Request 16](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.lhynd4r3dmbb)

[11. Usecase Check Account Balance 17](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.qcurxty9httf)

[12. Usecase Process Transaction 17](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.epf8hph8xif3)

[13. Usecase Save Printing Log 18](https://docs.google.com/document/d/1-GWAaV59_drevvm9DqQMEbtrQ2K12cqp/edit#heading=h.3s9uqfx1n26f)

# **Task 1: Requirement elicitation (1.1, 1.2)**

## **Domain Context**

For students, finding reference materials quickly is essential for their studies, but many face difficulties with traditional book management systems, especially in smaller bookstores or libraries. Complex processes for borrowing, returning, and accessing online materials can create delays, making studying less efficient. Small bookstores often have trouble managing large inventories, leading to disorganized systems that make it hard to keep track of books, loans, and returns, which reduces the quality of service. Additionally, as more people turn to digital books, many stores struggle to meet this demand due to outdated systems. Therefore, having a modern and easy-to-use book management system is important. It would not only help students access materials more easily but also improve how bookstores and libraries run, making the whole experience better for everyone.

The book management system is crucial for addressing the challenges faced by both libraries and bookstores. These systems serve as centralized platforms that streamline the organization and management of books, making it easier for users to locate, borrow, or purchase materials. By providing real-time tracking of inventory, borrowing statuses, and sales records, these systems enhance operational efficiency and improve user experiences. Libraries can offer an accessible catalog to support various academic and leisure needs, while bookstores can maintain accurate inventories to drive sales and customer satisfaction​.

* 1. **Stakeholder and Needs**

The book management system serves various stakeholders, each with distinct roles and needs that contribute to the overall effectiveness of the system. These stakeholders include library members and students who seek efficient access to materials, staff members responsible for managing daily operations, and system administrators who maintain the technology. Publishers and suppliers provide the necessary titles for libraries and bookstores, while management teams oversee operations to ensure sustainability. Additionally, authors and content creators benefit from increased visibility of their works, and government or regulatory bodies ensure compliance with relevant standards. Understanding the specific needs of each stakeholder is essential for optimizing the system’s functionality and improving the experience for all users.

**Reader - Library Members/Students**: Library members and students are primary users of the book management system, relying on it for easy access to reference materials essential for their studies. Their main needs include efficient search functionality to locate books quickly, as well as streamlined processes for borrowing and returning materials. By utilizing a modern system, students can enhance their learning experience, making studying more efficient and enjoyable.​

**Authors/Content Creators**: Authors and content creators also benefit indirectly from book management systems, as these platforms help ensure that their works are available and discoverable to readers. Their needs revolve around ensuring fair representation and accessibility of their books and articles. By improving inventory management and user engagement, a book management system enhances the likelihood that readers will find and access their content.

**Staff**: The staff working in libraries and bookstores play a critical role in managing day-to-day operations. Their relevance to the book management system lies in their need for tools that facilitate inventory management and customer service. A well-designed system enables staff to efficiently track loans, manage stock levels, and assist users effectively, ultimately improving the quality of service provided to library members and customers​.

**System Administrators/IT Support**: System administrators and IT support personnel are essential for maintaining the technology behind the book management system. They require reliable tools for monitoring system performance, managing user accounts, and ensuring data security. A centralized platform simplifies these tasks, allowing IT support to efficiently troubleshoot issues and ensure the system runs smoothly, thereby enhancing the user experience​.

**Publishers/Suppliers**: Publishers and suppliers are other crucial stakeholders in the book management ecosystem, providing the titles that libraries and bookstores need to meet user demands. Their needs include accurate data on sales and inventory levels to assess the performance of their publications. A book management system offers valuable insights into circulation trends, helping publishers make informed decisions about marketing and distribution strategies.

**Library/Bookstore Management**: The management teams of libraries and bookstores have a vested interest in the effective functioning of a book management system. They need comprehensive data and analytics to inform strategic decisions regarding resource allocation, staffing, and overall operational efficiency. By leveraging such insights, management can optimize services and ensure the sustainability of the institution​.

**Government/Regulatory Bodies**: Government and regulatory bodies have an interest in book management systems as they often set standards and guidelines for libraries and educational institutions. Their needs include ensuring compliance with copyright laws and promoting access to information. A robust book management system aids in maintaining compliance and fosters a culture of transparency and accountability within these organizations.

* 1. **Benefits Of The System**

A **book management system** serves as a digital solution for organizing and managing books in environments like libraries or bookstores. This system helps users locate, borrow, or purchase books, while staff can easily manage inventory and lending processes. In both libraries and bookstores, the system enables real-time tracking of book availability, borrowing statuses, and sales records. Whether for educational institutions or commercial businesses, a centralized system ensures efficient book management and enhances user experience.

Libraries aim to provide an organized and accessible catalog of books to support academic, leisure, or research needs. Meanwhile, bookstores focus on ensuring inventory is accurate and well-managed to drive sales and customer satisfaction. The system needs to handle various functions, including book borrowing and returning in libraries, and sales transactions in bookstores, while ensuring that the catalog is always up to date. In both cases, it serves to streamline operations and improve efficiency in managing large collections of books.

## **Functional Requirements**

Functional requirements concentrate on the relationships between the overall system and its users:

1. Library Members

* **Book Searching**: Search for books by title, author, genre, or ISBN and view books’ information.
* **Book Borrowing**: Borrow books and view borrowing history.
* **Notifications**: Receive notifications for due dates, overdue books, and reserved book availability.
* **Reviews and Ratings**: Write reviews and rate books.
* **View most popular books**: Can view most borrowed books in the library.
* **Event registration**: Can register to organize book events at the library..

1. Staff

* **Catalog Management**: Add, update, and remove books from the catalog.
* **Member Management**: Can see library member accounts’ information.
* **Circulation Management**: Check out and check in books.
* **Inventory Management**: Track book inventory and manage stock levels.
* **Reporting**: Generate reports on book circulation, member activity, and inventory status.

1. Publishers, Books Suppliers

* **Catalog Submission**: Submit new book titles and updates to the staff to consider if they should update the library catalog.
* **Submission State View**: View orders for new books and track delivery status.
* **Book offers to the library manager**: Can provide new books’ information, discounts, prices, and quantities.
* **Books’ status tracking**: Can view how many times a book has been borrowed.

1. Library Manager

* **Books import history viewing**: Track the history of receiving books from publishers and suppliers.
* **Account Management**: Manage users’ and staffs’ accounts.
* **Event management**: Create and configure events for customers and writers.
* **Policy Management**: Define and enforce library policies and procedures.

