★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★

System Analysis and Design

Smart Library

Team Member

2151268 Li Haocheng
2152038 Ge Yuyang
2152588 Wang Jingyue
2152602 Wang Xinglin
2153487 Ding Yimin

Instructor

Sun Ping

1. Introduction

1.1 Project Goals

The university library provides a large number of book resources for students and teachers, which is an important space for learning and exchanging ideas. Therefore, it is necessary to establish an efficient and reliable library management system.

Meanwhile, human society is experiencing a new round of technological and industrial revolutions led by artificial intelligence, big data, and other technologies. However, these emerging technologies are not fully applied in the field of library management. Based on the concept of Smart+, we believe that a library management system should provide more convenient and efficient services for schools and society.

Through software engineering and big data technologies, we plan to develop a smart university library management system. The target users of our system include four kinds of people. They are:

- University students: They can inquire, borrow, and renew books on the system, and deal with violations. They can also make reservations for seats and log on to the forum to communicate with people who have similar reading interests.
- Teachers: They have similar functions as students, but they have a higher authority to query and borrow books and periodicals.
- Public users: They can search for books on the system and read books and periodicals in the reading room, but they cannot take them out of the library. They can also reserve the reading room seats.
- Librarians: They can manage all the borrowing-returning transactions through this system, check out the information and borrowing status of the books, and manage the borrowing rights of various users.

This is how we envision our system to work and benefit different user groups.

1.2 Progress and Current Status

In the previous assignment, we divided the smart library system into five subsystems: Book Management System, Account Management System, Book Borrowing Management System, Venue Management System, and Reader Communication System. We also presented their use case models and analyses in detail.

For this assignment, we first provide the architecture diagram of the entire system and explain its components and rationale. The architecture diagram shows how the subsystems interact with each other and with external entities such as users, databases, and hardware devices. It also shows the layers of abstraction that separate the presentation, business logic, and data access of the system. We also describe the design principles and patterns that we used to create a modular, scalable, and reliable system.

In this assignment, we also design the components at each level based on this architecture. We use class diagrams to show the classes and their attributes, methods, and associations for each subsystem. We also use interaction diagrams to show how the classes collaborate to realize the use cases of each subsystem. We use sequence diagrams to show the temporal order of messages exchanged between objects, and we use communication diagrams to show the structural organization of objects and their links.

During the analysis, we also identify and improve the poorly designed elements in the use case models of the previous assignment. For example, we eliminate redundant or unnecessary use cases. We also modify inconsistent or incorrect use cases.

Therefore, we have currently completed the architectural analysis of the entire system and the class and interaction analysis of each subsystem.

2. Architectural Analysis

2.1 System-Level Architecture Analysis and Designs

The smart library system is a web-based platform focused on library affairs and book lover communication. To promote our system and seize opportunities, we prioritize rich functions, secure personal information, and extract business mechanisms for improved performance.

2.1.1 System Detailed Analysis

During the requirement phase, our primary focus is on understanding the interactions between various objects external to the system and the system itself. This involves identifying the stakeholders, gathering their requirements, and comprehending how they interact with the system. Through this process, we gain a comprehensive understanding of the system's functional and non-functional requirements.

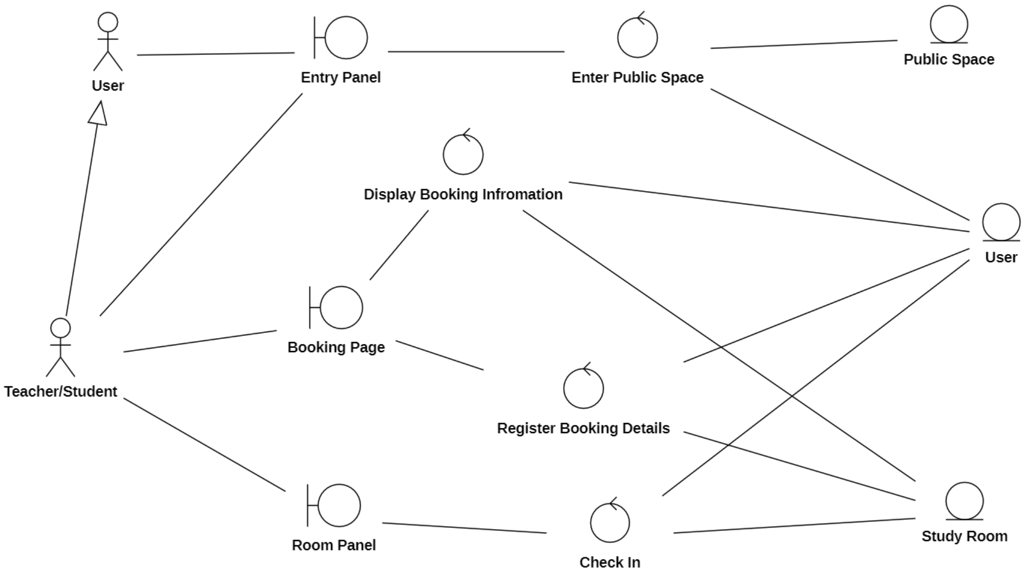
To effectively capture and analyze these requirements, we utilize use cases as a valuable tool. Use cases allow us to define and illustrate the system's behavior from the perspective of the users, actors, or external entities involved. By creating use case diagrams and narratives, we establish a clear understanding of how different actors interact with the system and the desired outcomes.

In the early stages of the architecture design, we delve into the specific steps and behaviors required to realize each use case. This involves breaking down the system into smaller, more manageable components or subsystems based on their functional characteristics. By doing so, we can identify the specific responsibilities and functionalities of each component, ensuring a modular and organized system structure.

Dividing the system into smaller granularity systems based on functionality allows for better control, maintenance, and scalability. It enables us to focus on individual components and their interactions, ensuring that each component performs its designated function effectively. This approach promotes modularity, reusability, and simplifies system maintenance and enhancements in the future.

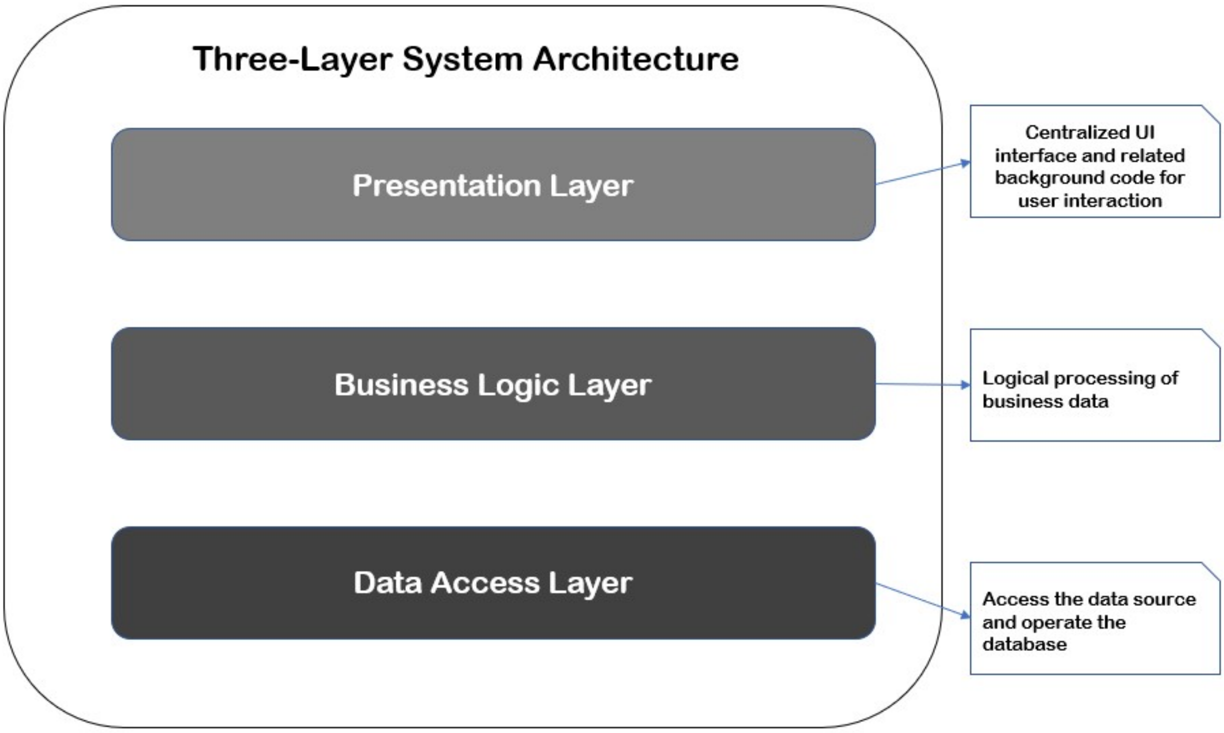
By conducting a comprehensive analysis of the system's requirements and decomposing it into smaller granularity systems, we lay a solid foundation for the subsequent stages of the system design and development process.

- A demonstrative Robust Diagram of the Venue Management System

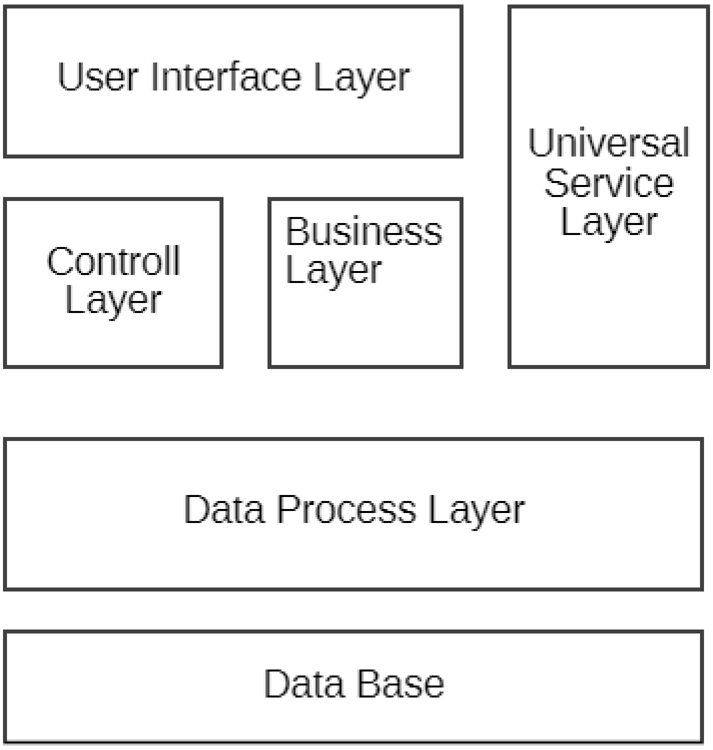


2.1.2 Hierarchical Architecture Design

According to the reference, the traditional three-layers architecture design has presentation, business logic and data access layer.

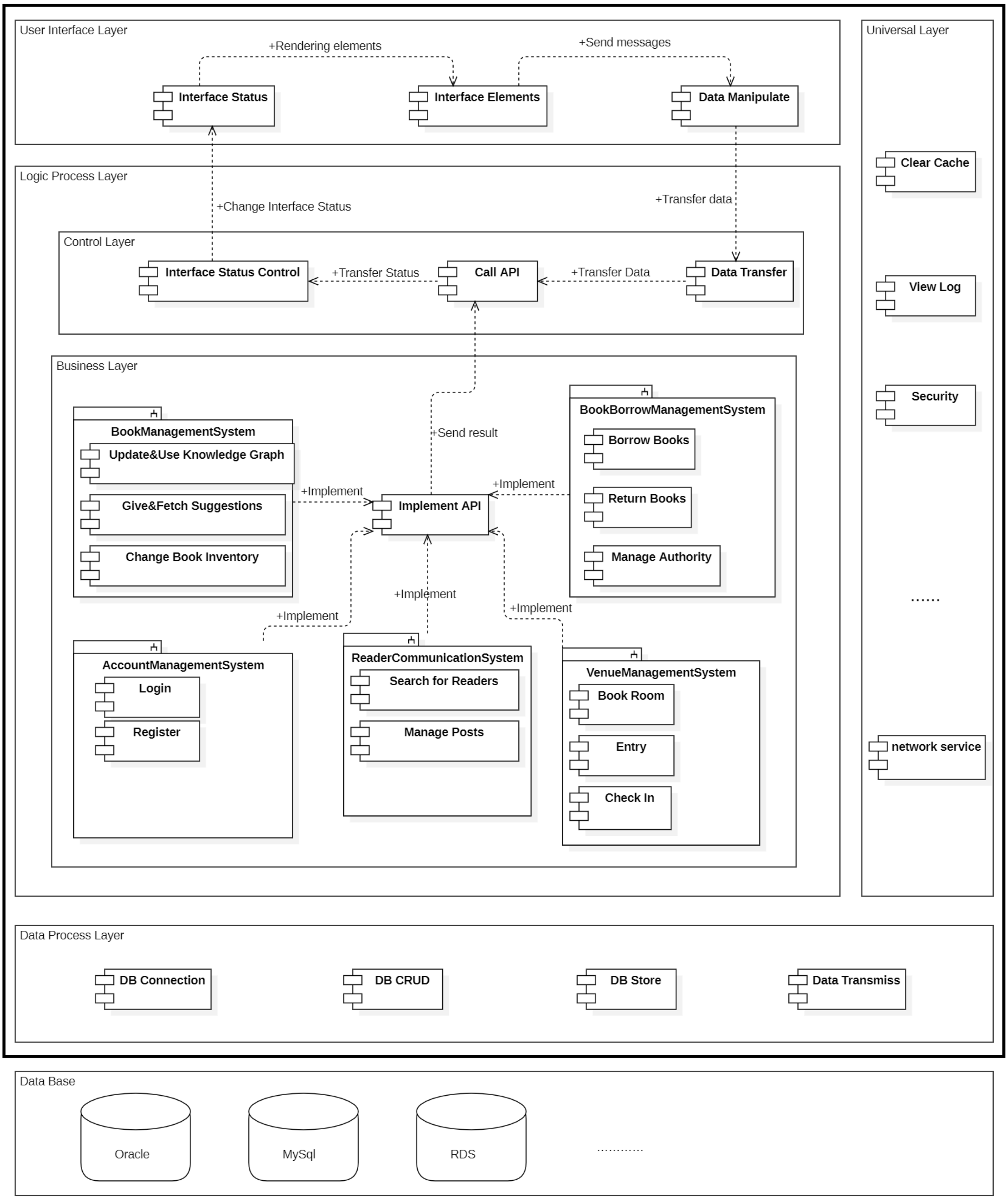

Three-Layer System Architecture
Presentation Layer
Business Logic Layer
Data Access Layer
Centralized UI interface and related background code for user interaction
Logical processing of business data
Access the data source and operate the database

We enhance and strengthen the architecture analysis and design of the system by dividing the traditional three-layer logic system architecture in to fine-grained sub-layers. The system-level architecture can be denoted by the blocks of sub-layers as follow:



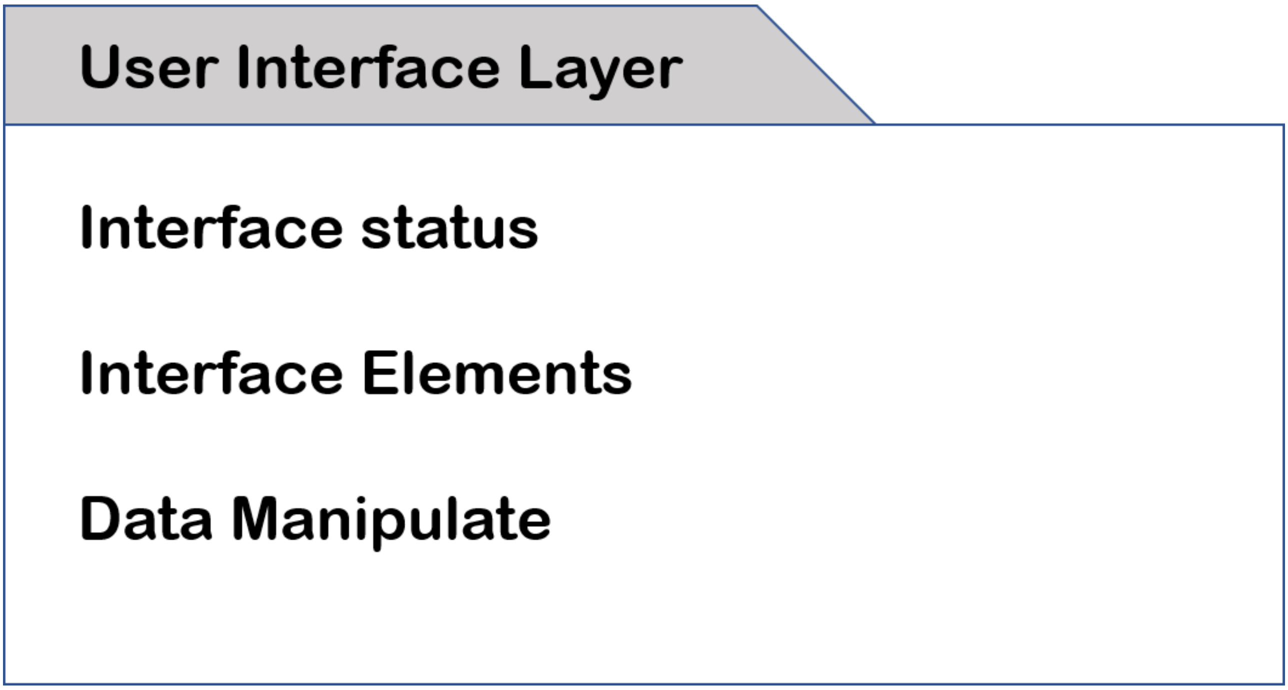
By means of a package diagram and components, we seamlessly incorporate the aforementioned architecture design into the smart library management system, enhancing its overall functionality. The integration process capitalizes on the system's existing capabilities and harmoniously merges them with the proposed architecture design. This strategic amalgamation creates a cohesive and robust framework that optimizes the system's performance and efficiency. Through the careful alignment of the package diagram and components, we ensure a seamless integration that preserves the intended purpose and objectives of the smart library management system.

