Smart Digital Library

Metaverse Dido

Software Design Specification

2021.10.31.

Introduction to Software Engineering

TEAM 15 (Metaverse Dido)

Team Leader	Keunha Kim
Team Member	Minsu Kim
Team Member	Gyeonghyeon Cho
Team Member	Ukcheol Choe
Team Member	Aizat Hamizuddin Bin Azlan
Team Member	Fatdzirul Izzat Bin Abdul Radzi
Team Member	Dasol Lee

LIST OF FIGURES

LIST OF Tables

1. Preface

This chapter contains the readership, scope, objective of this document and the document structure of this Software Design specification for “Metaverse Dido”.

1.1. Readership

This Software Design Document is divided into 10 sections with various subsections [Section 1.4] . Team 15 is the main reader for this document, but professor, TAs and student in SWE course.

1.2. Scope

This Software Design document is used to provide descriptions of the designs to implementing the Virtual digital library on “VRCHAT” platform and subsystems.

1.3. Objective

The main objective of this Software Design Specification document is to explain the design aspects for the Metaverse Dido. This document contains overall software architecture of project. All of aspects will be shown as detailed functions and several forms of diagrams.

1.4. Document Structure

- **1. Preface:** This chapter provides expressions and tools for this document, UML diagrams and the references, and scope of the project.
- **2. Introduction:** This chapter provides expressions and tools for this document, UML diagrams and the references, and scope of the project.
- **3. Overall System Architecture:** This chapter provides the overall architecture of the system with system organization and several diagrams.
- **4. System Architecture - Frontend:** This chapter provides overall architecture of frontend system with diagrams.

- **5. System Architecture - Backend:** This chapter provides overall architecture of backend system with diagrams.
- **6. Protocol Design:** This chapter provides protocol design for communicate between user client and server, database.
- **7. Database Design:** This chapter provides database design for the system data structures with diagrams and SQL DDL.
- **8. Testing Plan:** This chapter provides development, release and user testing plan.
- **9. Development Plan:** This chapter provides development environment and tools to develop the system. It also contains constraints, assumption, and dependencies for system.
- **10. Supporting Information:** This chapter provides basic form of document and change history of document.

2. Introduction

The purpose of this project is campus life of the digital library in metaverse. This project should provide users with the following four things. There are online meeting system where you can meet people, a chatbot that informs you of school information and system where you can read books. Finally, there is a system where you can gather and enjoy videos together such as cinema. Each system contains a function of providing user convenience while keeping its purpose. This document will provide detailed requirements and designs for each subsystem.

2.1 Objectives

This chapter provides definitions of the diagrams used for document and applied tools for diagrams.

2.2 Applied Diagrams

2.2.1 UML

UML, short for Unified Modeling Language, is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing object-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects. Using the UML helps project teams communicate, explore potential designs, and validate the architectural design of the software.

2.2.2 Use case Diagram

A use-case model describes a system's functional requirements in terms of use cases. It is a model of the system's intended functionality and its environment. Use cases enable you to relate what you need from a system to how the system delivers on those needs.

2.2.3 Sequence Diagram

The Sequence Diagram models the collaboration of objects based on a time sequence. It shows how the objects interact with others in a particular scenario of a use case. With the advanced visual modeling capability, you can create complex sequence diagram in few clicks. Besides, some modeling tool such as Visual Paradigm can generate sequence diagram from the flow of events which you have defined in the use case description.

2.2.4 Class Diagram

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of objectoriented

systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

2.2.5 Context Diagram

A system context diagram in engineering is a diagram that defines the boundary between the system, or part of a system, and its environment, showing the entities that interact with it. This diagram is a high-level view of a system. It is similar to a block diagram.

2.2.6 Entity Relationship Diagram

An entity relationship diagram, also known as an entity relationship model, is a graphical representation that depicts relationships among people, objects, places, concepts or events within an information technology system.

2.3 Applied Tools

2.3.1 SmartDraw

SmartDraw is a diagram tool used to make flowcharts, organization charts, mind maps, project charts, and other business visuals.

2.3.2 GitMind

GitMind is a online mind map maker for brainstorming, project planning, development, action and other creative tasks.

2.3.3 Diagrams.net

Diagrams.net (previously draw.io) is an online platform and desktop diagram software. It is used to draw flowchart, network diagram, block diagram, electrical circuit diagram to create UML, as an Er diagram tool, for design. Moreover, it supports saving files to cloud services.