1. General users

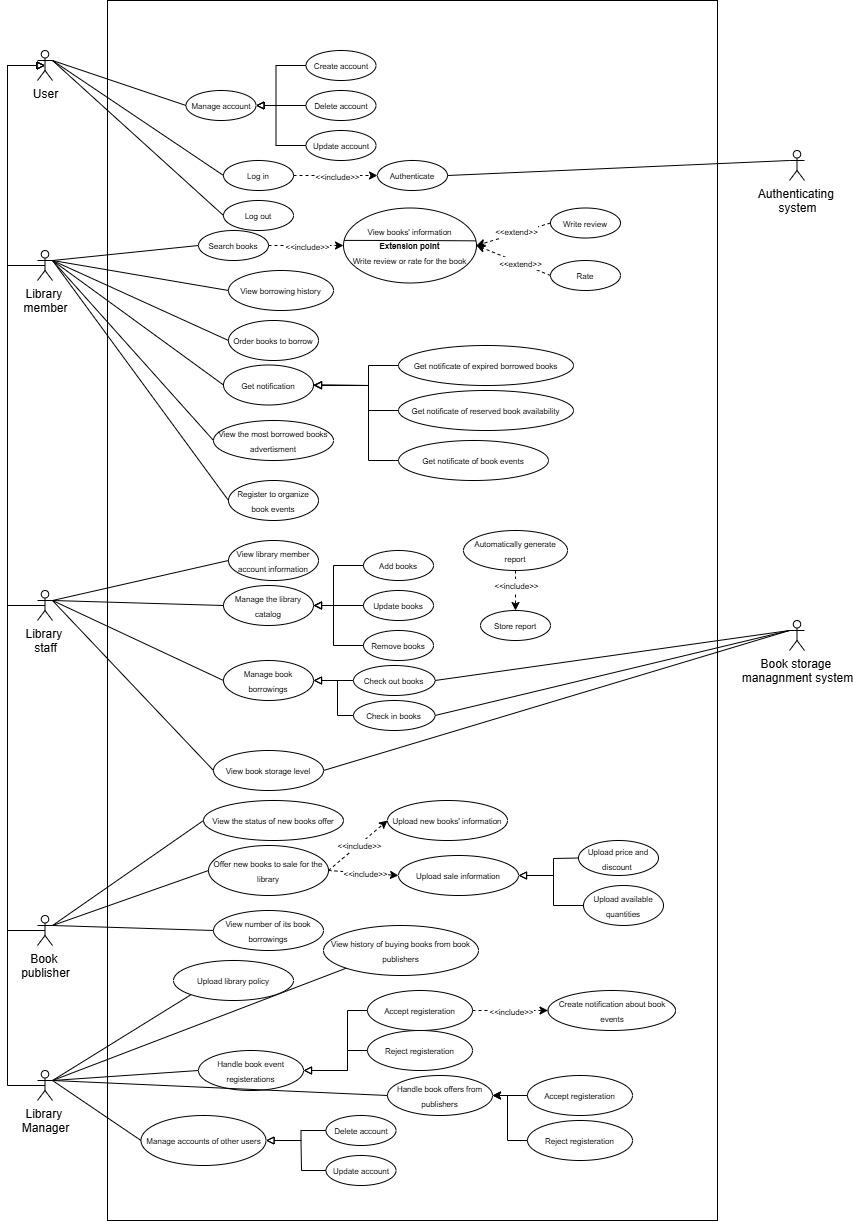
* **Account Management**: Ability to create, update, and delete personal accounts.
* **Authentication**: user must login to an account before using the web application and logout after using.

## **Non- Functional Requirements**

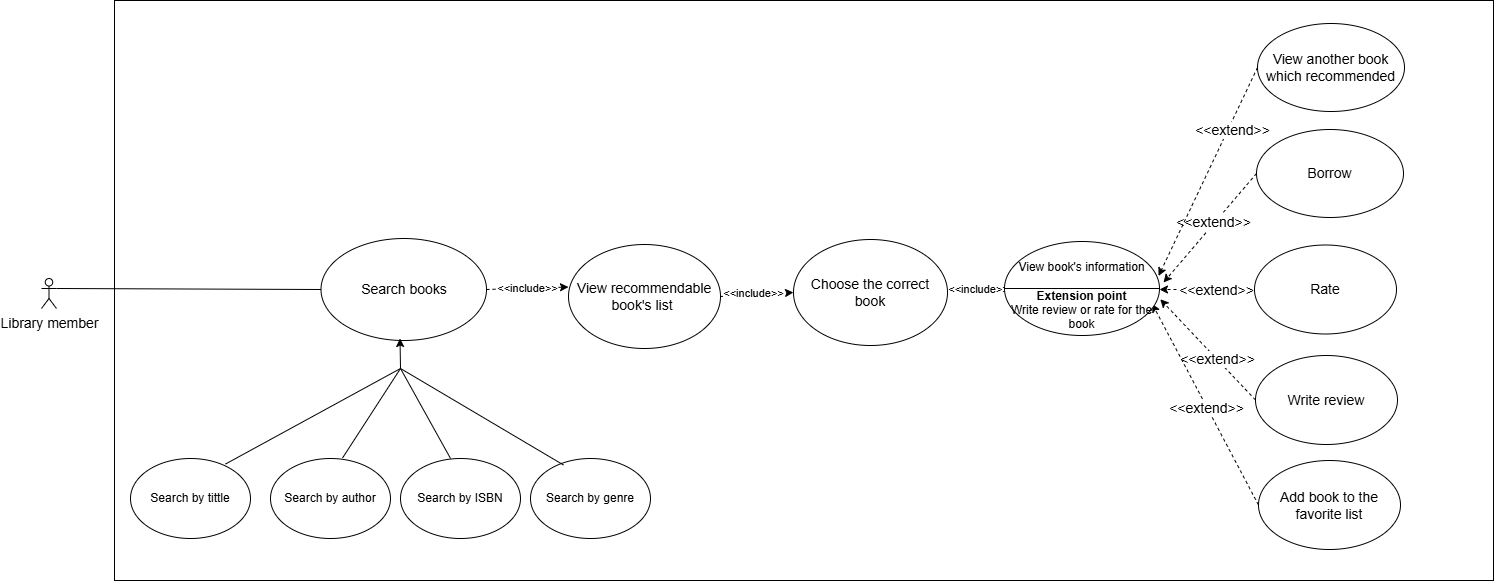
* **Performance:**
  + The system shall support 1000 concurrent users with a response time no more than 5 seconds.
  + For every 2 seconds, the system should load up to 1,000 books.
* **Usability:**
  + The interface should be user-friendly for both administrators and users, and easy to use after completing 3 basic actions.
  + Support Vietnamese language, can add some languages if needed.
* **Availability:**
  + The system will run 7 days a week, 24 hours a day.
  + The web-based app must support consistent display across popular browsers (Firefox, Chrome, IE, Safari, Opera) without requiring additional software installations.
* **Reliability:**
  + Data should be backed up regularly to avoid loss.
  + The system must perform without failure in 90% of use cases during a month.
* **Security:** Ensure security through strong authentication and authorization mechanisms, allowing only verified users to access appropriate resources.
* **Maintainability:** The system should have a container to save source code for developers in the future.
* **Scalability:** Facilitate seamless database updates and enable the development and expansion of features for future enhancements.