The high level architecture diagram is as follow:



2.1.2.1 User Interface Layer

The User Interface layer contains the interface information and components that interact with users. The presentation layer responds to user events through components, and encapsulates the page data as VO (view object), which can be directly transferred to the application layer components for further business logic processing, and can also directly call the external API to realize functions.



- Interface status

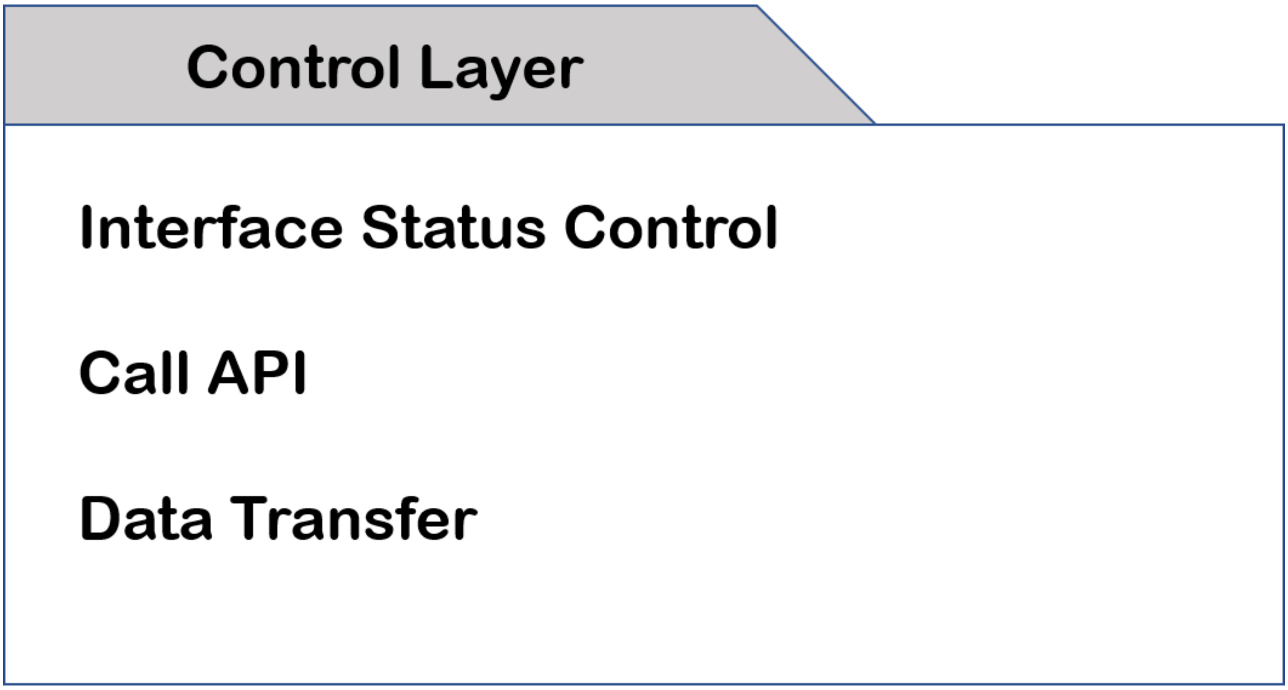
  Provides real-time feedback and system status updates to users.
- Interface Elements

  Includes buttons, forms, menus, and other interactive elements for seamless user interaction.
- Data Manipulate

  Handles data processing, validation, and transformation within the system.

2.1.2.2 Control Layer

The control layer receives and processes user requests. When the user initiates a request, the Control layer will judge the target of the request according to the requested URL, HTTP method and other information, and call the Business layer to process the business logic. When the Controller needs to process the business logic, the method of the Business layer will be called, and the Business layer will process it accordingly according to the specific business requirements. Finally, return to the processing results. After completing the request processing, the Control layer will send the processing results to the front-end user page for display.



- Interface Status Control

  Manages and updates the user interface status to keep users informed.

- Call API

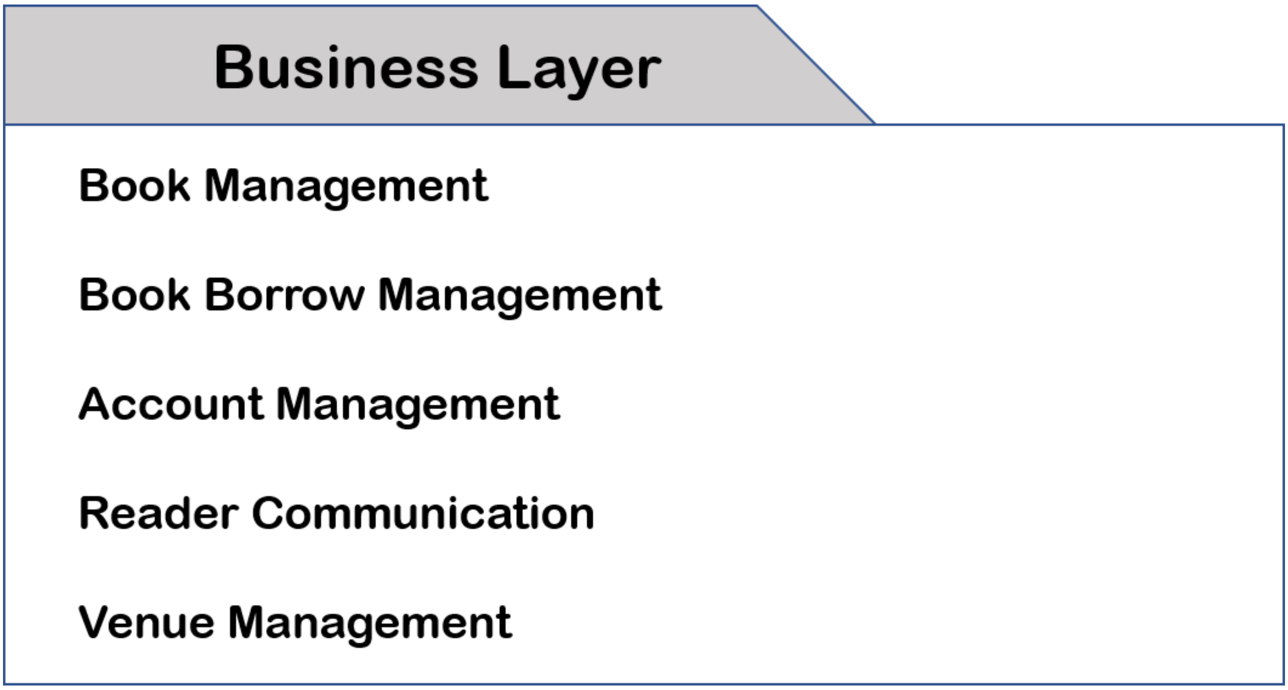
  Facilitates communication between the user control layer and the system's backend through API calls.

- Data Transfer

  Handles secure data exchange within the user control layer and with external systems.

2.1.2.3 Business Layer

The business layer is the core value part of the system architecture. Its focus is mainly on the formulation of business rules, the implementation of business processes and other system design related to business requirements, that is to say, it is related to the field (Domain) logic that the system deals with. In many cases, the business logic layer is also called the domain layer.



- Book Management

  Realizes the management of book resources and supports the classification, cataloguing, labelling, indexing, input, modification, deletion, and other functions of the collection of books.

- Book Borrow Management

  Manages the borrowing information and support readers' borrowing, returning, renewing, and other businesses.

- Account Management

  Supports the management of reader information and the operation of user account registration, login, and password modification. In addition, readers can be divided into teachers, students, and the public for separate management, providing different permissions for different types of readers.

- Reader Communication

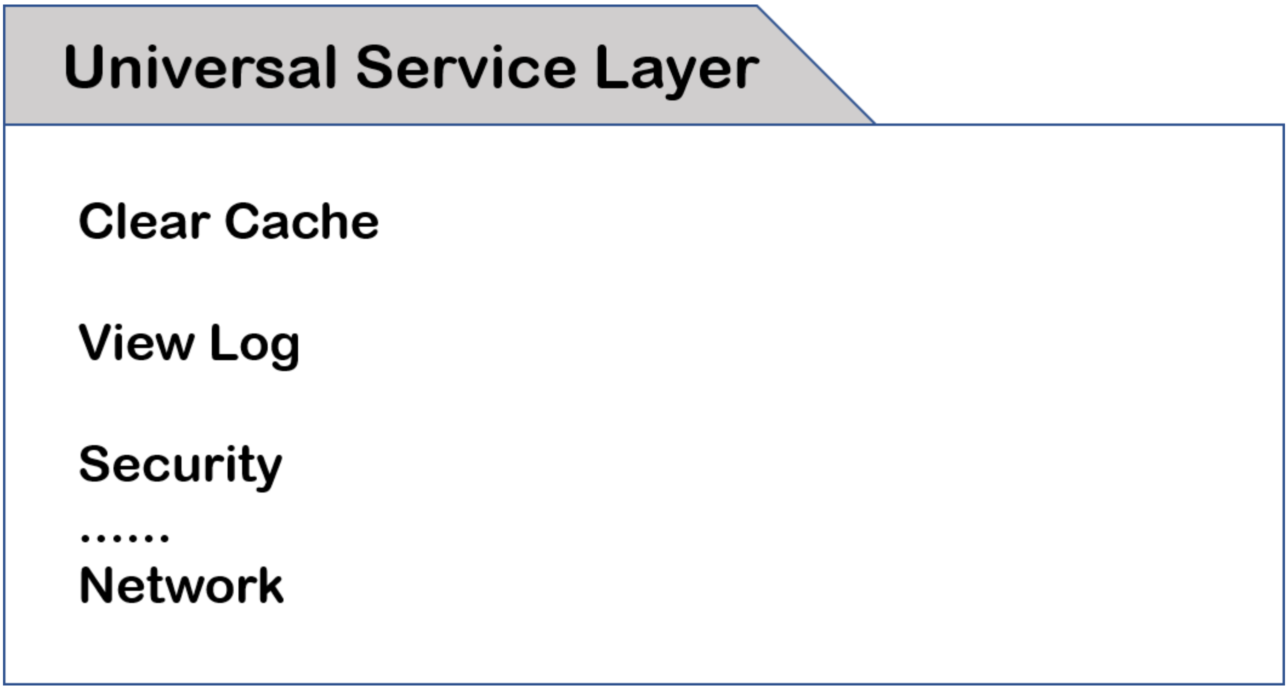
  Supports the reader communication function, which can analyze readers' reading preferences according to their borrowing records, and help readers find other readers with the same interests to communicate according to their reading preferences.

- Venue Management

  Supports the management of venue resources and users' reservations of venues.

2.1.2.4 Universal Service Layer

The universal layer contains some functional services which are closely related to all levels of the system architecture for the components of all levels to call.



- Clear Cache

  Manages cache clearing and refreshing for up-to-date data.

- View Log

  Allows users to analyze system logs for monitoring and issue tracking.

- Security

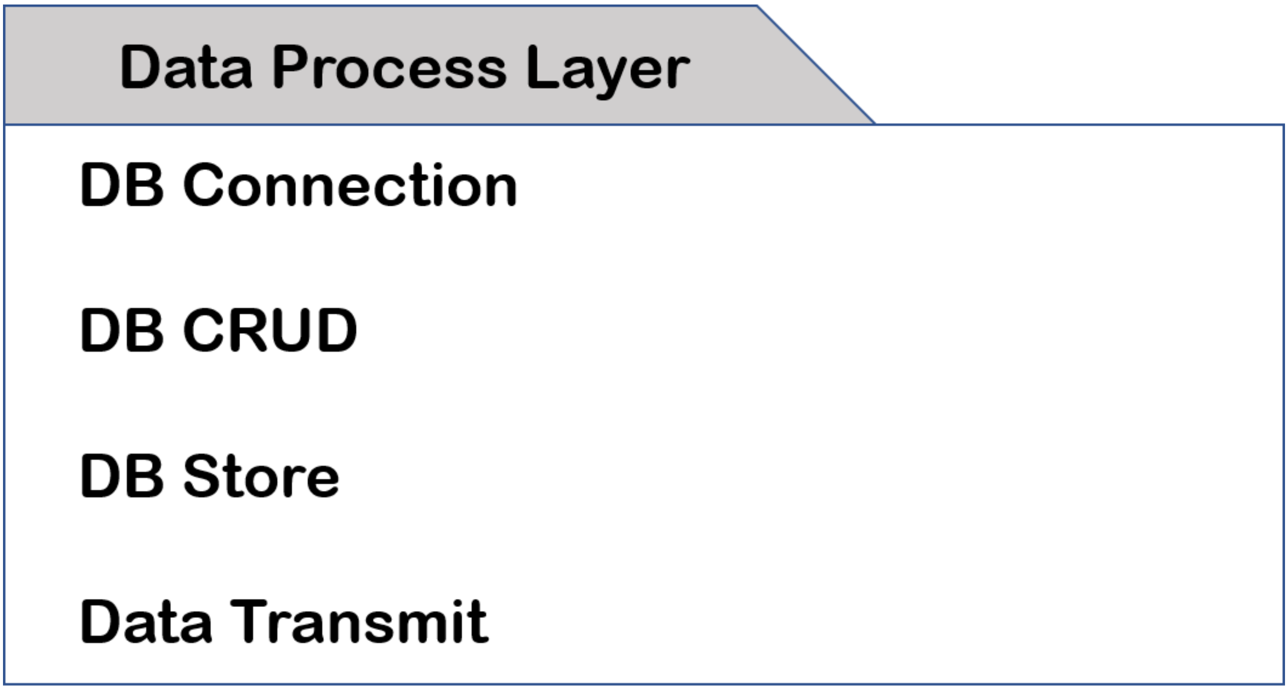
  Enforces authentication, authorization, and data protection measures.

- Network

  Handles communication and connectivity between components and systems.

2.1.2.5 Data Process Layer

The function of the data processing layer is mainly responsible for the database access, and you can have access to the database system, binary files, text documents, or XML documents. The simple statement is to implement the operation of the data table Select, Insert, Update, Delete. If you want to include elements of the ORM, then it includes the mapping between the object and the data table, and the persistence of the object entity. The transactions done by this layer directly operate the database, to add, delete, modify, and find the data, etc. The function in the data access layer is atomic, namely minimum and inseparable.



- DB Connection

  Manages the connection between the system and the database.
- DB CRUD

  Handles basic database operations like create, read, update, and delete.
- DB Store

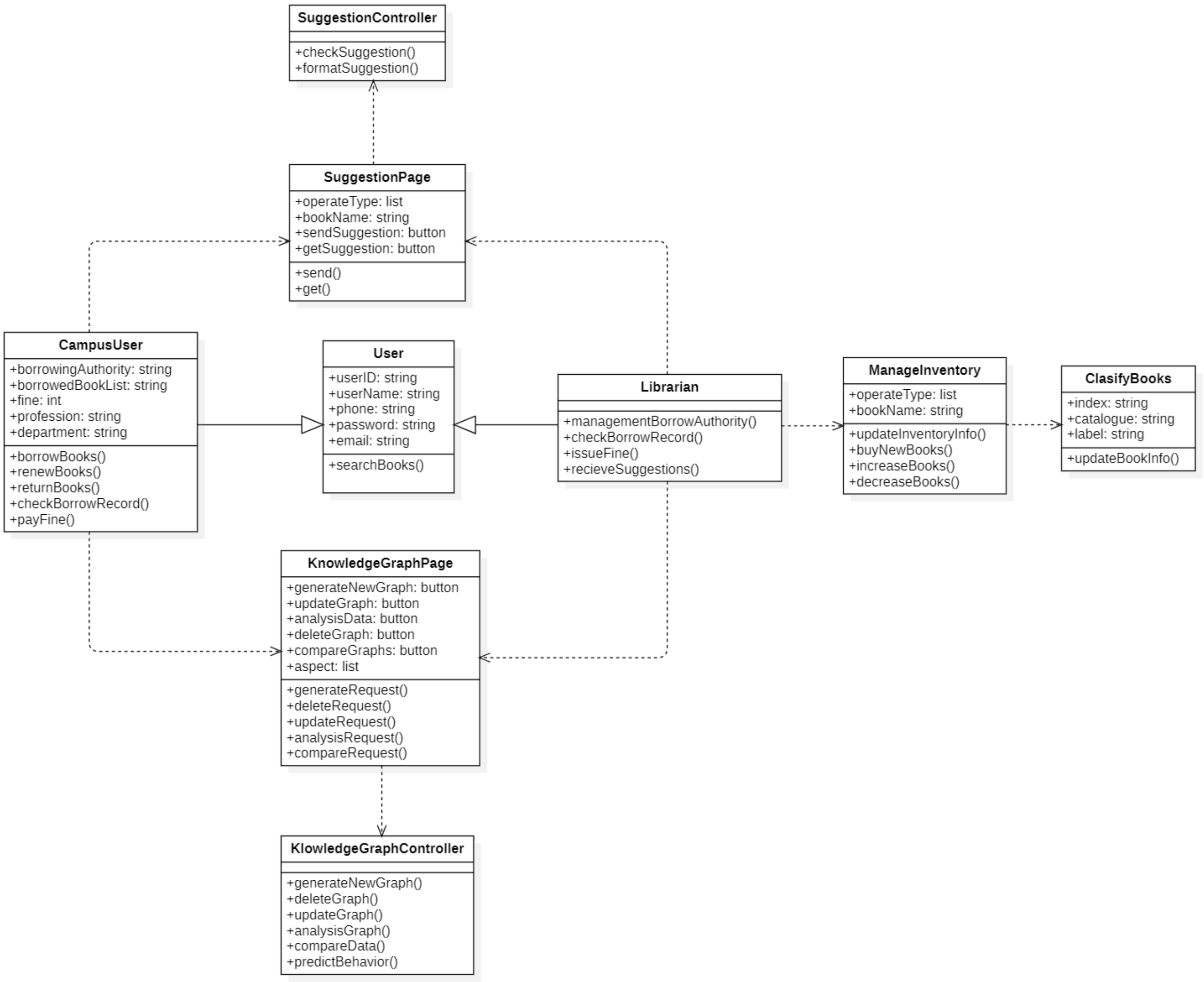
  Stores and organizes data within the database.
- Data Transmit

  Facilitates secure data transmission between the system and the database.