2.4 Project Scope

This project will provide virtual digital library to user who cannot enjoy campus life in the COVID-19 pandemic. It will provide not only books, which are the basic library roles, but also various entertainment or meeting environments and provide the ability to obtain information about schools. This environment provides chance to communicate with other students for their campus life.

2.5 References

The user of this SDD may need the following documents for reference:

- Team 7, 2021 Spring, Software Design Document, SKKU.
- https://www.smartdraw.com, n.d. Web.
- https://www.visual-paradigm.com, n.d. Web.

3. System Architecture - Overall

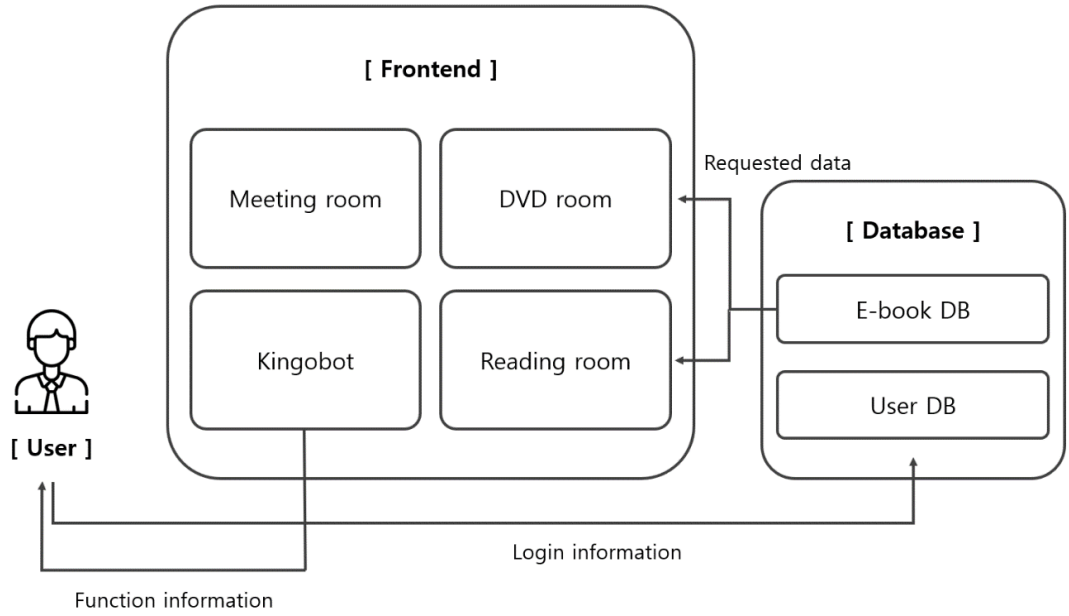
3.1. Objectives

This chapter describes the overall system organization of the frontend and database of the metaverse dido. In addition, the system structure is summarized and expressed in three different diagrams: context diagram, sequence diagram, and use case diagram.

3.2. System Organization

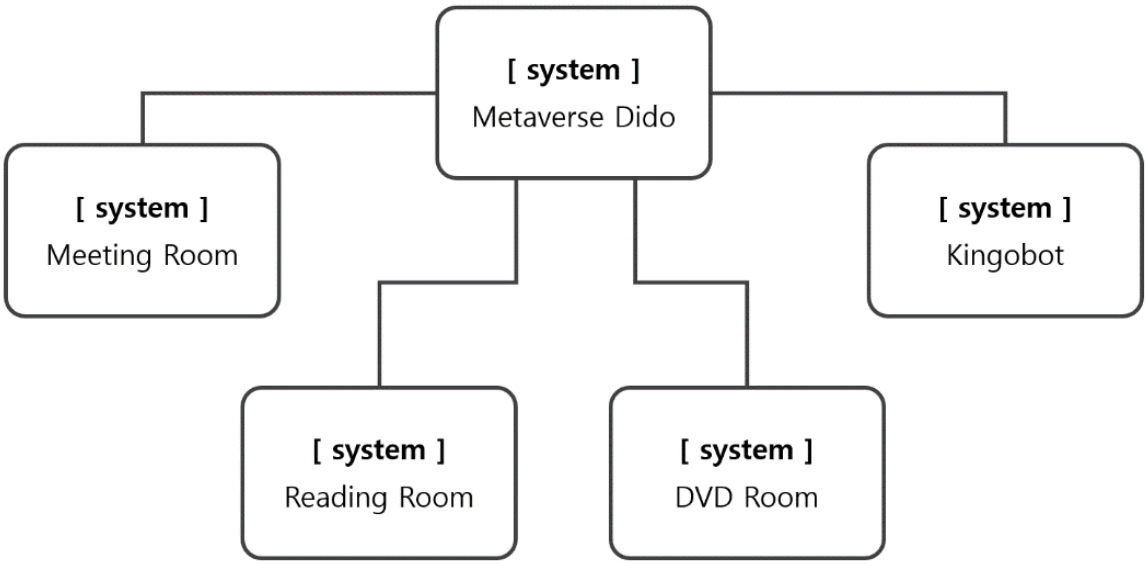
The metaverse dido system is divided into two parts: frontend and database to implement a smart digital library. The frontend part uses Unity's VRCSDK3 package to implement online meeting rooms, reading rooms, DVD rooms, and a chatbot called is a chatbot that acts as a guide for metaverse dido functions. In the database part, various e-books, papers, and video materials are stored using the SKKU Academic Information Center database as well as user information.