# **2. Use-case Diagrams (1.3)**

## **2.1 Use-case Diagram for the Whole System.**



## **2.2 Use-case Diagram for Module “Searching for books”.**

****

## **2.3 The Details of Usecases in Module “Searching for books”.**

* **Use Case: Search for books**

| **Use Case ID** | UC-SE01 | | |
| --- | --- | --- | --- |
| **Use-case name** | **Search for books** | | |
| **Created by** | Thái Bảo | **Last Updated By** |  |
| **Date Created** | 10/10/2024 | **Date Last Updated** | dd/mm/2024 |
| **Actor** | Library Member | | |
| **Trigger** | Users want to search for a book. | | |
| **Description** | Allows the user to search for books by various criteria. | | |
| **Preconditions** | - The user has logged into the system  - The user's device can connect to the internet and the system | | |
| **Postconditions** | A list of books matching the search criteria is displayed. | | |
| **Normal Flow** | 1. User initiates a book search.  2. User selects a search criterion (title, author, ISBN, genre).  3. System displays books matching the criterion. | | |
| **Alternative Flows** | 1. User changes the search criterion.  2. System suggests popular searches. | | |
| **Exceptions** | 1. No books match the search criteria.  2. System error occurs during search. | | |

* **Use Case: View Recommendable Books List**

| **Use Case ID** | UC-SE02 | | |
| --- | --- | --- | --- |
| **Use-case name** | **View Recommendable Books List** | | |
| **Created by** | Thái Bảo | **Last Updated By** |  |
| **Date Created** | 10/10/2024 | **Date Last Updated** | dd/mm/2024 |
| **Actor** | Library Member | | |
| **Trigger** | User requests recommended books. | | |
| **Description** | Displays a list of recommended books based on search results. | | |
| **Preconditions** | Books must be available in the library database. | | |
| **Postconditions** | Recommended books are displayed to the user. | | |
| **Normal Flow** | 1. System generates a list of recommended books.  2. User views the recommended list. | | |
| **Alternative Flows** | 1. User filters recommendations by genre. | | |
| **Exceptions** | 1. No recommendations available.  2. System error occurs during recommendation retrieval. | | |

* **Use Case: Choose the Correct Book**

| **Use Case ID** | UC-SE03 | | |
| --- | --- | --- | --- |
| **Use-case name** | **Choose the Correct Book** | | |
| **Created by** | Thái Bảo | **Last Updated By** |  |
| **Date Created** | 10/10/2024 | **Date Last Updated** | dd/mm/2024 |
| **Actor** | Library Member | | |
| **Trigger** | User selects a book from the list. | | |
| **Description** | Allows the user to select a book from the search results. | | |
| **Preconditions** | Search results are displayed. | | |
| **Postconditions** | The selected book's details are displayed. | | |
| **Normal Flow** | 1. System generates the book list.  2. User selects a book from the list.  3. System displays the book details. | | |
| **Alternative Flows** | 1. User returns to search results to select another book. | | |
| **Exceptions** | 1. Selected book is unavailable.  2. System error occurs during book selection. | | |

* **Use Case: View Book's Information**

| **Use Case ID** | UC-SE04 | | |
| --- | --- | --- | --- |
| **Use-case name** | **View Book's Information** | | |
| **Created by** | Thái Bảo | **Last Updated By** |  |
| **Date Created** | 10/10/2024 | **Date Last Updated** | dd/mm/2024 |
| **Actor** | Library Member | | |
| **Trigger** | Book is selected for viewing details. | | |
| **Description** | Displays detailed information about the selected book. | | |
| **Preconditions** | A book must be selected. | | |
| **Postconditions** | Book information is displayed. | | |
| **Normal Flow** | 1. System retrieves book details.  2. User views the information. | | |
| **Alternative Flows** | 1. User accesses additional resources or related books. | | |
| **Exceptions** | 1. Book details not found.  2. System error occurs during data retrieval. | | |

* **Use Case: Borrow Book**

| **Use Case ID** | UC-SE05 | | |
| --- | --- | --- | --- |
| **Use-case name** | **Borrow Book** | | |
| **Created by** | Thái Bảo | **Last Updated By** |  |
| **Date Created** | 10/10/2024 | **Date Last Updated** | dd/mm/2024 |
| **Actor** | Library Member | | |
| **Trigger** | User opts to borrow the book. | | |
| **Description** | Allows the user to borrow a selected book. | | |
| **Preconditions** | Book information is displayed. | | |
| **Postconditions** | Book is marked as borrowed by the user. | | |
| **Normal Flow** | 1. User opts to borrow the book.  2. System processes the borrowing request. | | |
| **Alternative Flows** | 1. User cancels the borrowing process. | | |
| **Exceptions** | 1. Book is not available for borrowing.  2. System error during borrowing process. | | |

* **Use Case: Rate Book**

| **Use Case ID** | UC-SE06 | | |
| --- | --- | --- | --- |
| **Use-case name** | **Rate Book** | | |
| **Created by** | Thái Bảo | **Last Updated By** |  |
| **Date Created** | 10/10/2024 | **Date Last Updated** | dd/mm/2024 |
| **Actor** | Library Member | | |
| **Trigger** | User selects a rating for the book. | | |
| **Description** | Allows the user to rate a book. | | |
| **Preconditions** | Book information is displayed. | | |
| **Postconditions** | Book rating is updated. | | |
| **Normal Flow** | 1. User selects a rating for the book.  2. System updates the rating. | | |
| **Alternative Flows** | 1. User changes the rating before confirming. | | |
| **Exceptions** | 1. Invalid rating input.  2. System error during rating update. | | |