3. Analysis Model

3.1 Book Management System

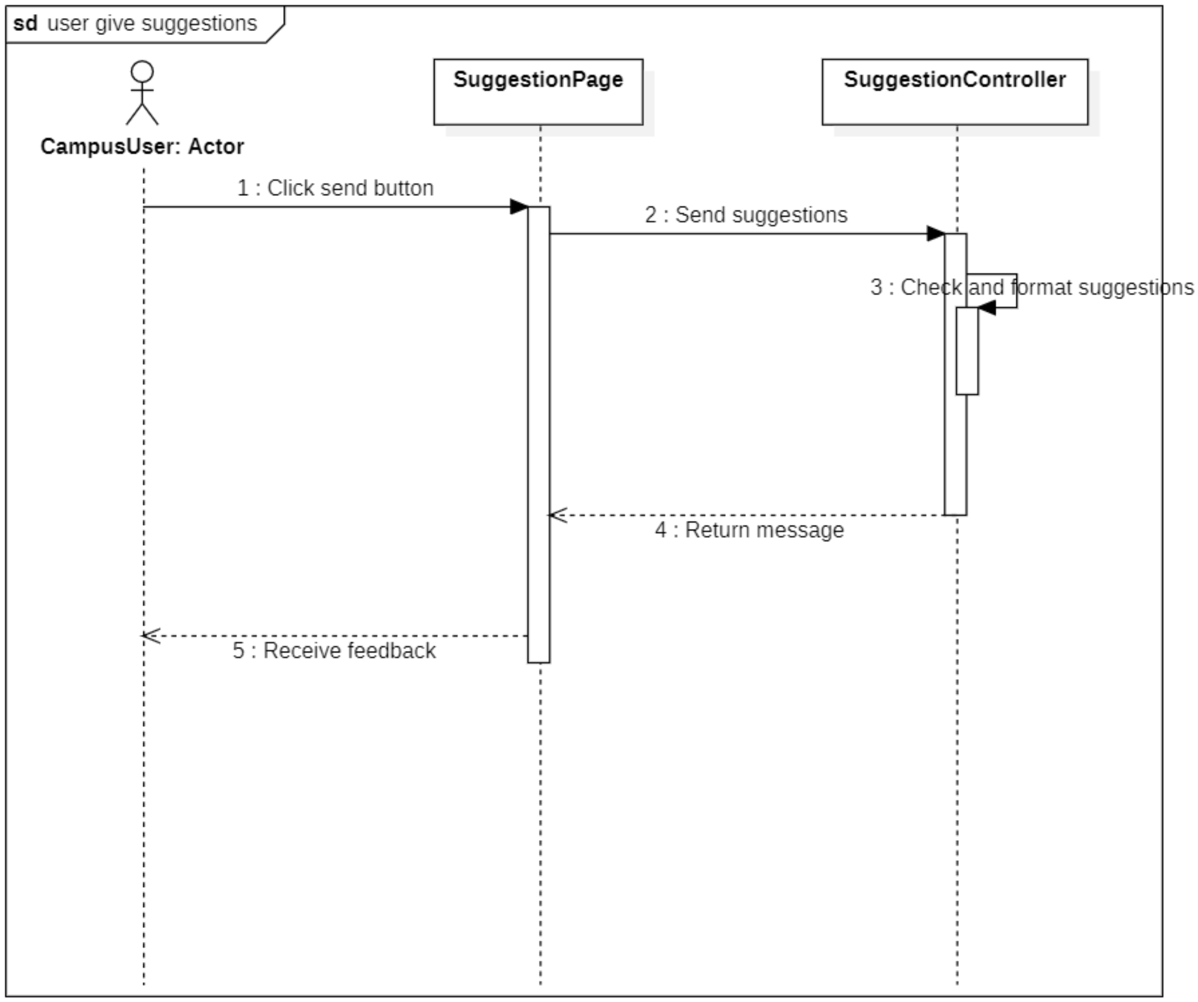
3.1.1 Class Diagram



3.1.2 Sequence Diagrams

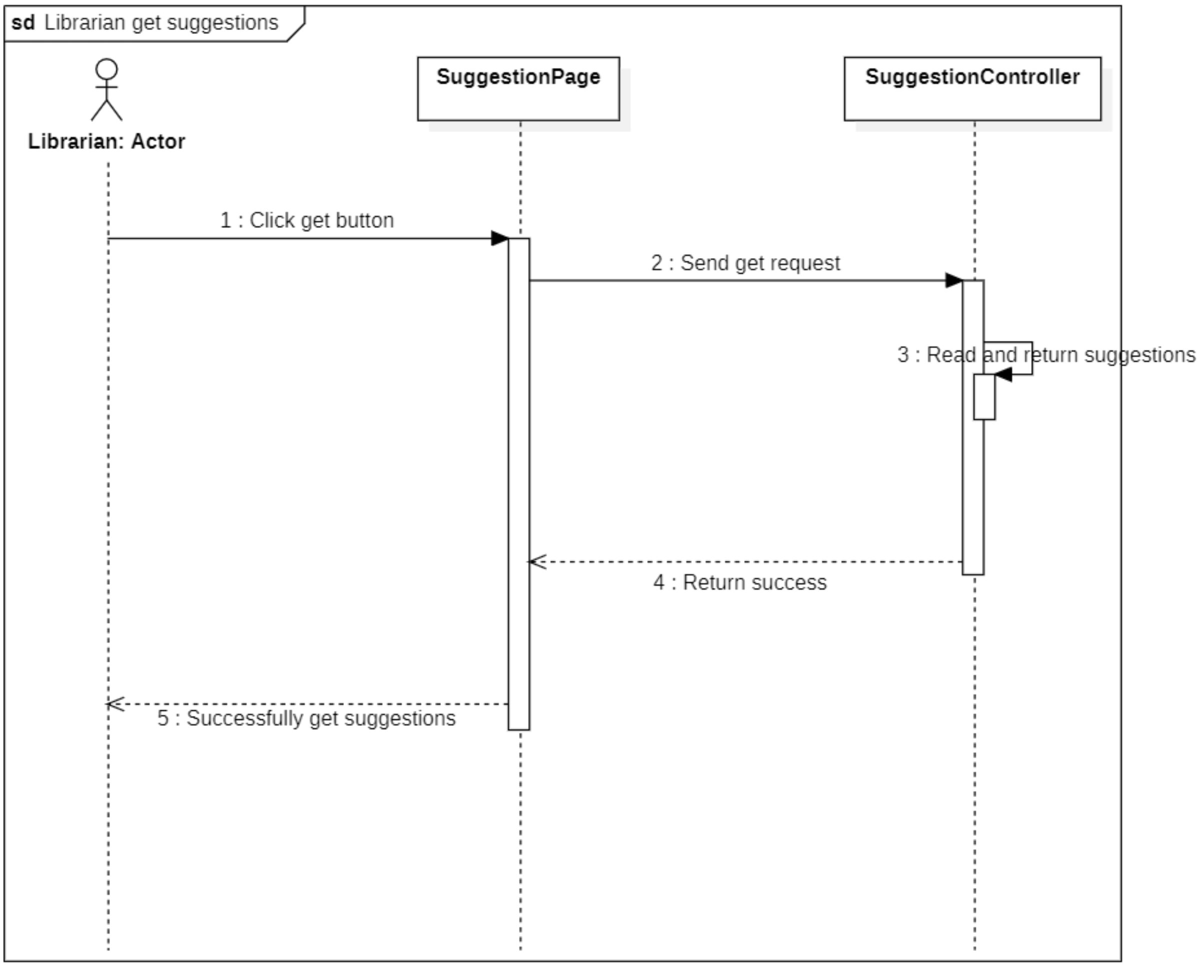
3.1.2.1. User Give Suggestions

This sequence diagram shows what happens after a user submits a suggestion. The page sends the suggestion as a request to the backend, which receives it, returns a success message, and gives feedback in the user interface.



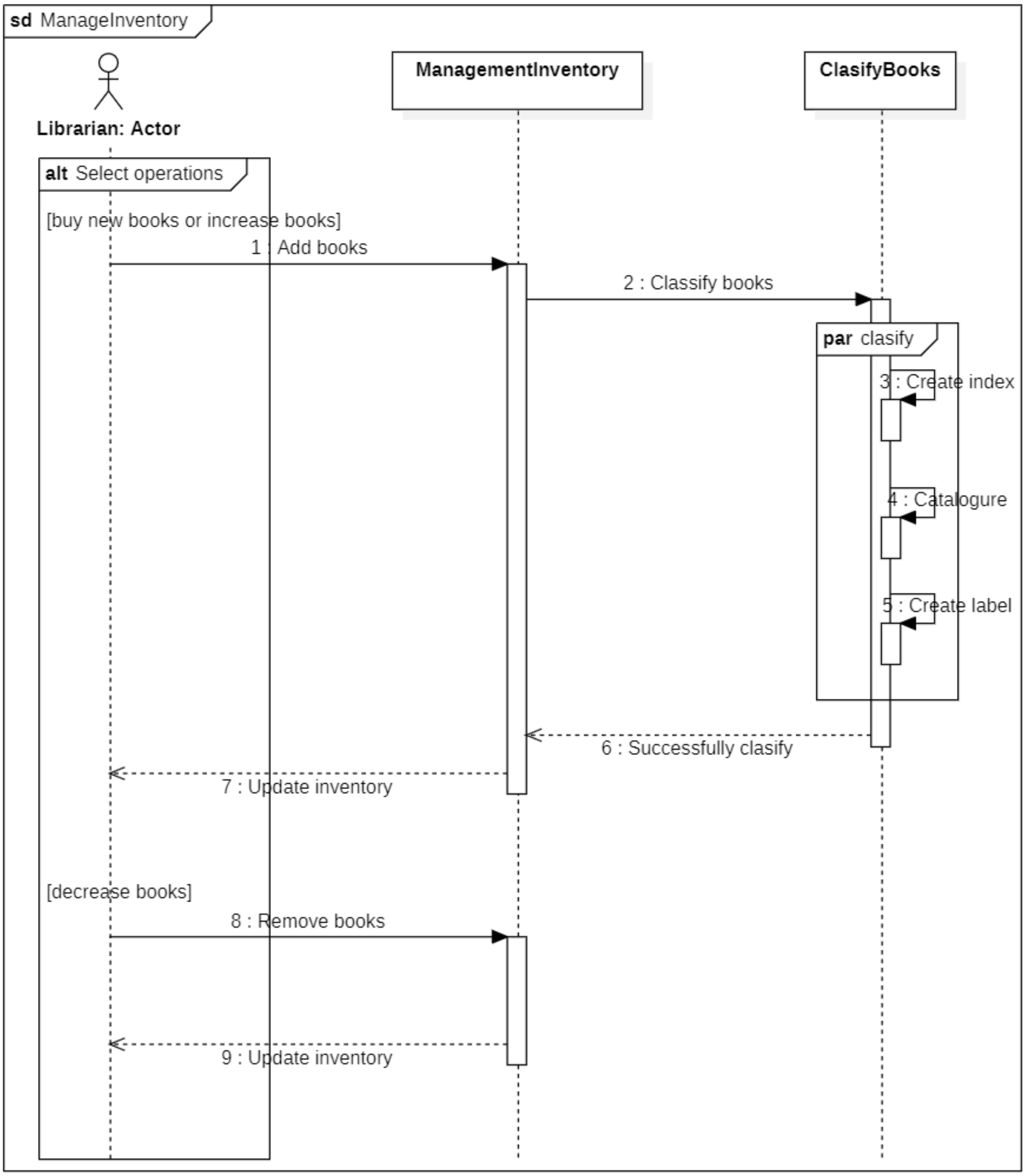
3.1.2.2. Librarian Get Suggestions

The process of administrator viewing suggestions is similar to the process of user sending suggestions, except that the request sent by the interface changes from a suggestion to a get request, and the data returned by the backend changes from success information to the content of the suggestion



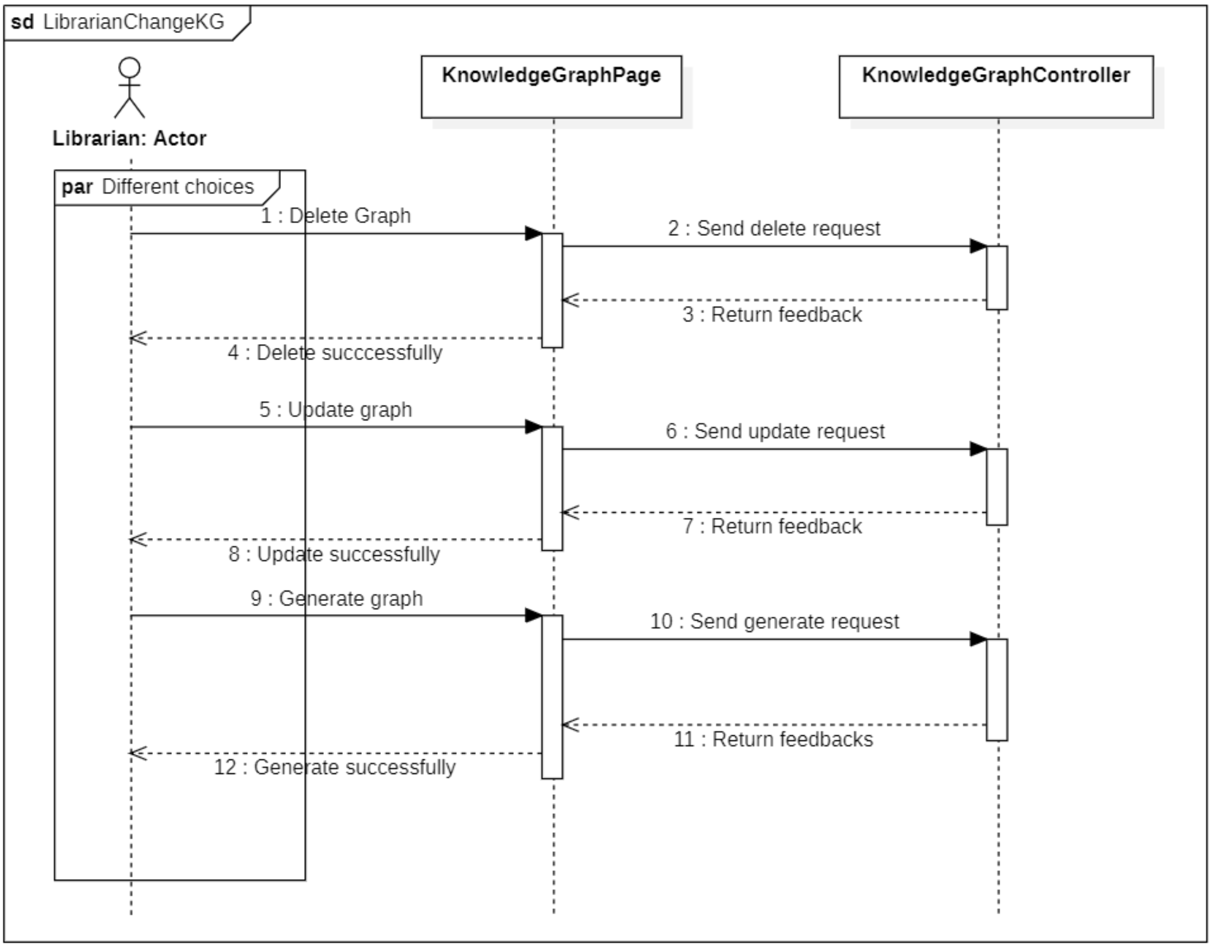
3.1.2.3. Manage Inventory

This sequence diagram shows the process by which a librarian receives a suggestion and then changes the book inventory based on the suggestion. The librarian first selects one from adding books and reducing books (adding books includes adding existing books and acquiring new books). If adding books is selected, the books are added and the newly added books are classified, tagged, indexed, etc., and then the information of the book inventory is updated; if reducing books is selected, the books are directly reduced and the book information is updated.



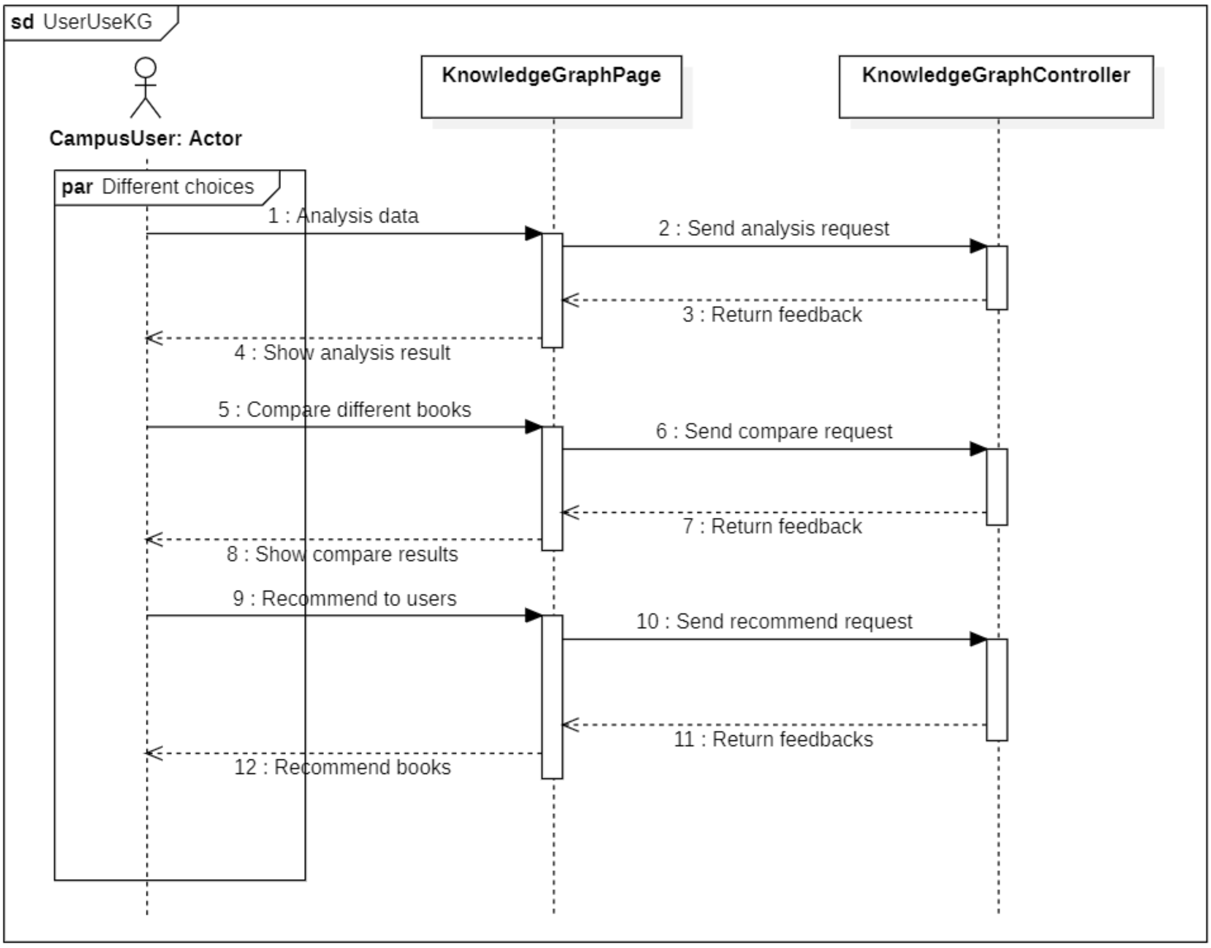
3.1.2.4. Librarian Change Knowledge Graph

This sequence diagram shows the various operations of the librarian on the knowledge graph. The librarian can delete, add, or update the knowledge graph, and all three processes are similar: the interface first sends a request to the backend, which receives the request and processes it logically, then returns the result to the interface and displays it.