Figure 1 Overall System Organization



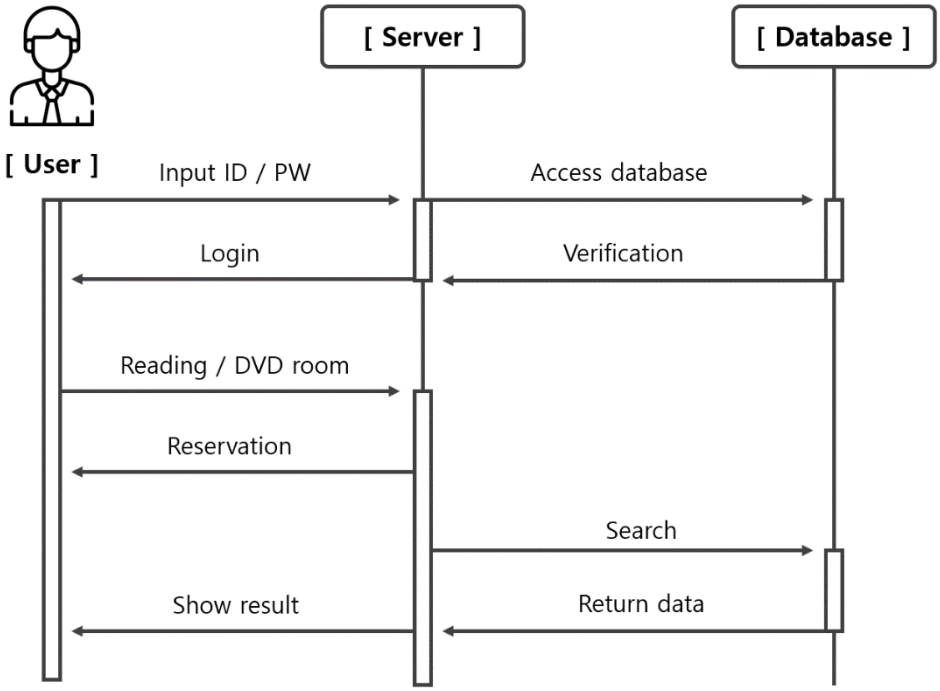
3.2.1 Context Diagram

Figure 2 Overall Context Diagram



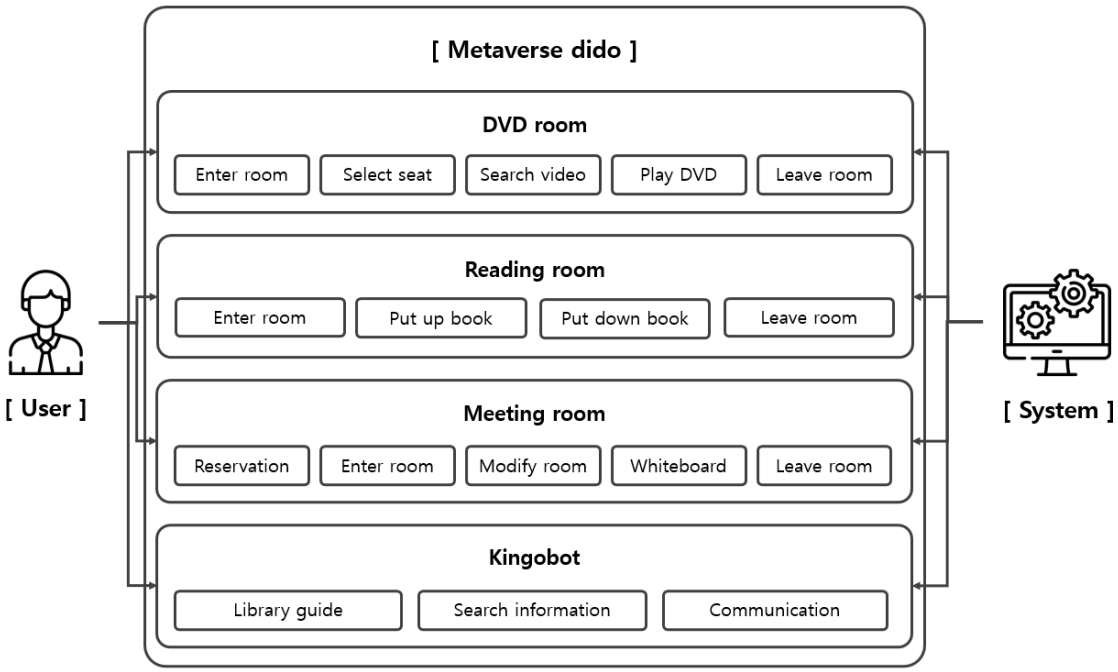
3.2.2 Sequence Diagram

Figure 3 Overall Sequence Diagram



3.2.3. Use Case Diagram

Figure 4 Overall Use Case Diagram



4. System Architecture - Frontend

4.1. Objectives

This chapter provides frontend of system. VRCHAT platform and UNITY are main UX/UI tools of system. Unity provide functions for designing worlds for VRCHAT platform.

4.2. Subcomponents

4.2.1. DVD Room

The DVD room class is a class that handles one of the location information in Metaverse world. When a user enters, the user can either select a DVD or find other audiovisual materials.

4.2.1.1. Attributes

These are the attributes that the DVD room class has.

- **total_users** : the total number of users who enter the DVD room

These are the attributes that the user of the DVD room has.

- **user_id** : ID of the users in the DVD room
- **seat_id** : ID of the seats in the DVD that users can read
- **DVD_id** : ID of the DVDs that users can select
- **DVD_genre** : category of the DVDs
- **Video_url** : url of a selected video(audiovisual material) by a user