* **Use Case: Write Review**

| **Use Case ID** | UC-SE07 | | |
| --- | --- | --- | --- |
| **Use-case name** | **Write Review** | | |
| **Created by** | Thái Bảo | **Last Updated By** |  |
| **Date Created** | 10/10/2024 | **Date Last Updated** | dd/mm/2024 |
| **Actor** | Library Member | | |
| **Trigger** | User opts to write a review. | | |
| **Description** | Allows the user to write a review for a book. | | |
| **Preconditions** | Book information is displayed. | | |
| **Postconditions** | Review is added to the book's record. | | |
| **Normal Flow** | 1. User writes a review.  2. System saves the review. | | |
| **Alternative Flows** | 1. User edits the review before submission. | | |
| **Exceptions** | 1. Review input is invalid.  2. System error during review submission. | | |

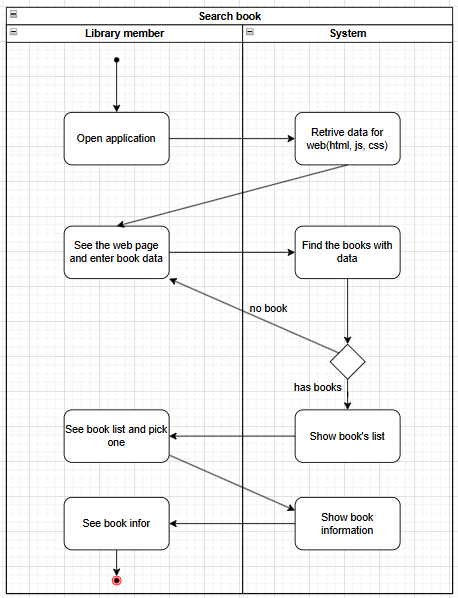
* **Use Case: Add Book to the Favorite List**

| **Use Case ID** | UC-SE07 | | |
| --- | --- | --- | --- |
| **Use-case name** | **Write Review** | | |
| **Created by** | Thái Bảo | **Last Updated By** |  |
| **Date Created** | 10/10/2024 | **Date Last Updated** | dd/mm/2024 |
| **Actor** | Library Member | | |
| **Trigger** | User opts to add the book to favorites. | | |
| **Description** | Allows the user to add a book to their favorites list. | | |
| **Preconditions** | Book information is displayed. | | |
| **Postconditions** | Book is added to the user's favorites list. | | |
| **Normal Flow** | 1. User adds the book to favorites.  2. System updates the favorites list. | | |
| **Alternative Flows** | 1. User removes the book from favorites after adding. | | |
| **Exceptions** | 1. Book already in favorites.  2. System error during update. | | |

**3. sdfds**

**3.1 Activity Diagram**

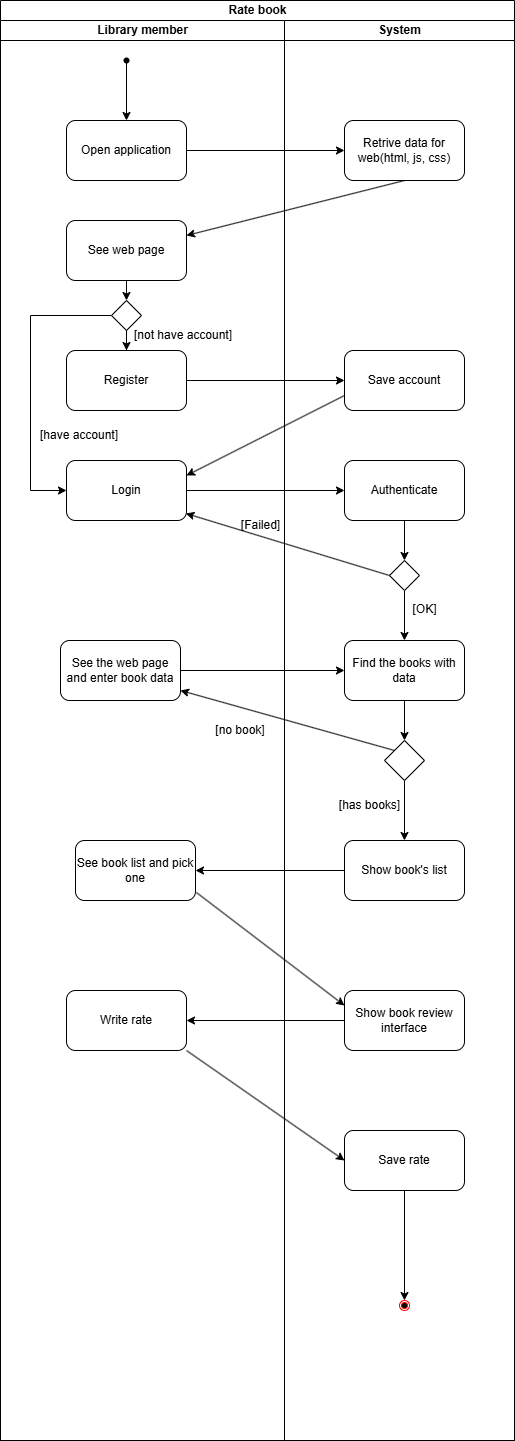
**3.1.1 Activity Diagram for Search Use Case**

****

After the library member enters the URL, a request will be sent to the system, which will retrieve the interface for the user. Once the user sees the interface, they can enter search terms in the search bar to find the books they are looking for. The system will then search and return a list of books that match the search criteria. If no books are found, the user will need to enter the search terms again. If there are matching books, a list of those books will be displayed for the user. The user can select a book to view its detailed information.