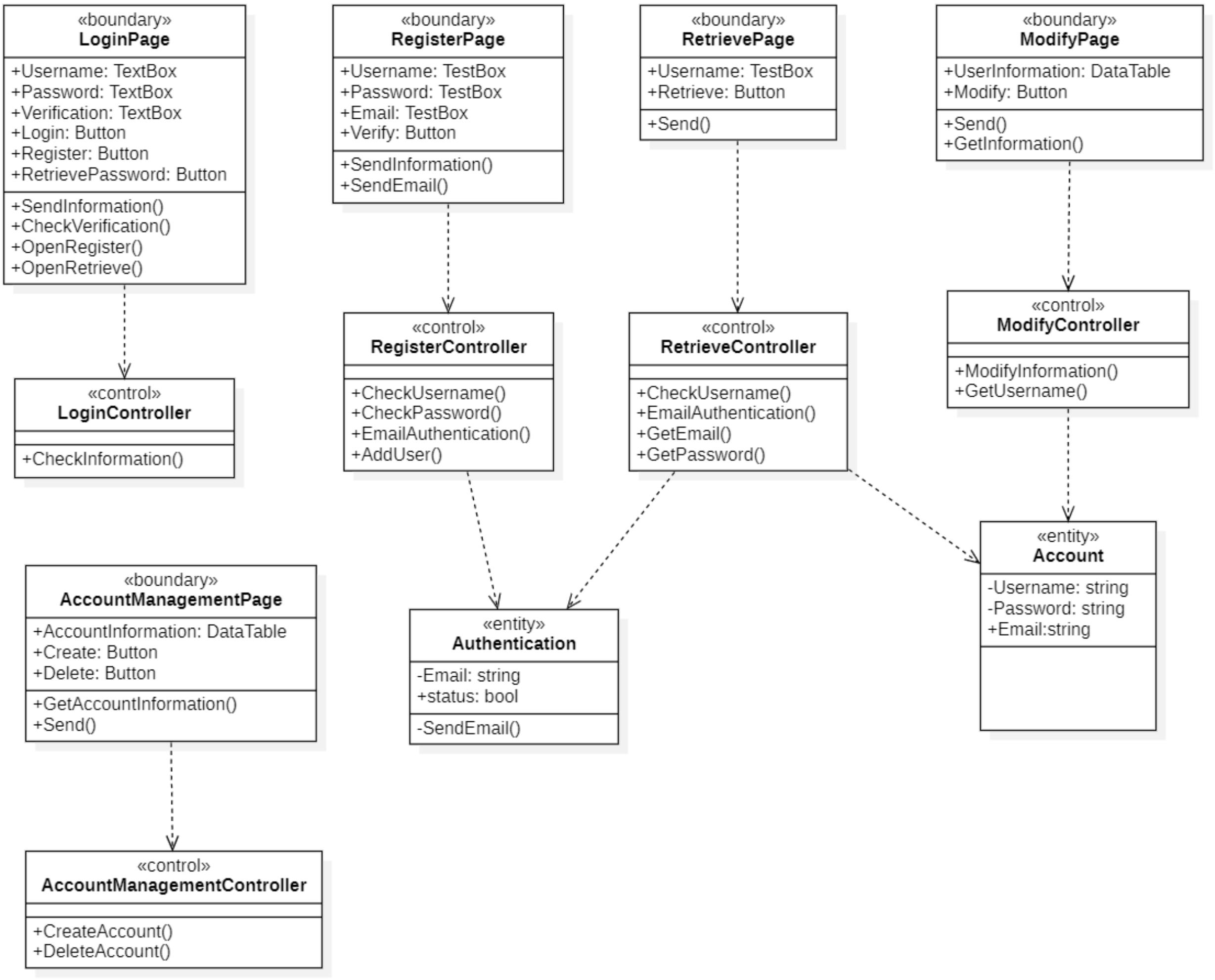
3.1.2.5. User Use Knowledge Graph

The process of using knowledge graphs by users is similar to that of librarians. Users can choose to view the current knowledge graph, use the knowledge graph to compare different books, analyze users' reading preferences based on their reading records and get recommended books, and other operations. All these operations are the process that the page initiates the request, the back-end processes it and returns the result and displays it.



3.2 Account Management System

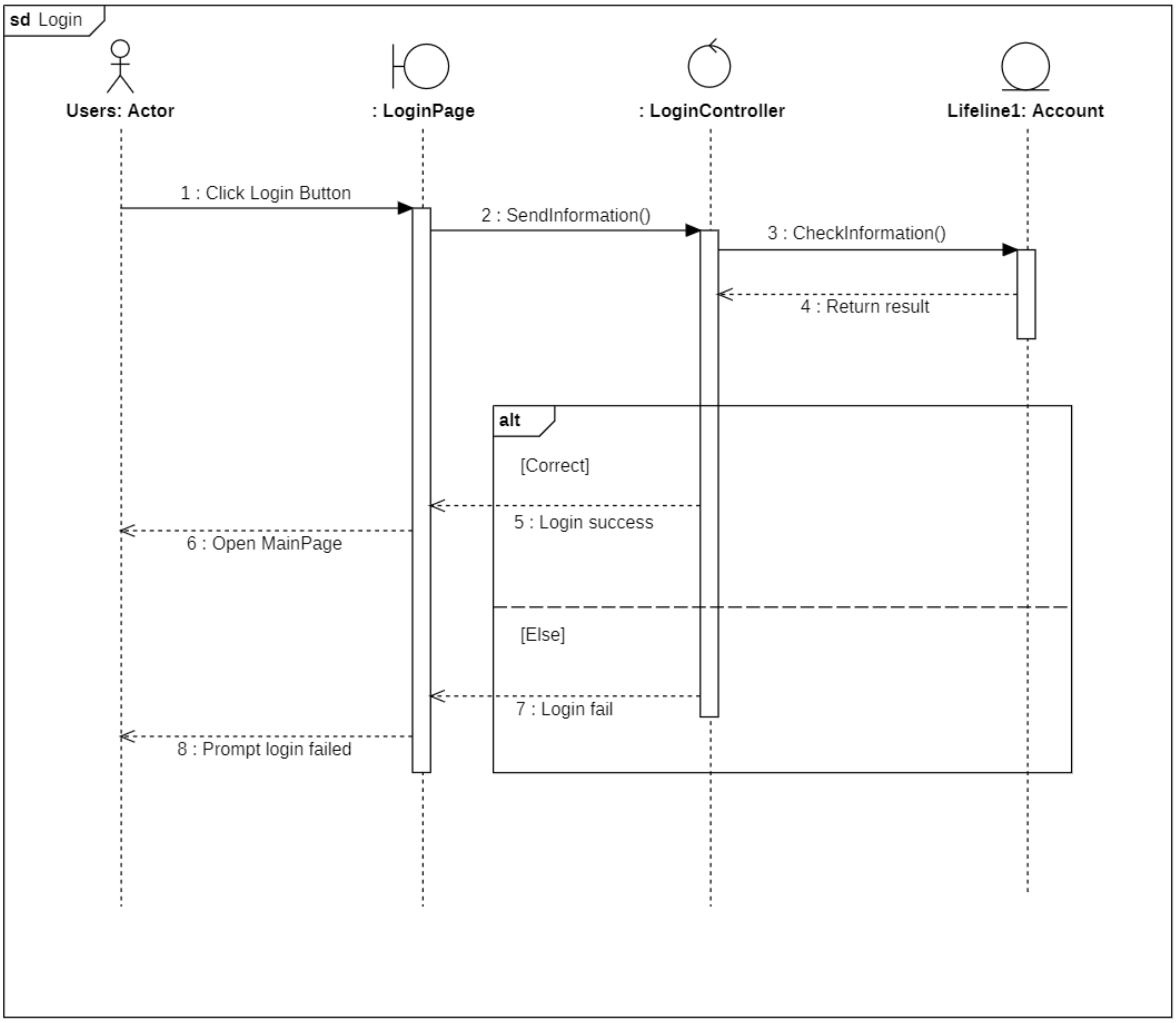
3.2.1 Class Diagram



3.2.2 Interaction Diagrams

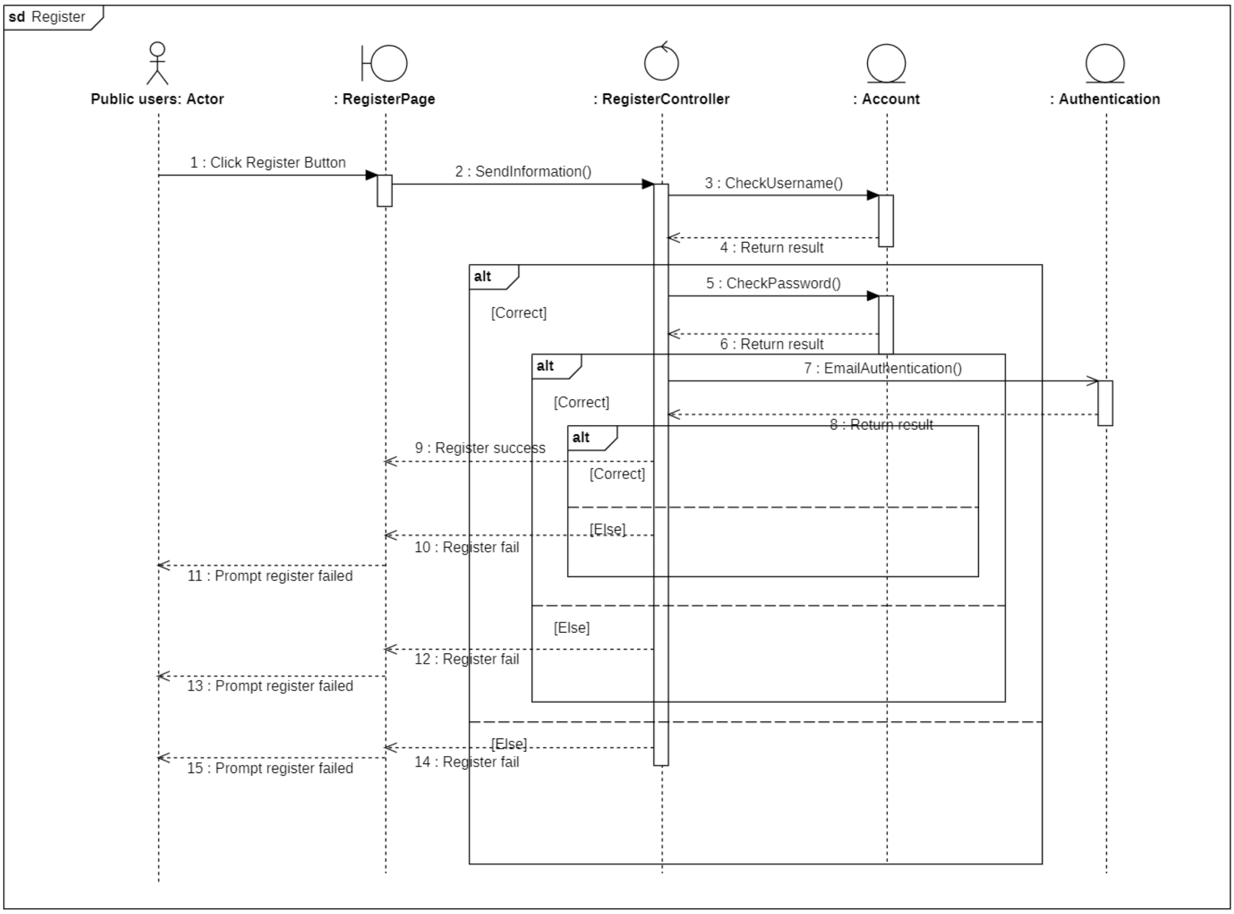
3.2.2.1 Users Login

This sequence diagram shows how users login to the system.



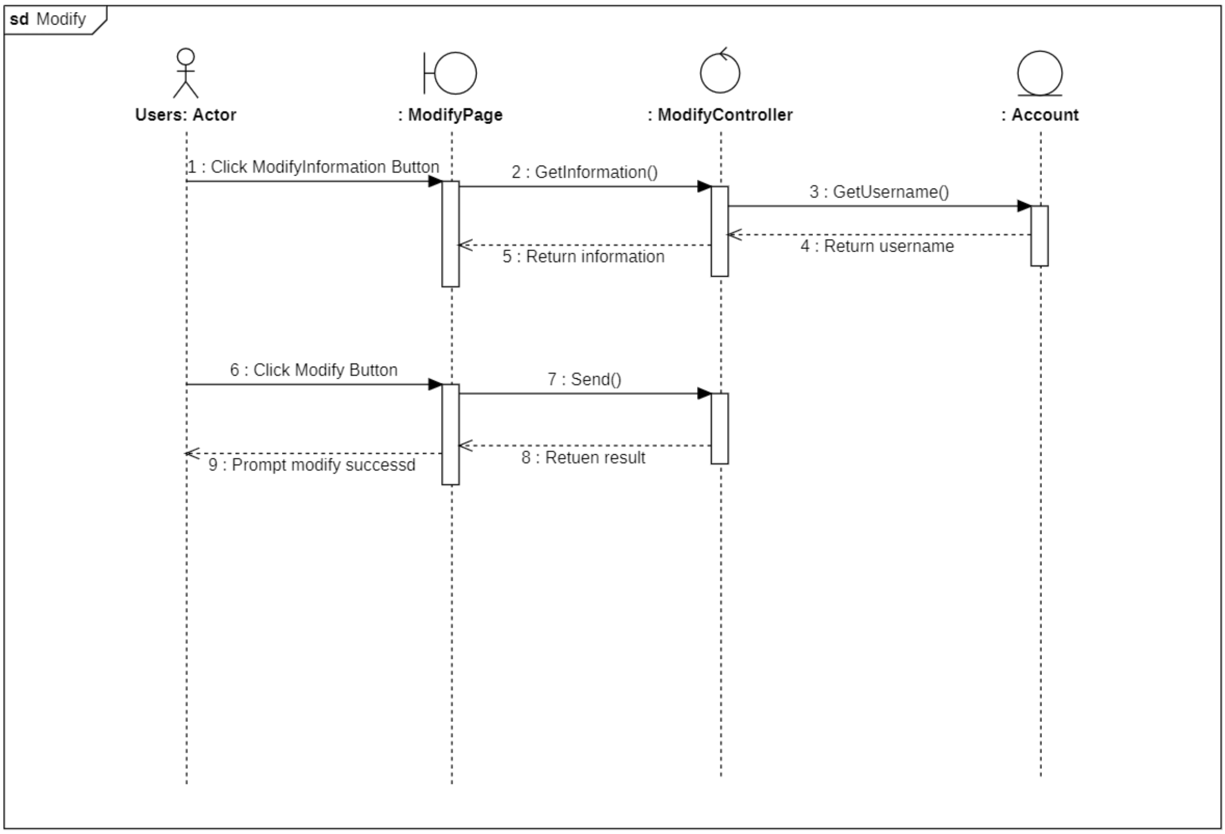
3.2.2.2 Public Users Register

This sequence diagram shows how a public user registers an account in the system.