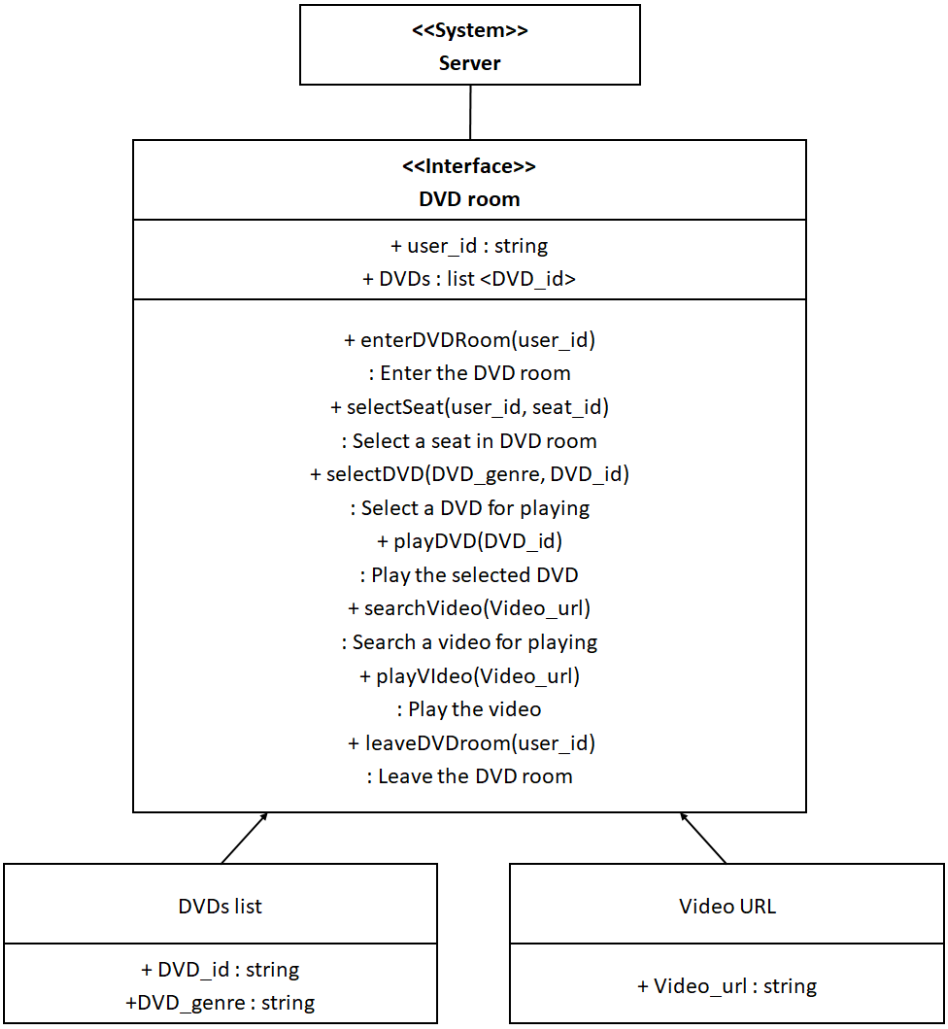
4.2.1.2 Methods

These are the methods that the DVD room class has.

- enterDVDRoom()
- selectSeat()
- selectDVD()
- playDVD()
- searchVideo()
- playVideo()
- leaveDVDRoom()

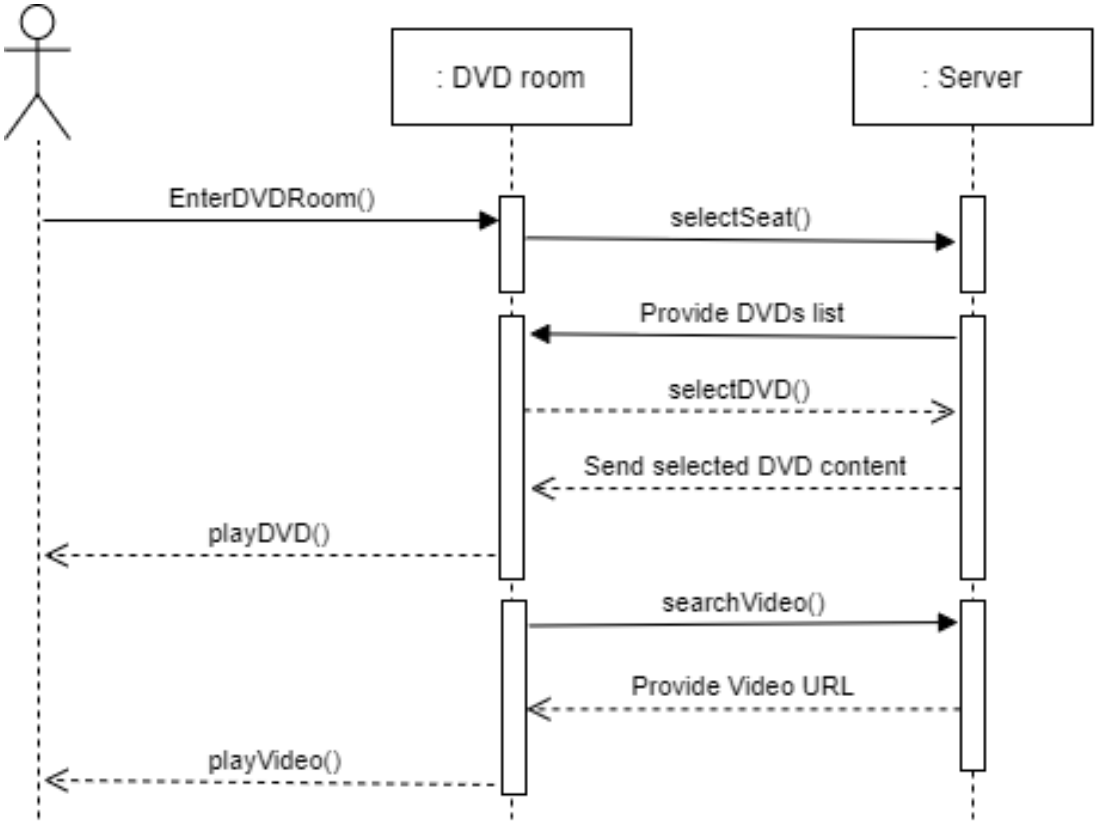
4.2.1.3 Class Diagram

Figure 5 Class diagram – DVD Room



4.2.1.4 Sequence Diagram

Figure 6 Sequence diagram – DVD Room



4.2.2. Reading room

The reading room class defines and manipulates virtual reading room of "Metaverse Dido".