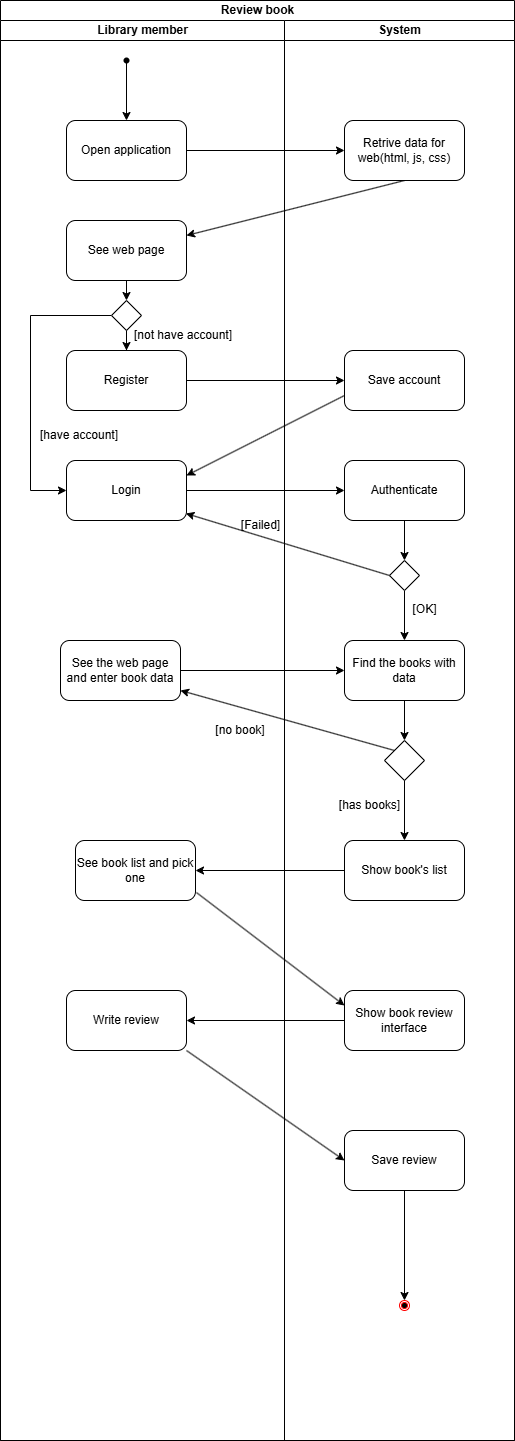
**3.1.2 Activity Diagram for Rate Use Case**

After the library member enters the URL, a request will be sent to the system, which will retrieve the interface for the user. If the user does not have an account, they will need to register, and the system will save that account. Once the account is created, the user must log in to the system; if the login fails, they will have to attempt to log in again. When the login is successful, the user will see the interface where they can enter search terms in the search bar to find the books they are looking for. The system will then search and return a list of books that match the search criteria. If no books are found, the user will need to enter the search terms again. If there are matching books, a list of those books will be displayed for the user. The user can select a book to rate it, and the system will save the rating before concluding the session.

****

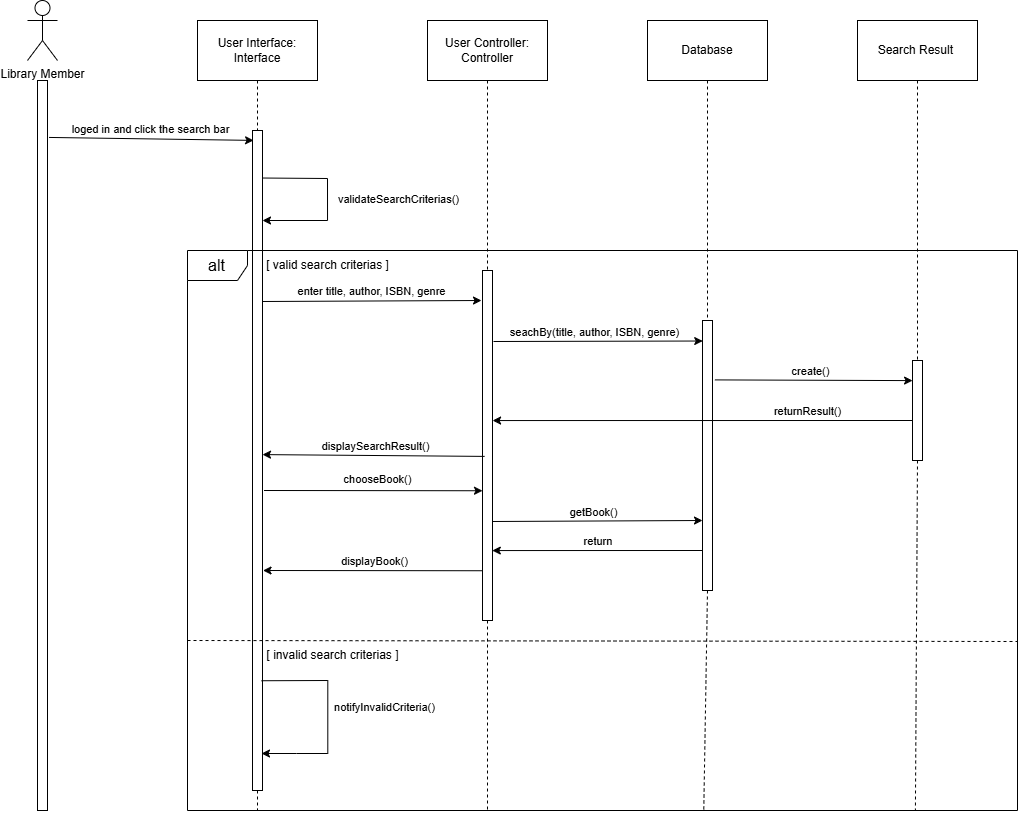
**3.1.3 Activity Diagram for Review Use Case**

After the library member enters the URL, a request will be sent to the system, which will retrieve the interface for the user. If the user does not have an account, they will need to register, and the system will save that account. Once the account is created, the user must log in to the system; if the login fails, they will have to attempt to log in again. When the login is successful, the user will see the interface where they can enter search terms in the search bar to find the books they are looking for. The system will then search and return a list of books that match the search criteria. If no books are found, the user will need to enter the search terms again. If there are matching books, a list of those books will be displayed for the user. The user can select a book to write a review for it, and the system will save those reviews before concluding the session.

****

**3.2 Sequence Diagram**

**3.2.1 Sequence Diagram for Search Use Case**

****

**Context**:  
 A library member wants to search for books in the book management system. This process includes entering search criteria, processing the search, and displaying the results.