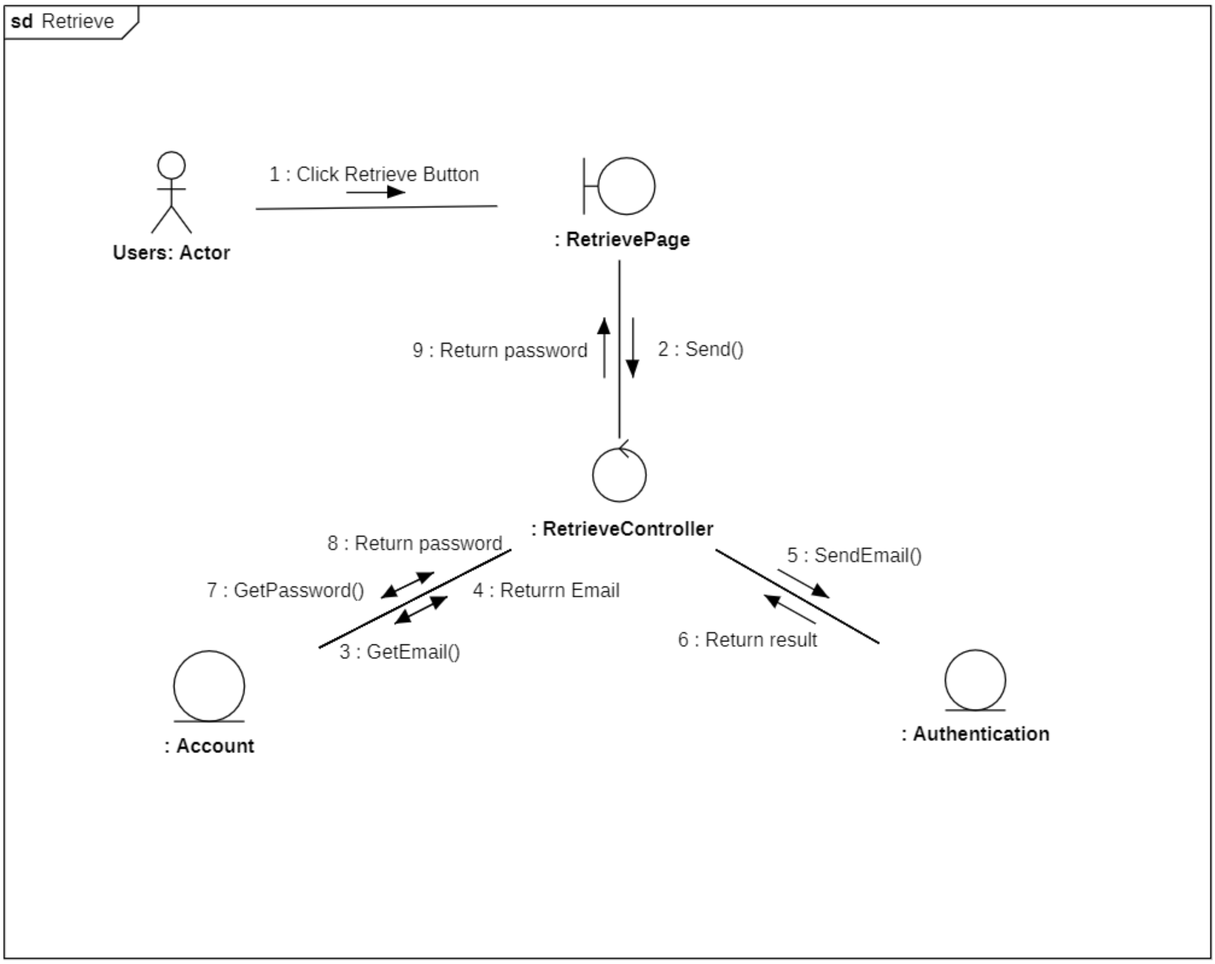
3.2.2.3 Users Modify Information

This sequence diagram shows how a user modify his information in the system.



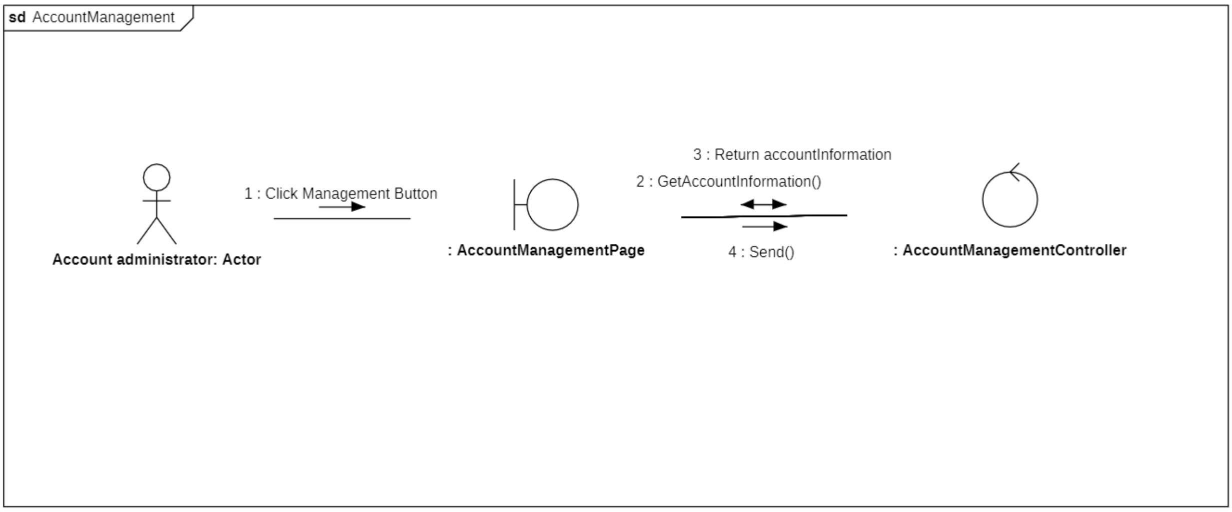
3.2.2.4 Users Retrieve Password

This communication diagram shows how a user retrieve his password in the system.



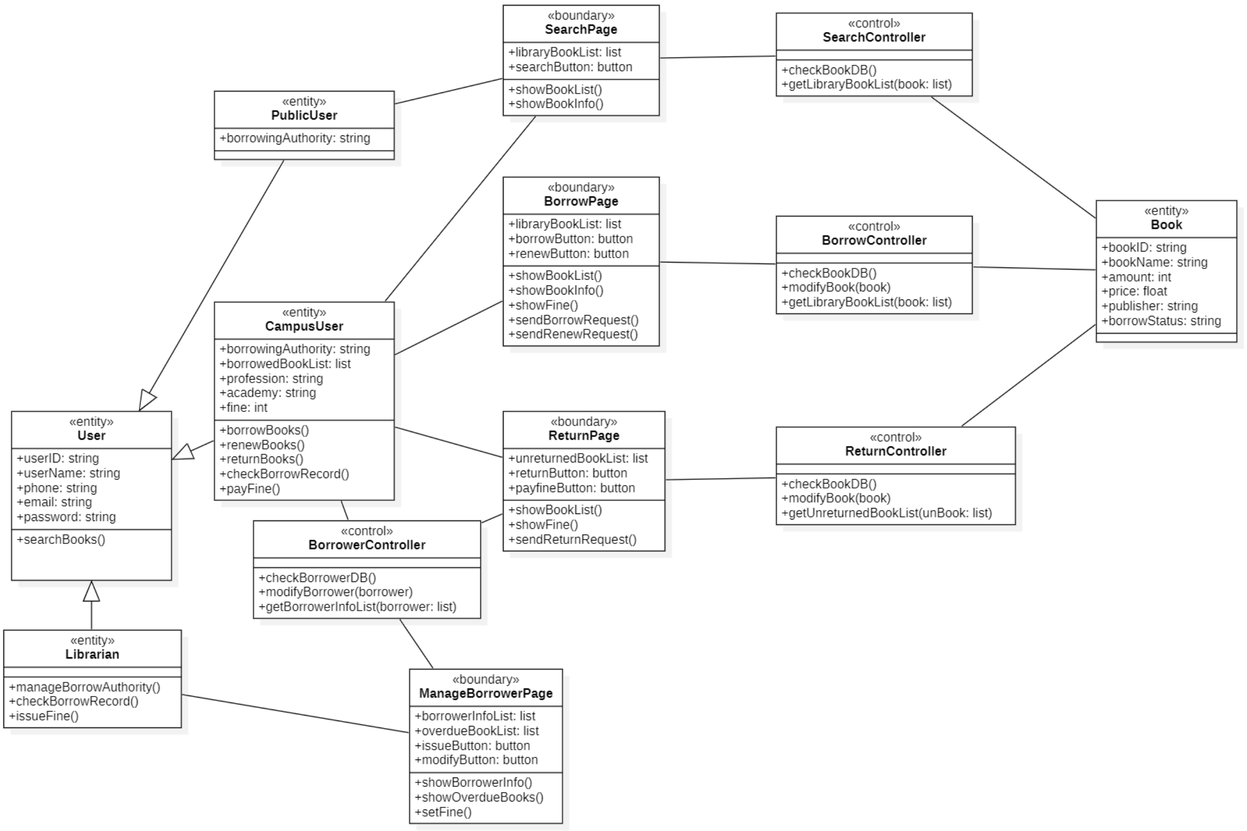
3.2.2.5 Account Administrator Manage Account

This communication diagram shows how an account administrator manages accounts in the system, including creating and deleting accounts.


sd AccountManagement
1 : Click Management Button
Account administrator: Actor
: AccountManagementPage
3 : Return accountInformation
2 : GetAccountInformation()
4 : Send()
: AccountManagementController

3.3 Book Borrowing Management System

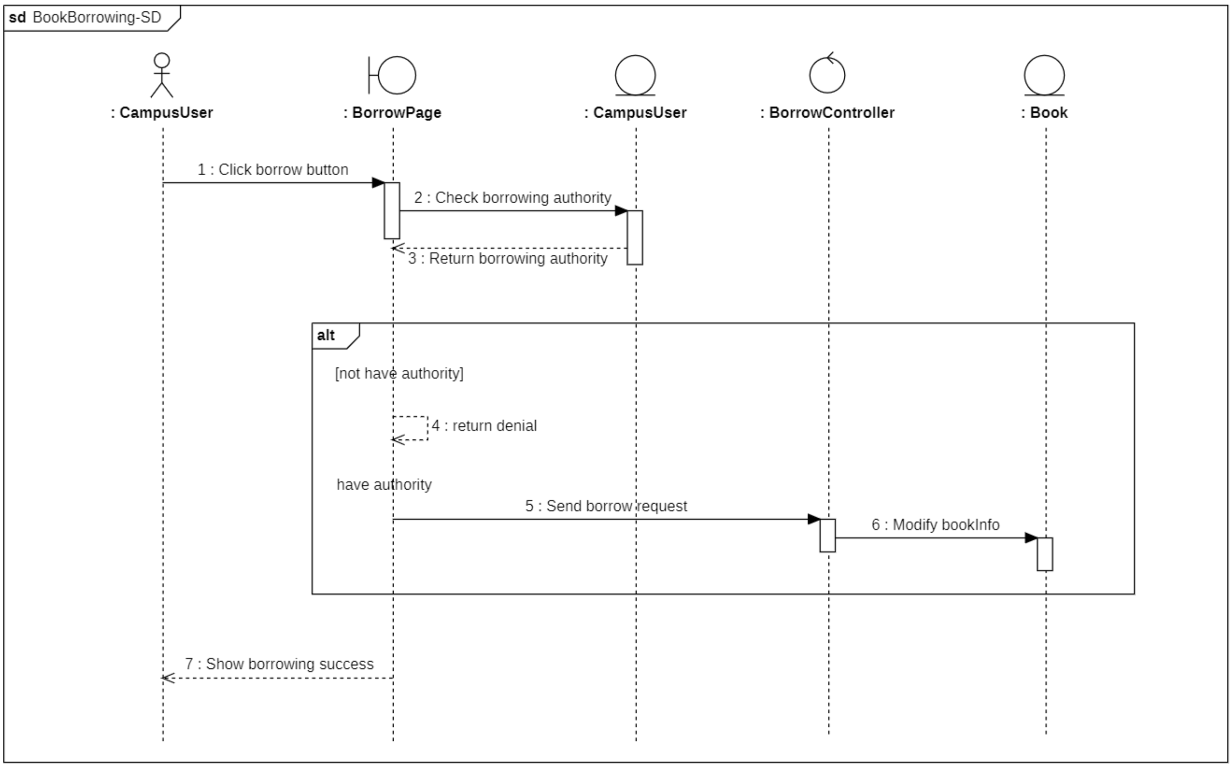
3.3.1 Class Diagram



3.3.2 Interaction Diagrams

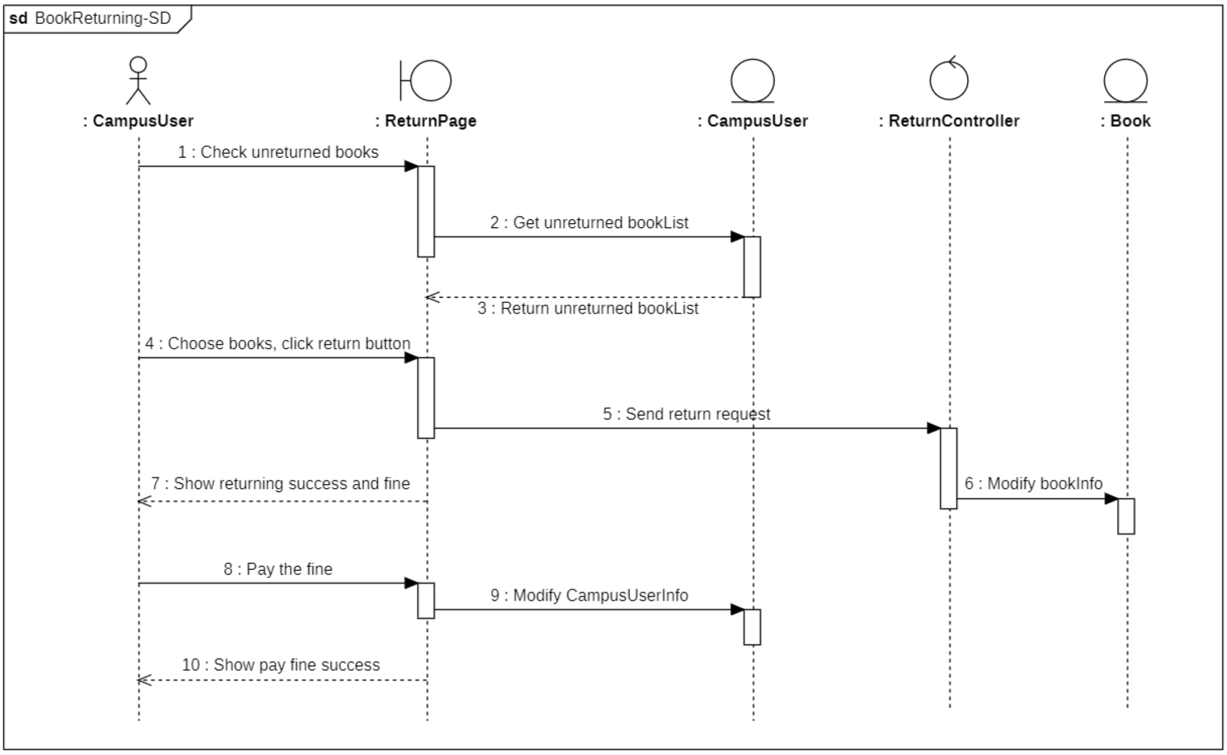
3.3.2.1 Borrow Books

This sequence diagram show how the borrowers borrow books.



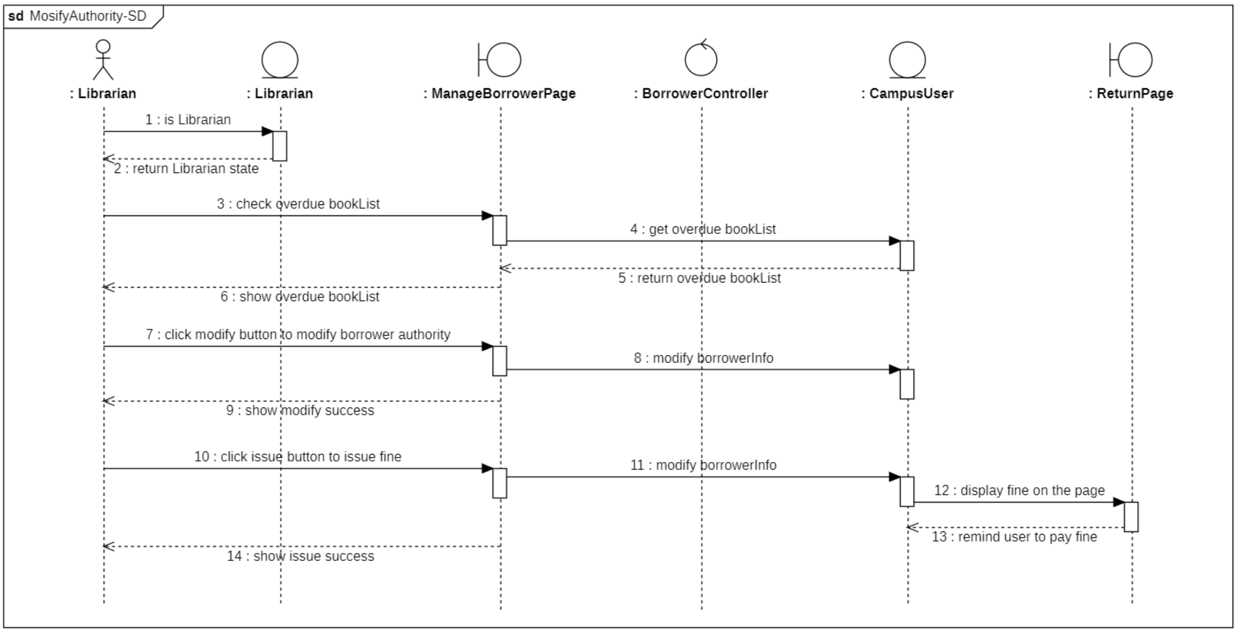
3.3.2.2 Return Books

This sequence diagram shows how the borrowers return the books.


sd BookReturning-SD
: CampusUser
: ReturnPage
: CampusUser
: ReturnController
: Book
1 : Check unreturned books
2 : Get unreturned bookList
3 : Return unreturned bookList
4 : Choose books, click return button
5 : Send return request
6 : Modify bookInfo
7 : Show returning success and fine
8 : Pay the fine
9 : Modify CampusUserInfo
10 : Show pay fine success