4.2.2.1. Attributes

These are the attributes that the reading room class has.

- **user_id** : ID of the users in the reading room.
- **book_id** : ID of the books in the reading room that user can read.
- **book_text** : text of the each book.

- **book_location** : location of the each book.
- **shelf_id** : ID of the shelf that store books
- **shelf_location** : location of the shelf
- **table_id** : ID of the table that user can put books
- **table_location** : location of the table
- **Seat_id** : ID of the seat that user can sit on, user should reserve before use it
- **Seat_location** : location of the seat
- **Seat_reserveInfo** : reservation information about each seat

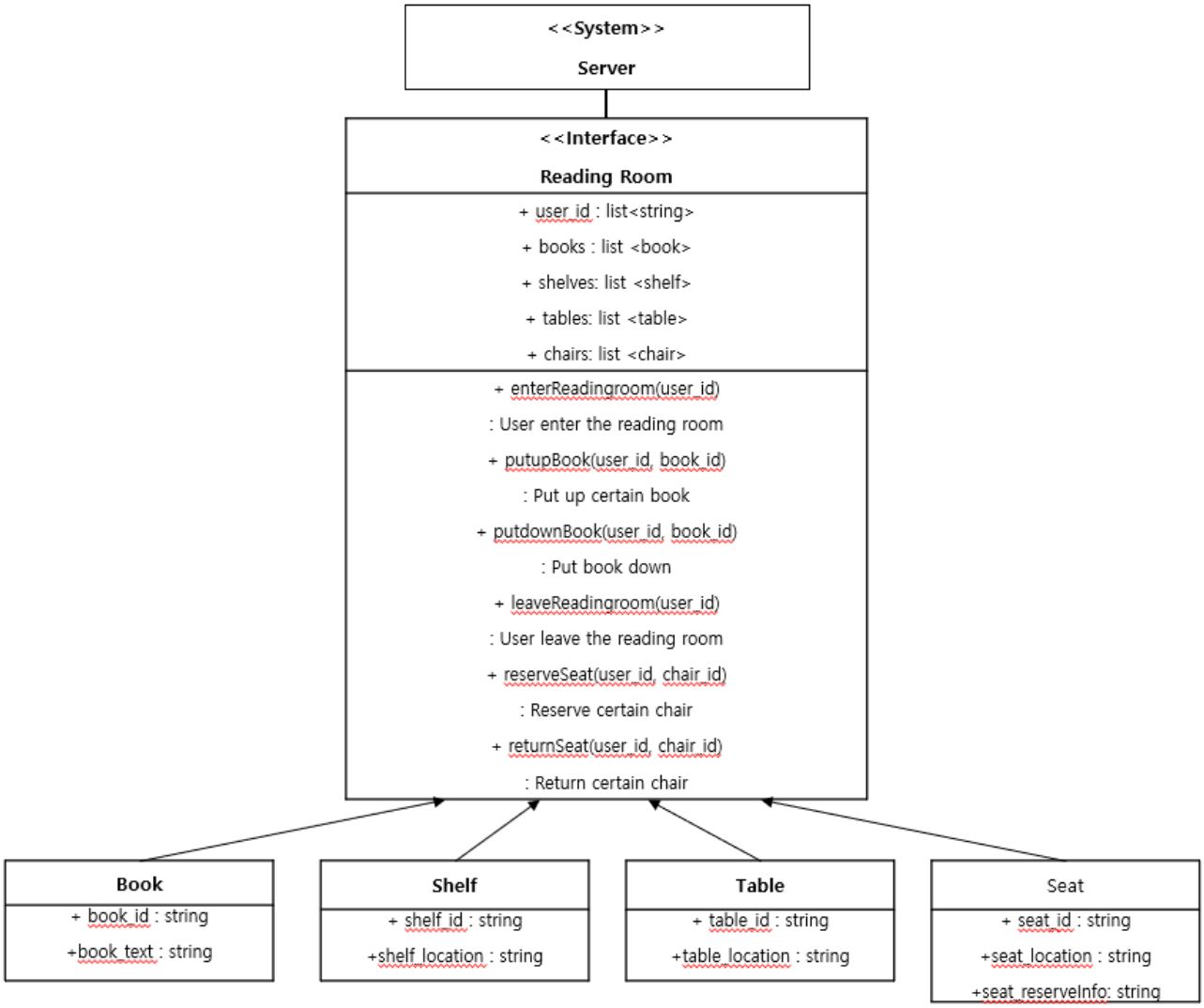
4.2.2.2 Methods

These are the methods that the profile class has.