**Participants**:

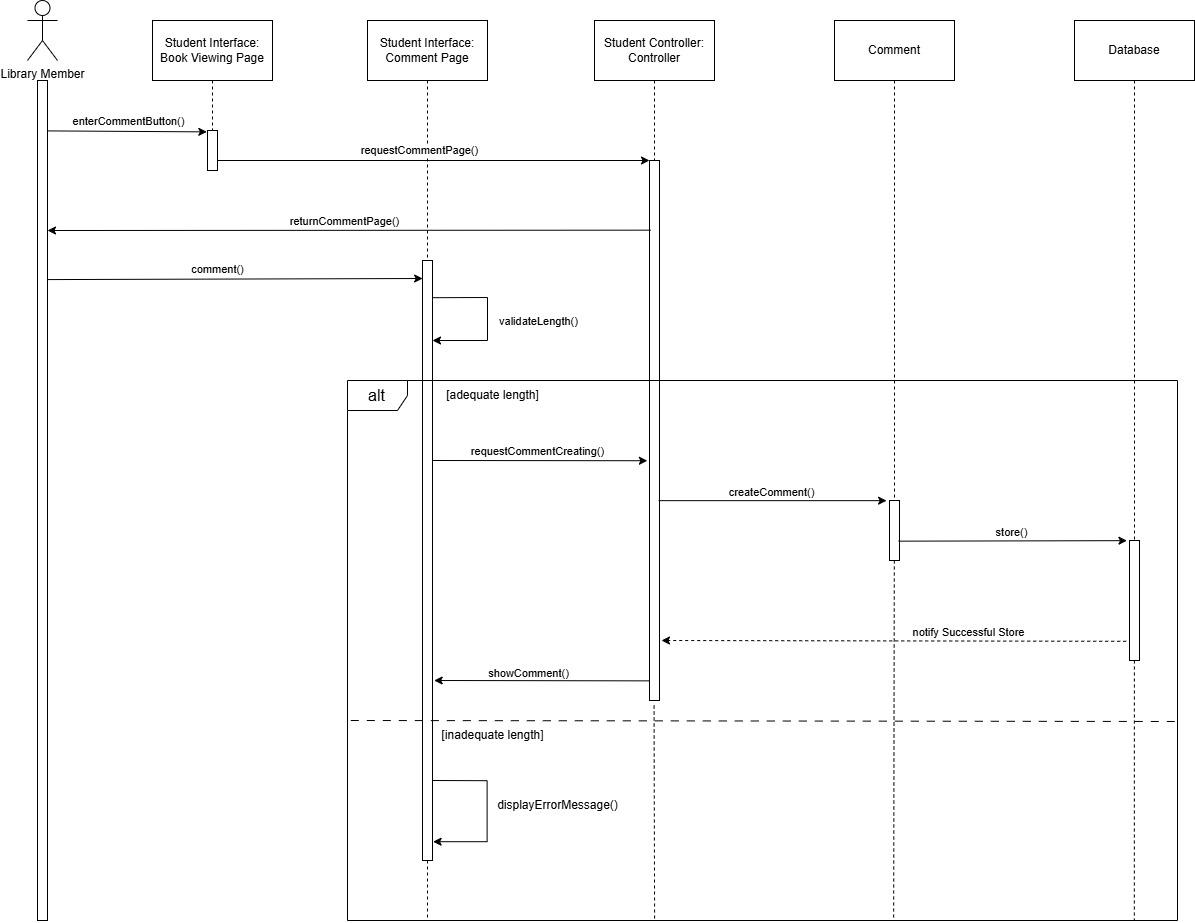
* Library Member: The user conducting the book search.
* User Interface (Interface): The interface where users interact to enter search criteria.
* User Controller (Controller): The controller handling the search logic.
* Database: The storage for book information.
* Search Result: The results returned after the search.

**Workflow**:

1. Initiate Search:  
   * The Library Member logs in and clicks the search bar to start the process.
2. Validate Search Criteria:  
   * The interface validates the search criteria through *validateSearchCriteria()*.
3. If Valid Search Criteria:  
   * The user enters search criteria including title, author, ISBN, and genre.
   * The Controller performs the search using *searchBy(title, author, ISBN, genre)*.
   * The Database processes the query and returns results via *create()* and *returnResult()*.
   * Search results are displayed using *displaySearchResult()*.
   * The user selects a book from the search results with *chooseBook()*.
   * Detailed book information is retrieved from the database using *getBook()*.
   * Book information is displayed using *displayBook()*.
4. If Invalid Search Criteria:  
   * If the search criteria are invalid, the system notifies the user of the error through *notifyInvalidCriteria()*.

**Conclusion**:  
 This sequence diagram provides a detailed depiction of the book search process, from entering criteria to receiving results or error notifications. It clarifies the steps and interactions between the components in the system.

**3.2.2** **Sequence Diagram for Commenting Use Case**

****

**Context**:  
 A library members are able to give comments on books in the system, the progress of it contains four main stages which are moving to comment page, enter comments, validate comment and display comments in the page

**Participants**:

* Library Member: The users who want to give comments on books.
* Book viewing page: The page where users can view details of books.
* Comment page (Interface): The place where users can enter comments.
* Controller: The module is responsible for handling logic of comment progress.
* Comment: The data objects are ready to be stored in the database.
* Database: The storage of all the comments.

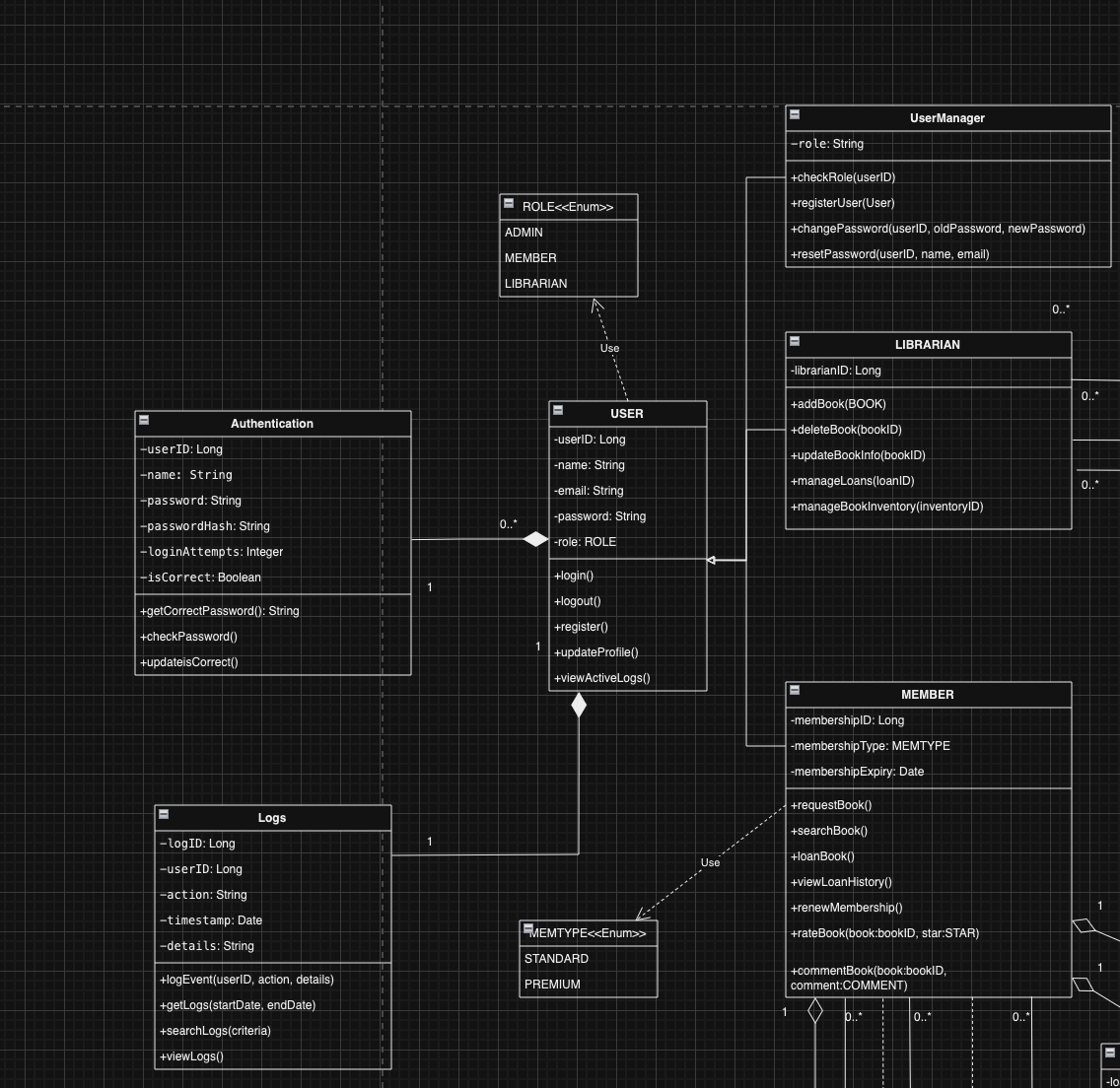
**Workflow**:

1. Moving to comment page:  
   * The Library Member logs in and clicks the search bar to start the process.
   * The users will click the comment button to move to the comment page, then the controller will handle the request and change view to the page.
2. Entering comments:
   * The users will enter their comments and the system will validate whether the comments are long enough.
3. If the comment is valid:
   * The comment page will send a request along with the comment data to the controller, using requestCommentCreating().
   * The controller will create a comment object to store in the database, using createComment().
   * A comment object will be stored in the database, using store() and then the database will send a successful message back to the controller.
   * When the controller receives the message, it will show the comments to the view page, using showComment().
4. If the comment is invalid:
   * When the comment is not long enough, the system will send an error message back to the user by showing on the view page, using displayErrorMessage().

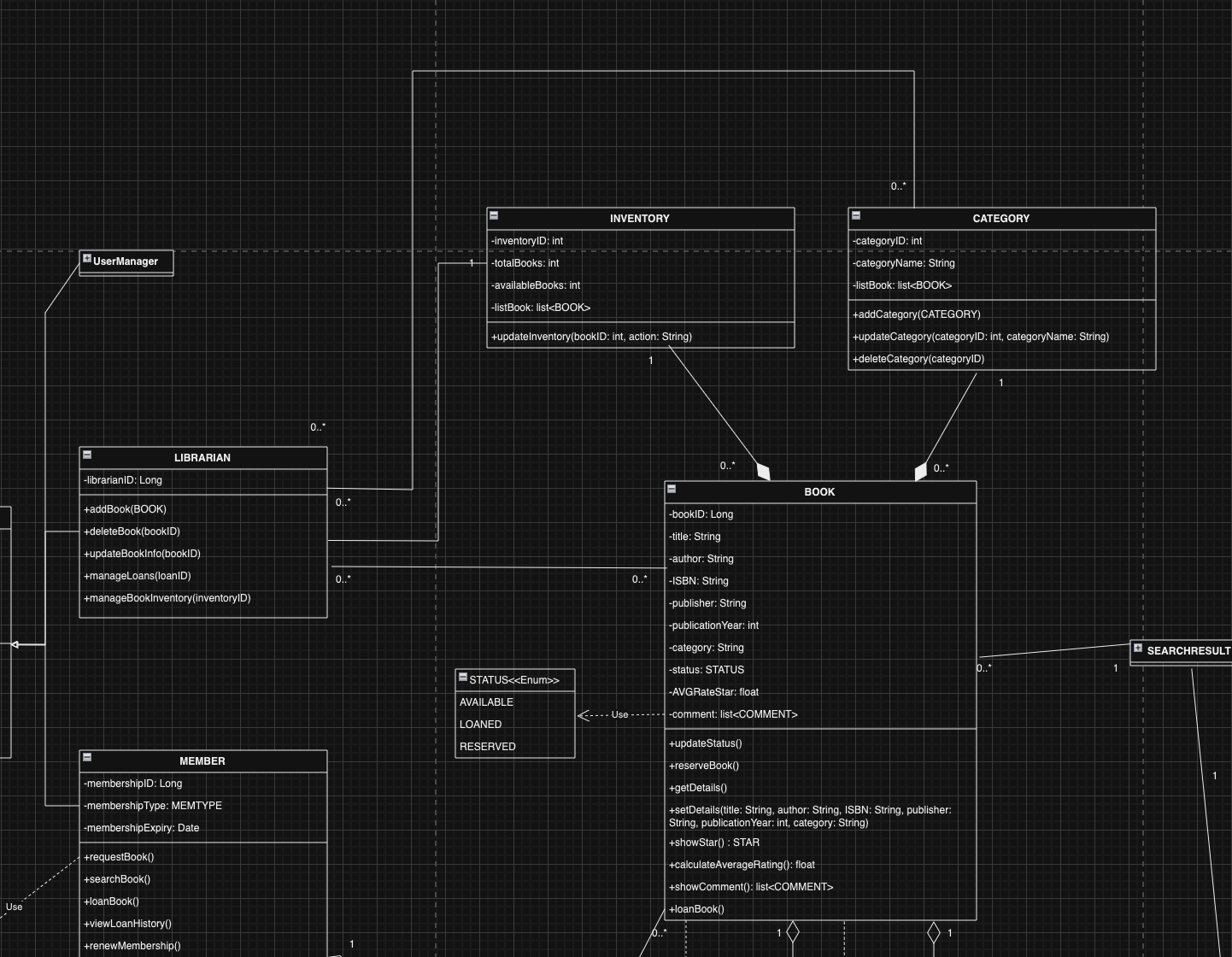
**Conclusion**:  
 The review sequence diagram illustrates the progress of giving comments on books by users. This diagram shows detailed steps of the process which are moving to the comment page, entering comments, validating comments and displaying comments in the page.