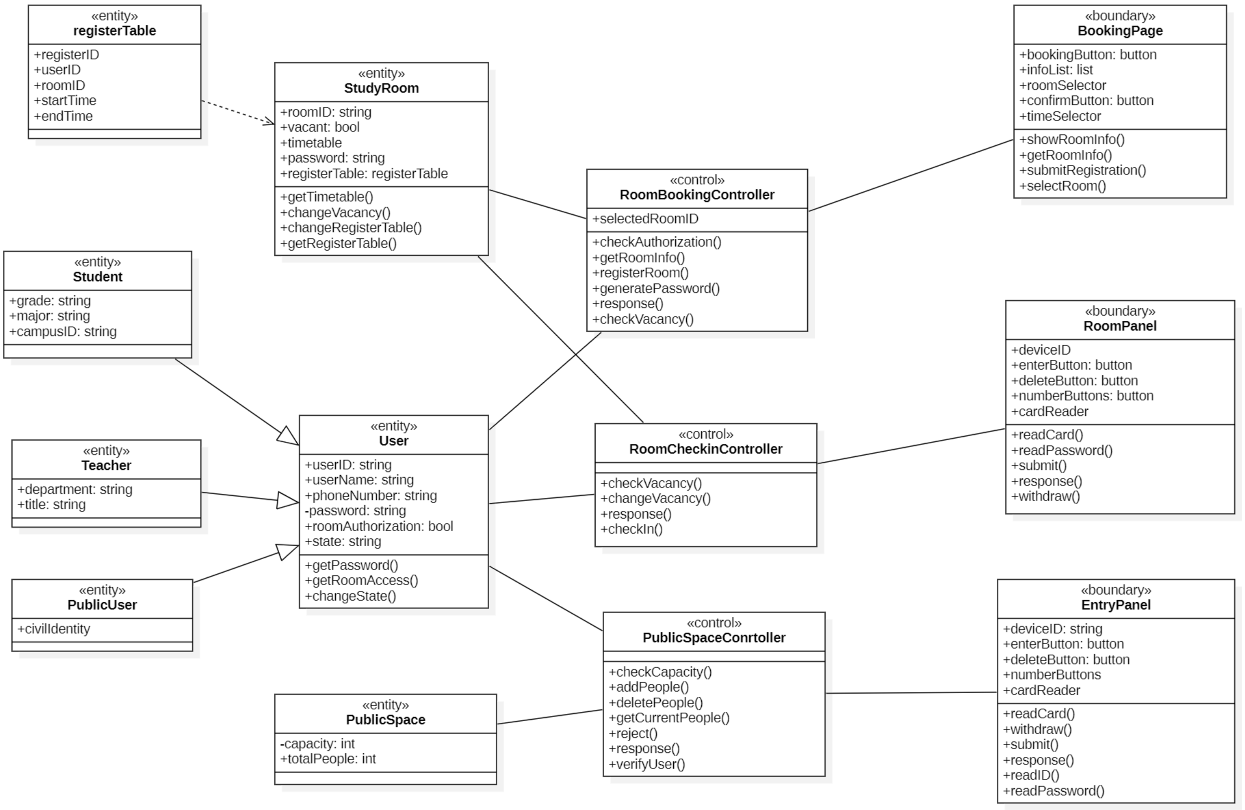
3.3.2.3 Modify Authority

This sequence diagram shows how the librarian modify the borrower's authority.


sd MosifyAuthority-SD
: Librarian
: Librarian
: ManageBorrowerPage
: BorrowerController
: CampusUser
: ReturnPage
1 : is Librarian
2 : return Librarian state
3 : check overdue bookList
4 : get overdue bookList
5 : return overdue bookList
6 : show overdue bookList
7 : click modify button to modify borrower authority
8 : modify borrowerInfo
9 : show modify success
10 : click issue button to issue fine
11 : modify borrowerInfo
12 : display fine on the page
13 : remind user to pay fine
14 : show issue success

3.4 Venue Management System

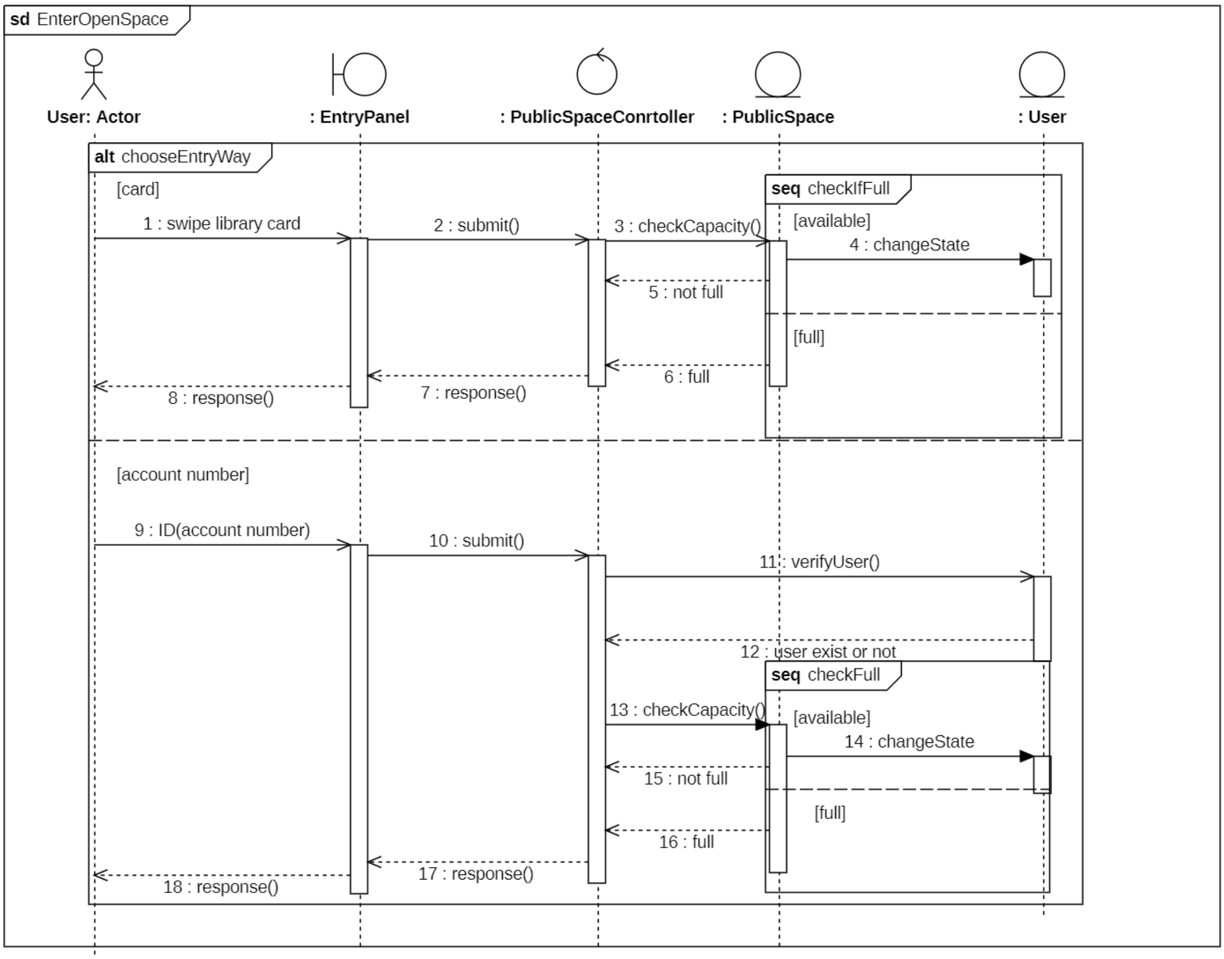
3.4.1 Class Diagram



3.4.2 Interaction Diagrams

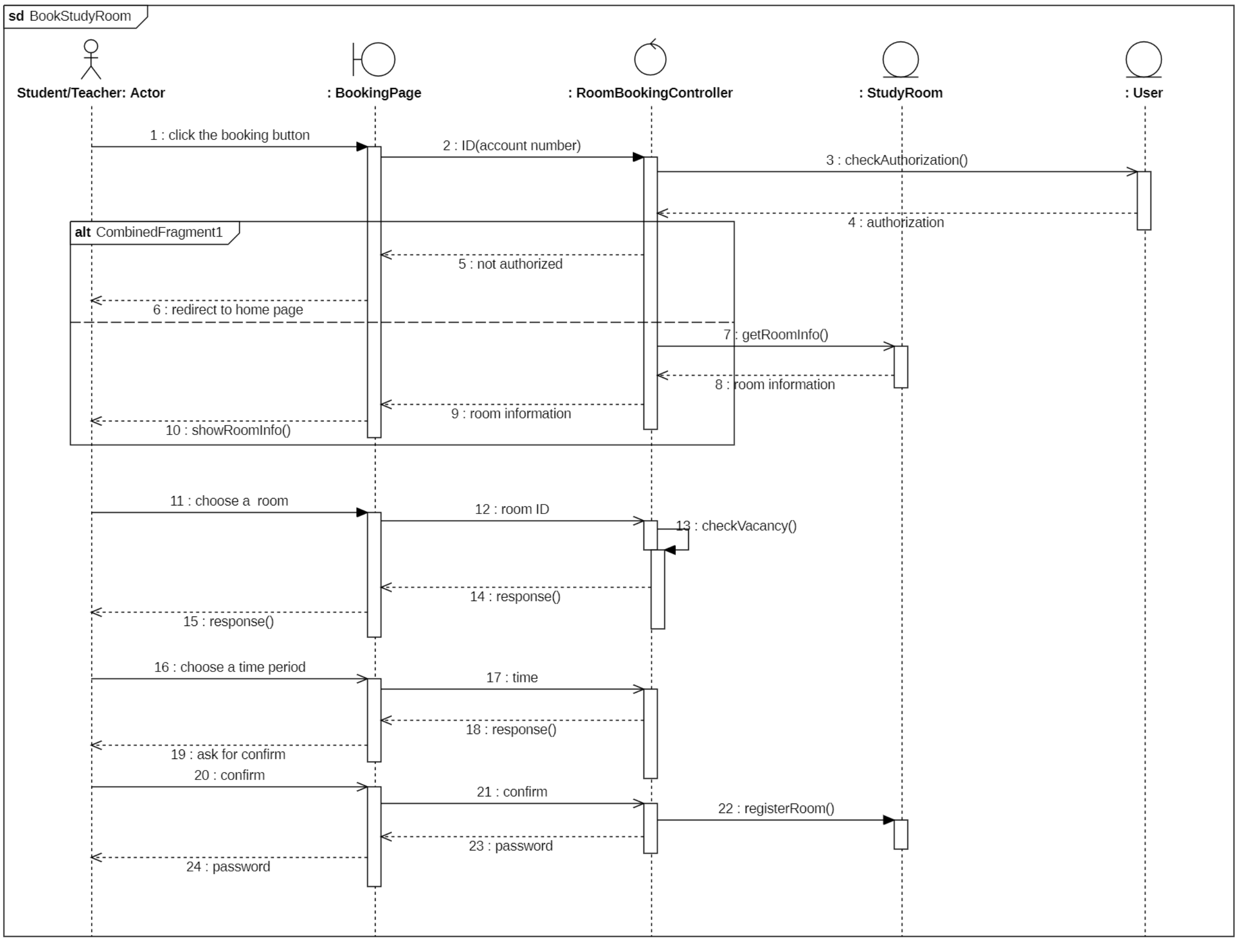
3.4.2.1 Enter Open Space

This sequence diagram shows how a user can enter open spaces of a University library. It demonstrates the data flow between the system and the user. They were connected by the interface Entry Panel. A controller is designed to arrange the data flow.



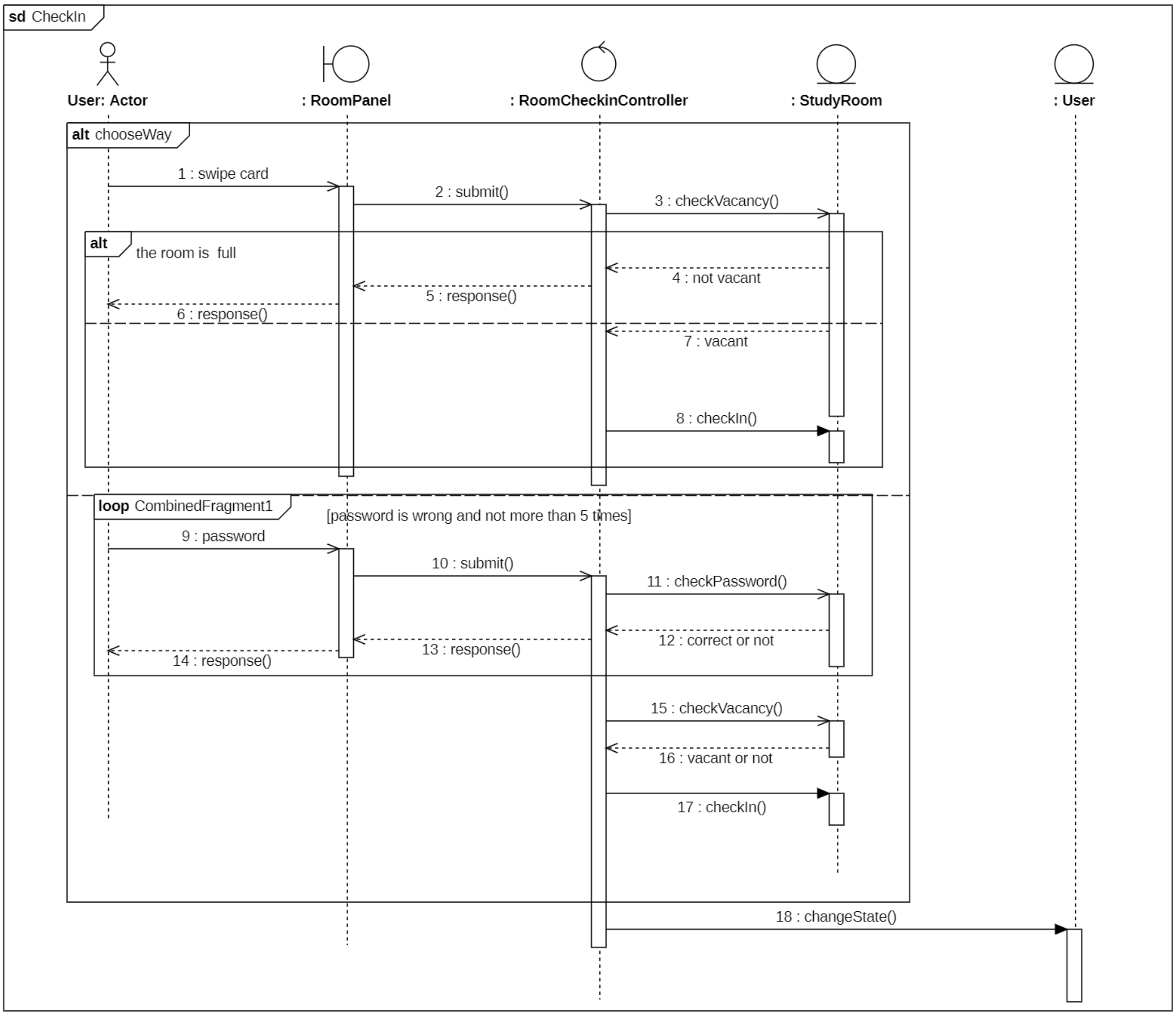
3.4.2.2 Book a Study Room

This sequence diagram shows how some of the users(teachers and students) can book a study room in the library. It demonstrates the data flow between the system and the user. They were connected by the interface of a booking webpage. A room booking controller is designed to arrange the data flow.



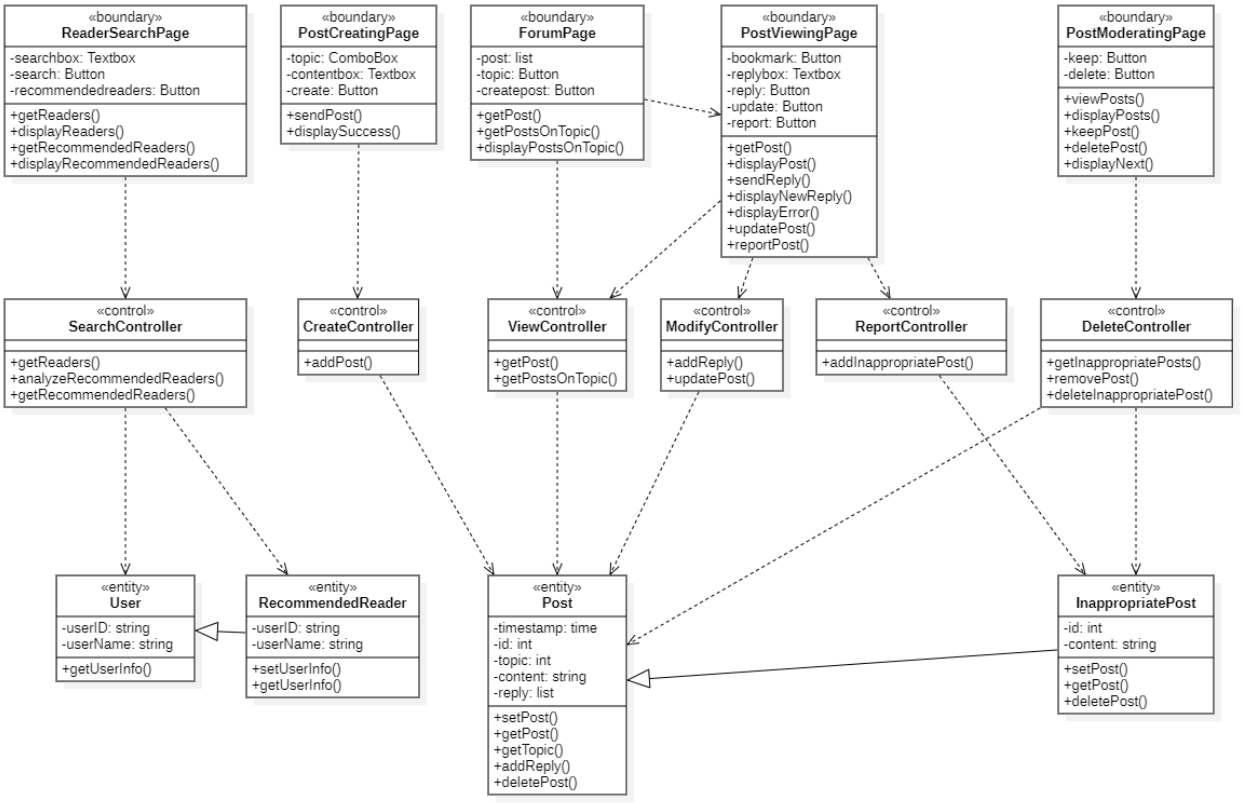
3.4.2.3 Check in to a Study Room

This sequence diagram shows how user who has already book a room can check in to it. The diagram demonstrates the data flow between the system and the user. They were connected by the interface of a room panel. A room check in controller is designed to arrange the data flow.