- enterReadingroom()
- putupBook()
- putdownBook()
- reserveSeat()
- returnSeat()
- leaveReadingroom()

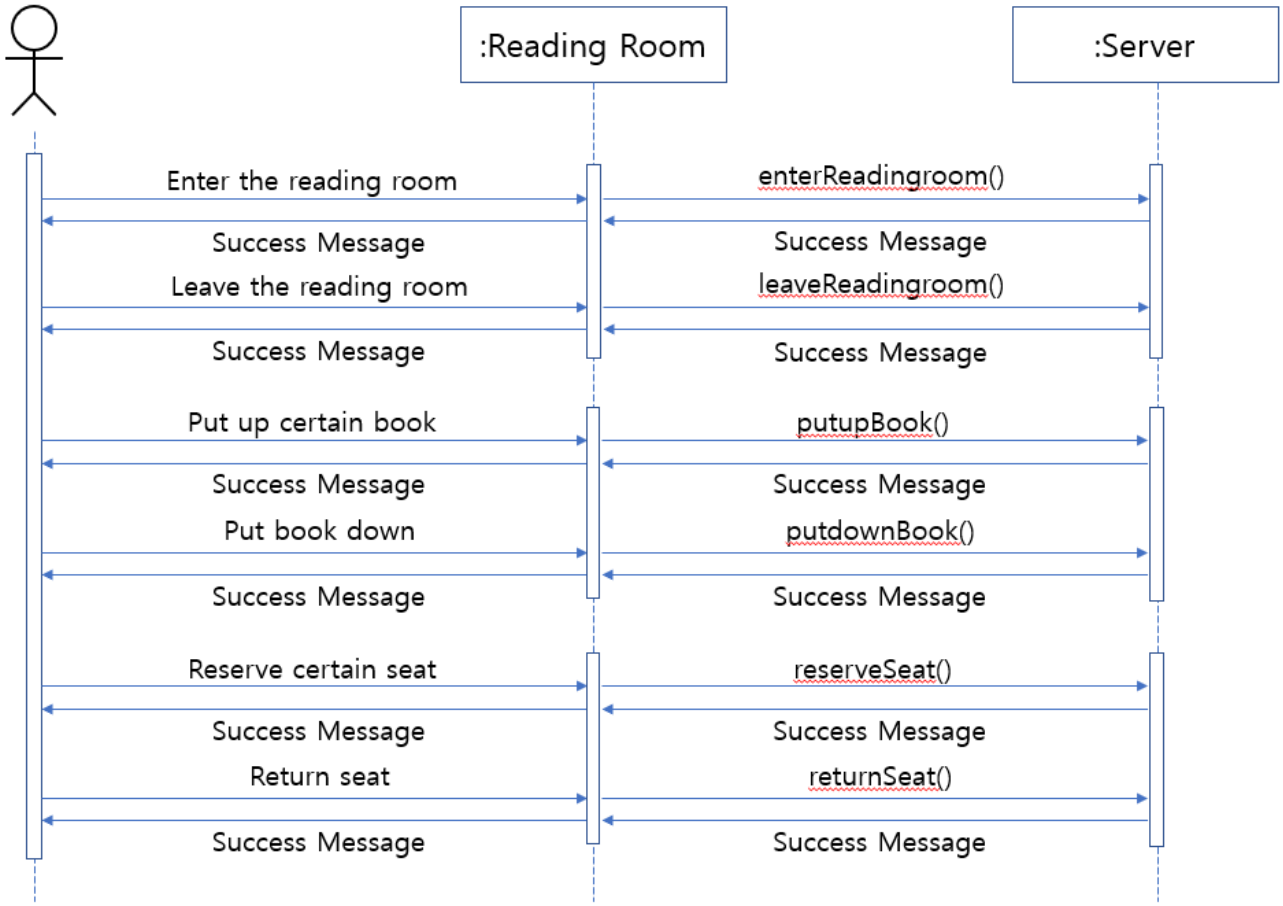
4.2.2.3 Class Diagram

Figure 7 Class diagram – Reading Room



4.2.2.4 Sequence Diagram

Figure 8 Sequence diagram – Reading Room



4.2.3. Meeting Room

4.2.3.1. Attributes

These are the attributes that the meeting room object has.