**3.3. Class Diagram**

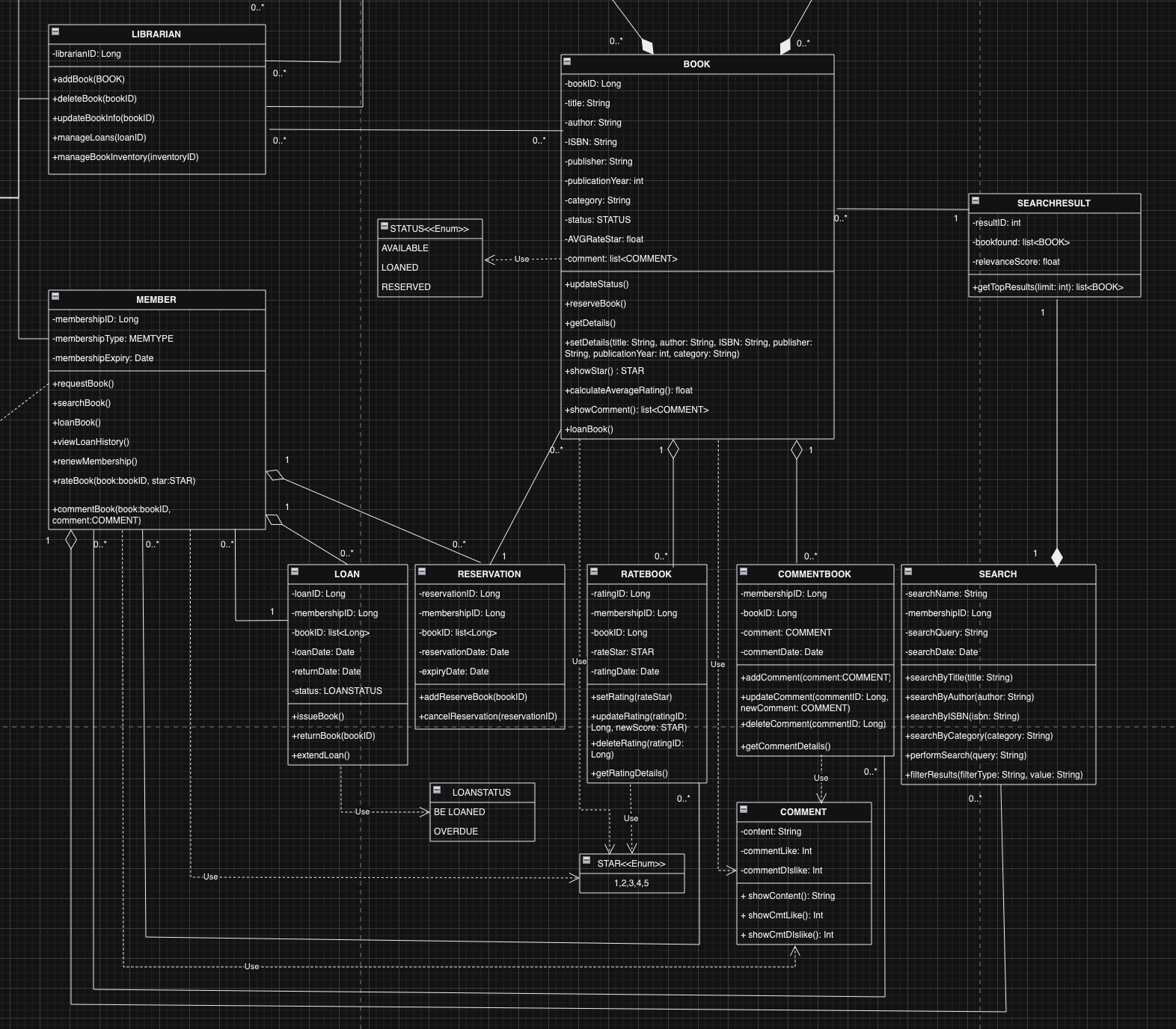
**3.3.1. Class Diagram for User Manager:**



**3.3.2. Class Diagram for Librarian Management:**



**3.3.3. Class Diagram for User Service:**



**Overview:**

* **User** can be a **Member** or **Librarian** (inherits from **User**).
* **Member** can borrow or reserve **Book**.
* **Librarian** has rights to manage books (add, delete, update).
* **Loan** is a relationship between **Member** and **Book**. Each **Loan** belongs to only one **Member** and one **Book**.
* **Reservation** is a relationship between **Member** and **Book**. A **Member** can reserve multiple **Books**, and each **Book** can be reserved by multiple **Members**.
* **Book** belongs to a **Category**. Each **Category** can contain multiple **Books**.

**SearchBook:**

* **User** can perform searches via the **searchBook()** method, related to the **Search** class.
* **Search** will return results as a collection of **SearchResult**, each containing a **Book** object.
* **Book** will have methods to search based on various attributes, such as title, author, ISBN, and category.
* **SearchResult** links with **Book** to store information about the book and the relevance of the search result.
* **User** performs a search and receives corresponding results.
* **Search** processes and filters search results.
* **SearchResult** contains information about the book and the relevance of the search result.

**ManageBook:**

* **Librarian** performs book management actions, such as adding, updating, or deleting books through the methods **addBook()**, **updateBook()**, and **deleteBook()**.
* **Book** is the object that the **Librarian** manages. **Book** has related attributes and methods to update book information.
* **Category** is the class used to manage book categories. A **Book** belongs to a **Category**, and **Librarian** can manage these categories.
* **Inventory** manages the quantity status of books in the library. Each time the **Librarian** adds or deletes a book, **Inventory** updates through the **updateInventory()** method.
* **Librarian** can add, update, or delete books and manage book information in the system.
* **Book** contains information about the book and can be updated or deleted.
* **Category** manages book categories and can be added, modified, or deleted.
* **Inventory** tracks the number of books available in the library.

**User** can use **rateBook()** and **commentBook()**. Each **User** can rate and comment on multiple **Books**.

* **Rating** and **Comment** are related to **Book**; each **Book** can have multiple ratings and comments from various **Users**.
* **Book** will calculate **AVGRating** based on user ratings.
* **Rating**: Users can rate books and add comments regarding the quality of the book.
* **Comment**: Users can comment on books, sharing their opinions with other users.
* **Book**: Will have a new attribute **AVGRating** to store the average rating and a method to calculate this rating based on user ratings.
* **User A** can rate **Book B** with a star rating and comment. **Book B** will update its average rating.
* **User B** can comment on **Book B** without a rating.