3.5 Reader Communication System

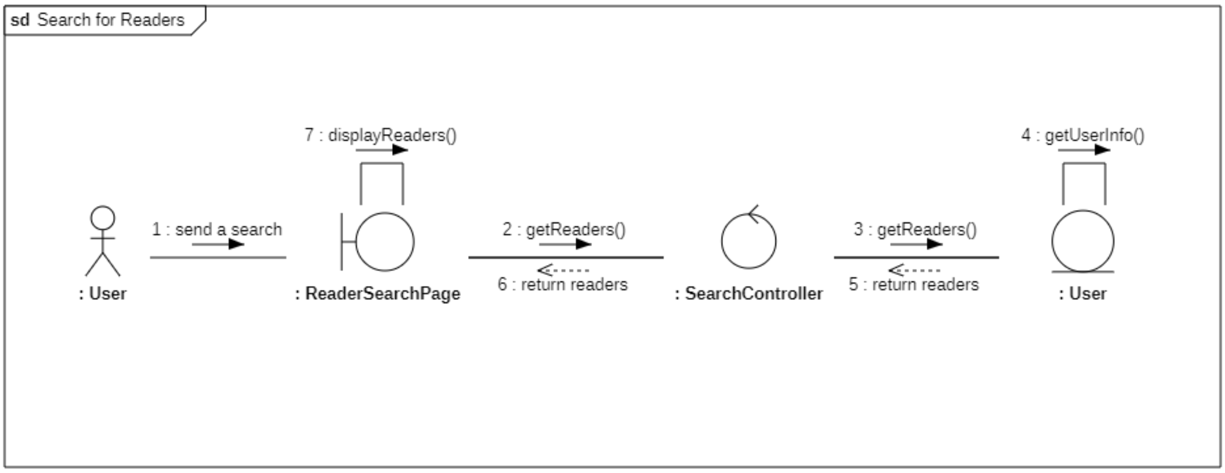
3.5.1 Class Diagram



3.5.2 Interaction Diagrams

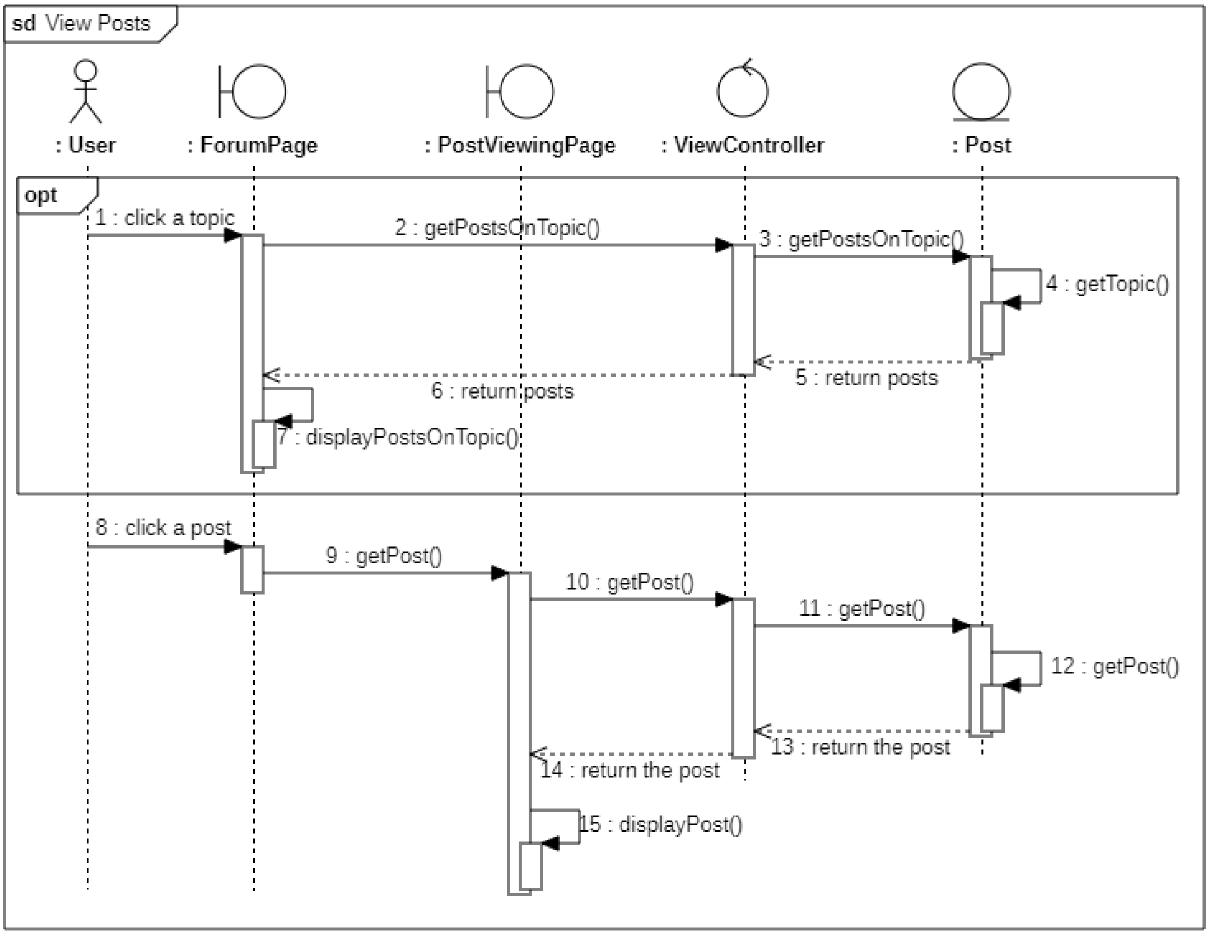
3.5.2.1 Search for Readers

This communication diagram shows how the objects in the system interact when a user sends a search for readers.



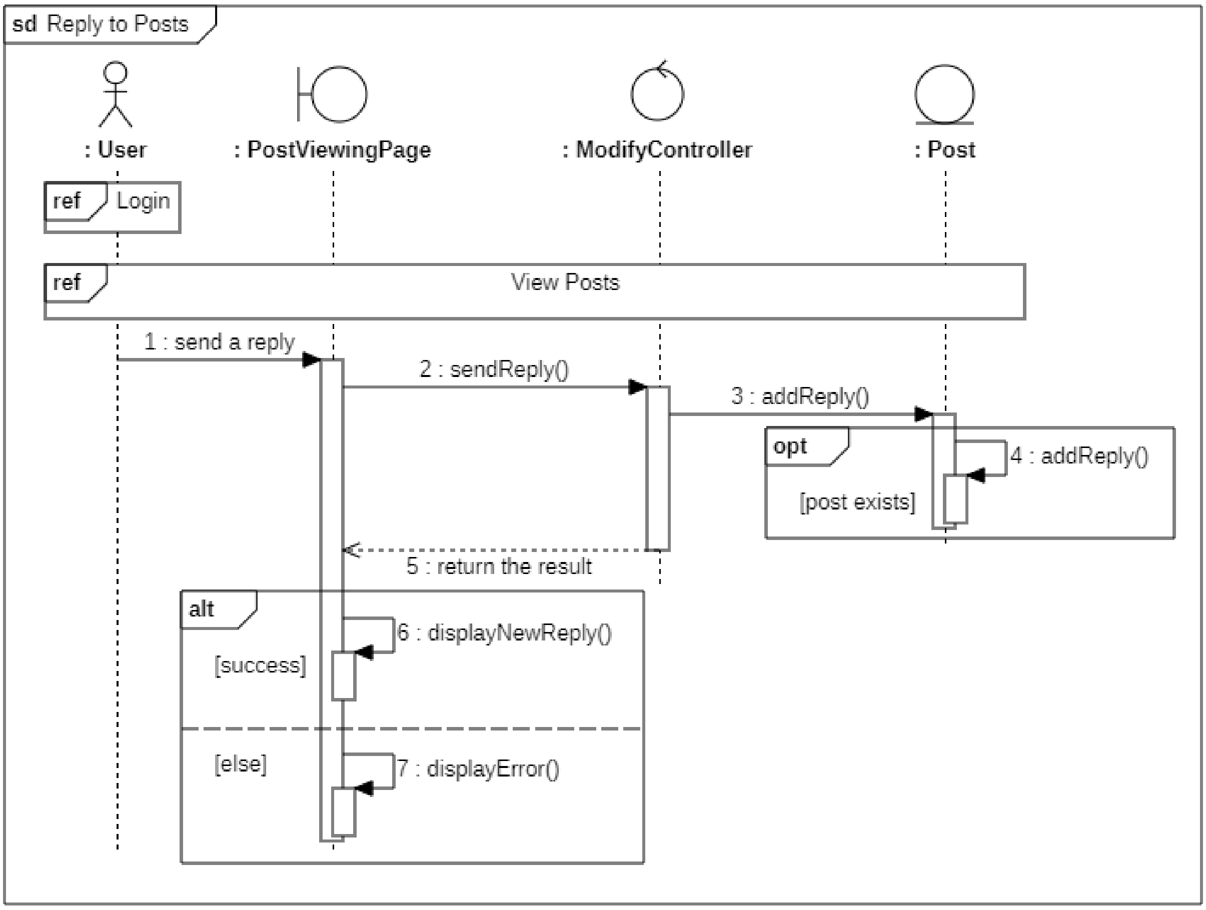
3.5.2.2 View Posts

This sequence diagram shows how the system processes a user's request to view a post. The user can choose a topic first.



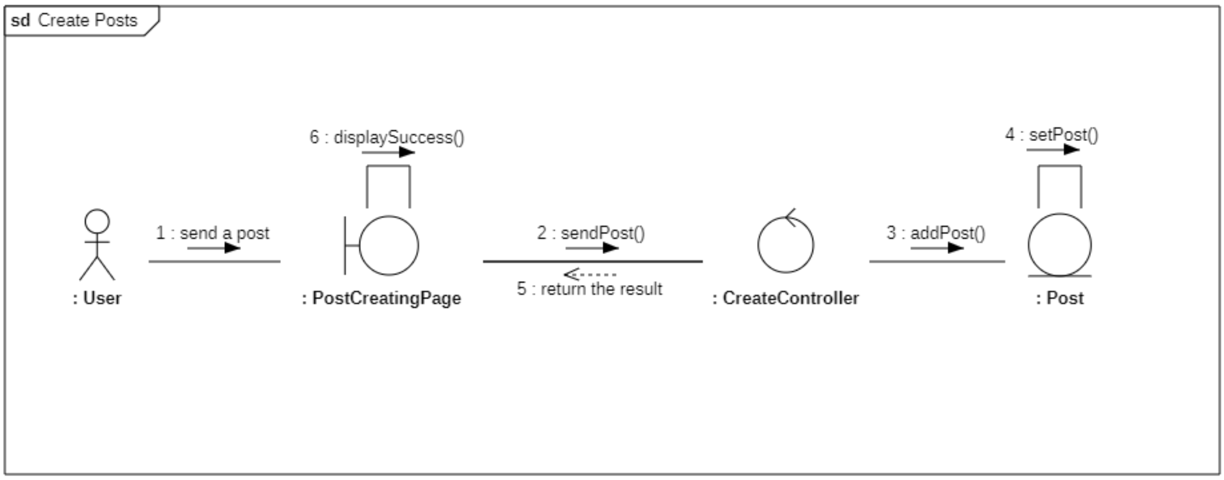
3.5.2.3 Reply to Posts

This sequence diagram shows how the system handles a user's reply to a post. The page displays a success or error message depending on the result.



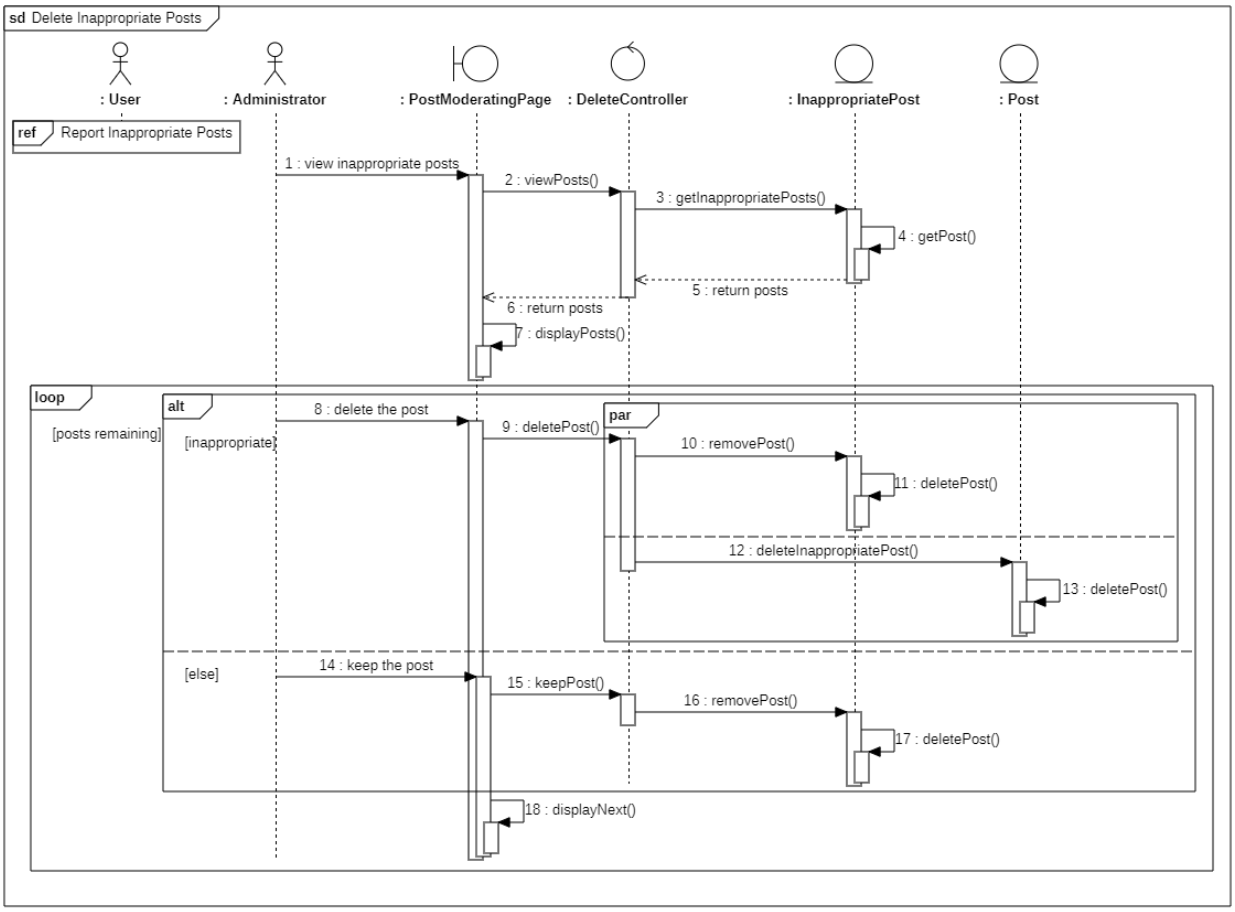
3.5.2.4 Create Posts

This communication diagram shows how the system creates a new post when a user submits one.



3.5.2.5 Delete Inappropriate Posts

This sequence diagram shows how an administrator deletes inappropriate posts. The administrator views the reported posts first, and decides whether to keep the post or delete it. The loop fragment shows that the administrator repeats the decision making process until there is no more post to moderate.

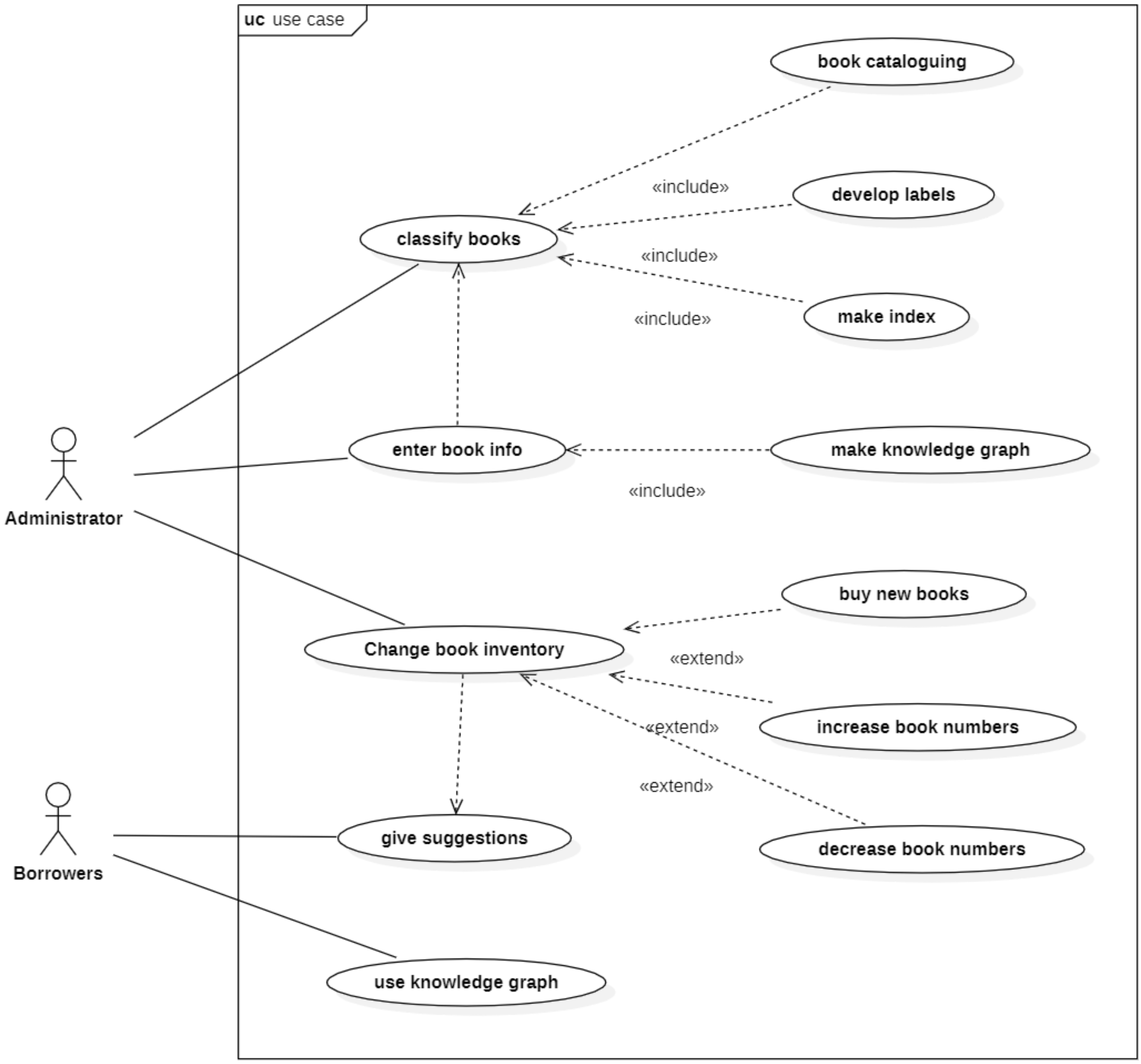


4. Updated Use Case Model

4.1 Updated System: Book Management System

In the last assignment, in terms of the knowledge graph, we only considered the librarian's modifications to it. For this assignment , we found that users also have the need to use the knowledge graph, so we added the function to use the knowledge graph to the use case model.

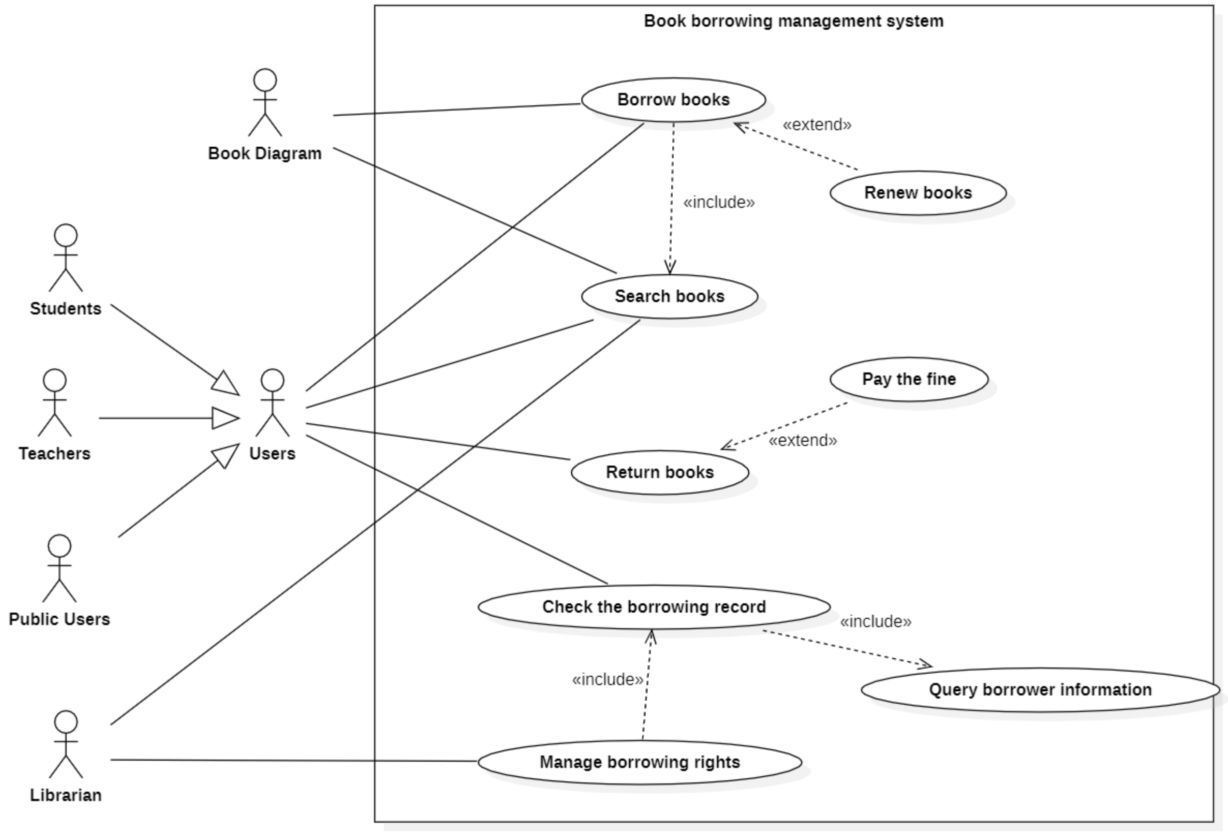
The updated use case diagram is as follows.



4.3 Updated System: Book Borrowing Management System

In the past use case diagram, the borrower has a use case called "reservebooks", which is a function that is not practical, and we have deleted it in this assignment. In the past use case diagram, Librarian directly contacted the use case called "Check the borrowing record", which is not reasonable, and we changed it to "Manage borrowing authority" include "Check the borrowing record" in this assignment

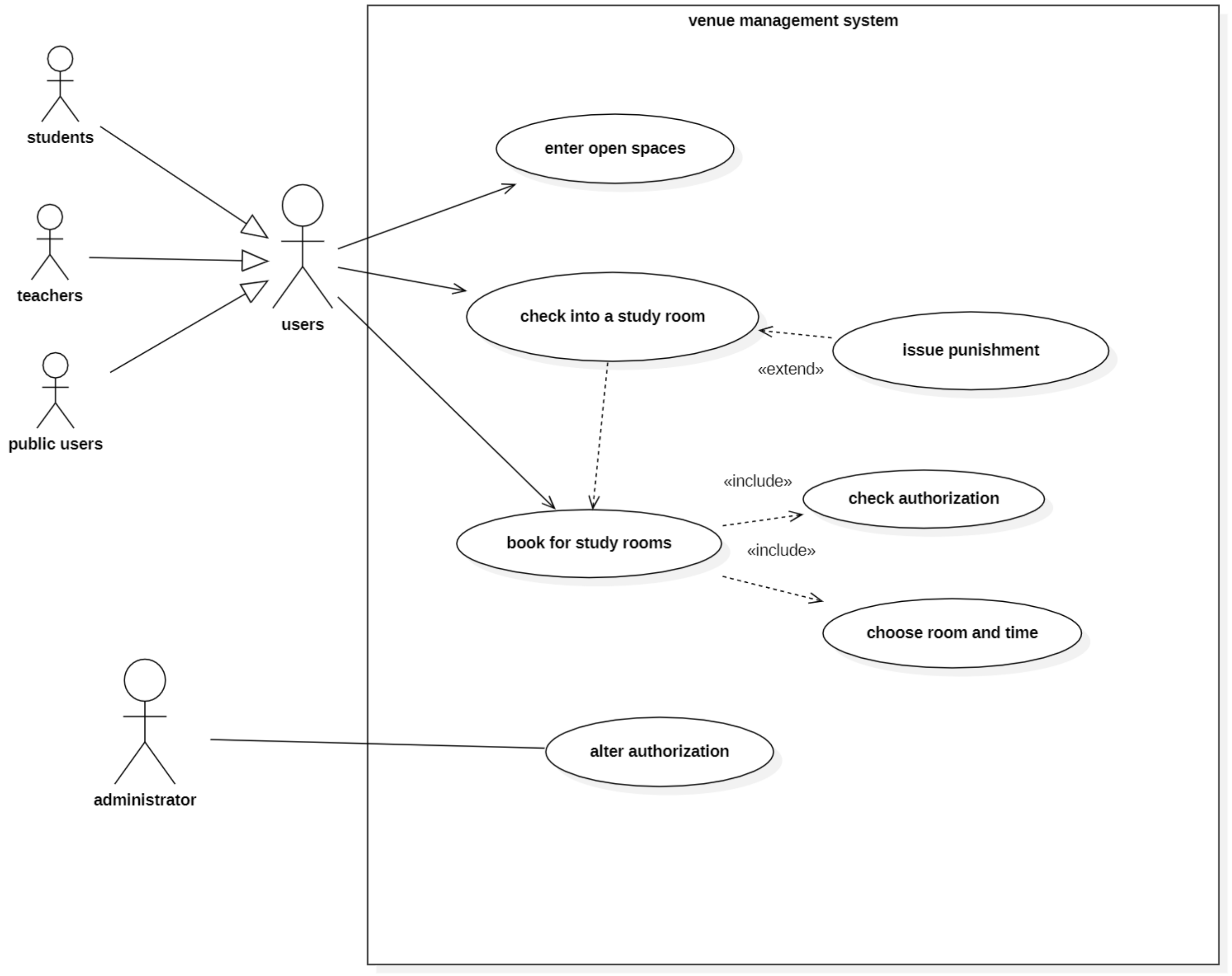
The updated use case diagram is as follows.



4.4 Updated System: Venue Management System

Based on the problems raised in the class two major changes have been made in the venue management system's use case diagram. In the past use case diagram, there is an arrow between check into a study room and enter open spaces, indicating the dependency relation. This arrow is now deleted because this dependency doesn't need to be realized in the system. It is promised physically. The authorization list is also deleted because it is not a actor of any use case, it is a detail that should be put in the detail specification.

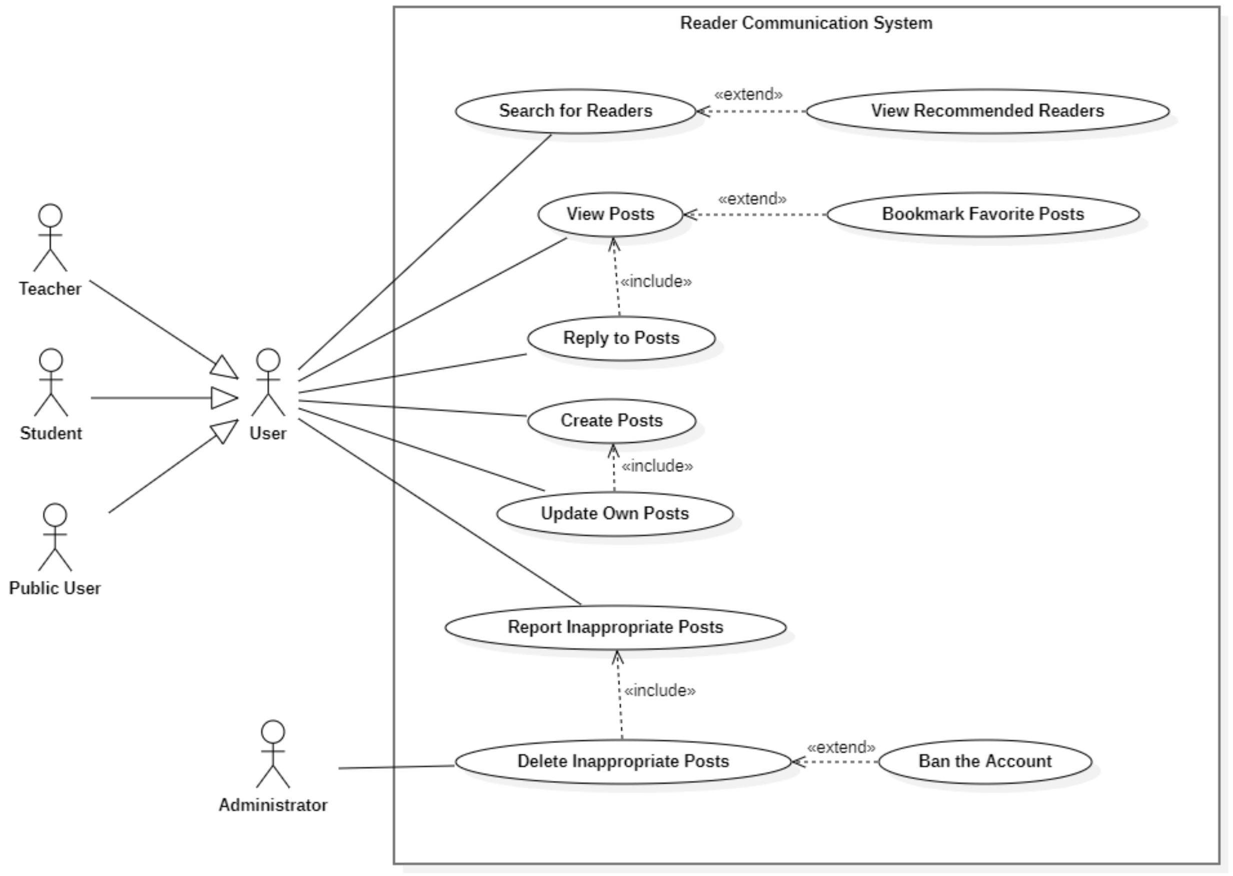
The updated use case diagram is as follows.



4.5 Updated System: Reader Communication System

In the last assignment, we grouped the use cases “Reply to Posts” and “Create Posts” under the general term “Share Thoughts”. However, we realized that this was not appropriate because these two use cases are quite different from each other. Moreover, this would also increase the complexity. Therefore, we removed the use case “Share Thoughts”.

The updated use case diagram is as follows.



4.6 Use Case Detailed Specification

System: Venue Management System

Use Case: Enter Open Spaces

Update: In the previous report the alternative flow of this use case is miswritten. The updated version corrected it.

USE CASE	ENTER OPEN SPACES
ID	UC07
Specification	This use case happens when a user wanted to enter the library building in order to borrow and return books, and use the facilities like tables and study rooms. There is no limitation unless the total number of people inside reaches the threshold.
Actors	**User**
Pre-Condition	User's account must be valid and active

USE CASE	ENTER OPEN SPACES
Basic Flow	1. The user reaches the turnstile at the front gate of the library building. 2. User verifies his or her identity at the smart device attached to the turnstile. 2.1 If the user has a library card, he or she could swipe the card. 2.2 The user could also enter his or her account number on the panel to get verification. 3. The user walks through the turnstile and enters the open spaces.
Alternative Flows	a. The user is not able to go through the verification process: a.1. If the user enters an invalid account number, he or she is allowed to renter the account number. a.2. If the user's card can't be recognized, he or she should turn to a librarian for help. b. The total number of people inside the building reaches the threshold. All the users outside are not allowed to go through the turnstile
Post-Condition	After successfully entering the open spaces, users could see the books on the bookshelves, use public PCs to log in to the system and borrow books, use the tables and check in to a study room.

System: Venue Management System

Use Case: Check in to a Study Room

Update: The previous report doesn't contain the case when the card and the room doesn't match. It is now added to the report.

USE CASE	CHECK IN TO A STUDY ROOM
ID	UC09
Specification	Users must check in to a study room at the registered time. The user could either wipe the library card or enter the temporary password he or she get after booking. The room door will be unlocked after the check-in process.
Actors	**Teacher or student**
Pre-Condition	The user must have successfully booked a spare study room already. The user must have already entered the open spaces.

USE CASE	CHECK IN TO A STUDY ROOM
Basic Flow	1. Users reach the door of the room he or she booked. 2. Users check-in: 2.1. If the user has a library card, he could swipe the card to check in. 2.2. If the user doesn't have a card, he could enter password through the panel. 3. The system unlocks the door so that the user could get into the study room.
Alternative Flows	a. If the user enters the wrong temporary password: he will be given 5 more times to enter the correct password, otherwise, he could not check-in and use the room. b. The user check-in 20 minutes later than the booked time spot: The user will be punished, he could not use the study room for the next 5 days. c. If the card does not match the room, or the room is no booked by this certain user now, the user is informed.
Post-Condition	none

5. Annotated References

5.1 Application Architecture Guide（https://developer.android.google.cn/jetpack/guide?hl=zh-cn）

This webpage provides an overview of the common application architectures for Android application development. It guides developers through a series of common architectural principles, recommended application architectures, best practices, and real-life application examples. The layered architecture and the user interface design used in this project are based on this design guideline.

5.2 30 minutes of learning the UML class Fig （https://zhuanlan.zhihu.com/p/109655171）

This web page explains in detail the basics of class diagram and the use of class diagram.It expounds the representation of specific classes, abstractions, interfaces and packages in class diagram and six relationships in class diagram, guiding the developers to correctly build class diagram in UML.We designed the class diagrams based on this article in this assignment.

5.3 Entity Class, Boundary Class and Control Class (https://blog.csdn.net/u012372720/article/details/71171660)

This article details the knowledge of entity, boundary, and control classes.The paper expounds the distinction and connection between the entity class, boundary class and control class, and tells how to abstract the entity class, boundary class and control class in the system.The article helps us to have a deeper understanding of classes by giving examples.We refer to this article when dividing the classes in this assignment.

5.4 Study notes of UML Sequence Diagram (https://blog.csdn.net/fly_zxy/article/details/80911942)

This article introduces the basics of the UML sequence diagram from various aspects.It tells the basic elements and their respective usage in the sequence diagram, the classification and corresponding usage of the combined fragments, and helps us to understand the steps of drawing the sequence diagram.We refer to this article for the design of the sequence diagram in this assignment.

6. Contributions of Team Members

1. Project goals: Ding Yimin and Li Haocheng
2. Progress and Current Status: Ge Yuyang and Li Haocheng
3. System Detailed Analysis: Ding Yimin
4. Hierarchical Architecture Design: Ge Yuyang, Ding Yimin and Wang Xinglin
5. Analysis model: All members in group participate in model analysis, working on their own subsystem
6. Updated use case model: All members in group updated their own subsystem
7. Use Case Specification: Ding Yimin
8. References: Wang Xinglin