- **room_id** : id of the room.
- **number** : number for the meeting room.
- **room_status** : status of the meeting room.
- **room_password** : password of the meeting room.

- **user_id** : id of the user.
- **reservation_time** : time of the reservation.
- **board_id** : id of the white board.
- **board_content** : content of the white board.
- **board_location** : location of the white board.

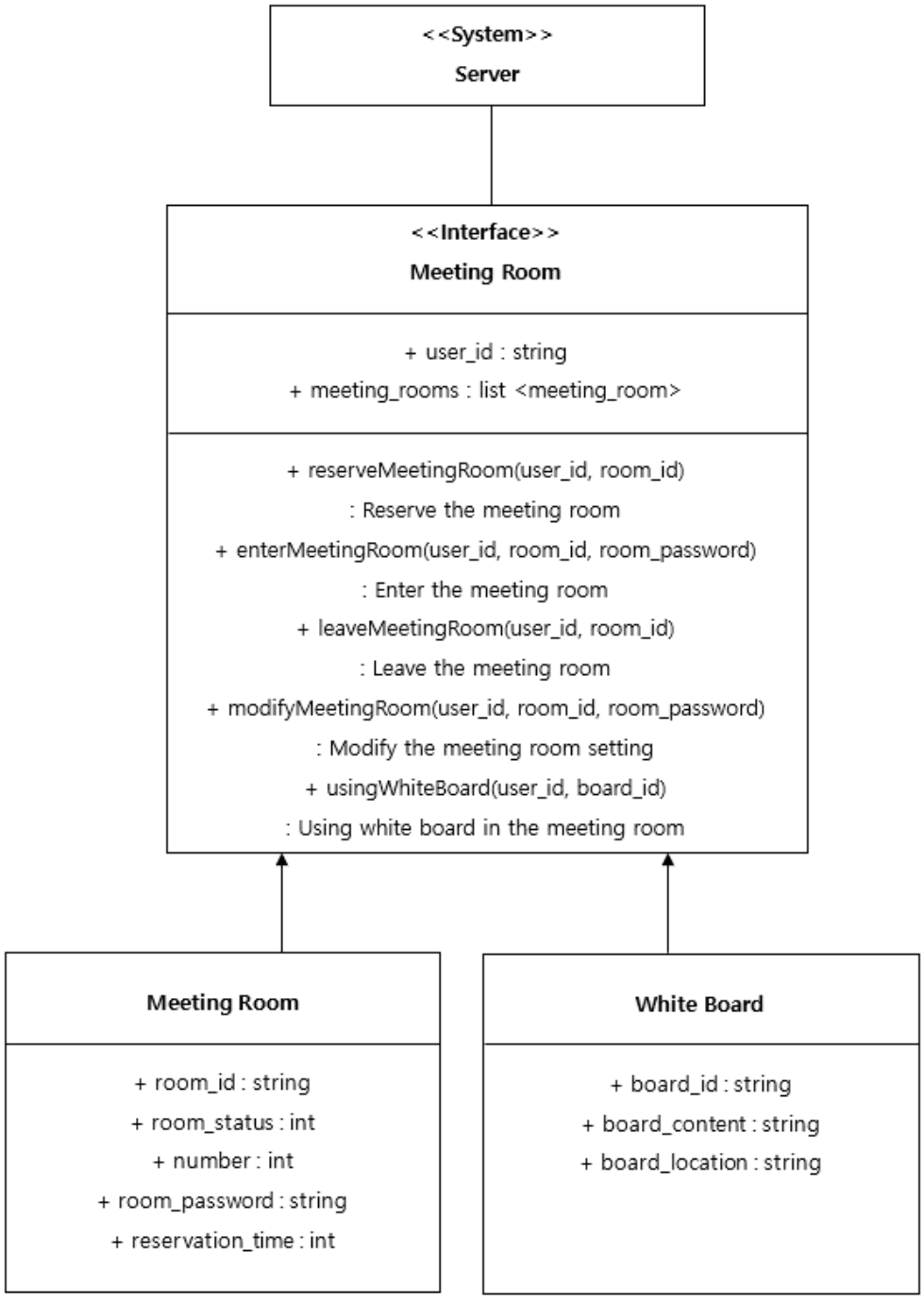
4.2.3.2 Methods

These are the methods that the meeting room class has.

- reserveMeetingRoom()
- enterMeetingRoom()
- leaveMeetingRoom()
- modifyMeetingRoom()
- usingWhiteBoard()

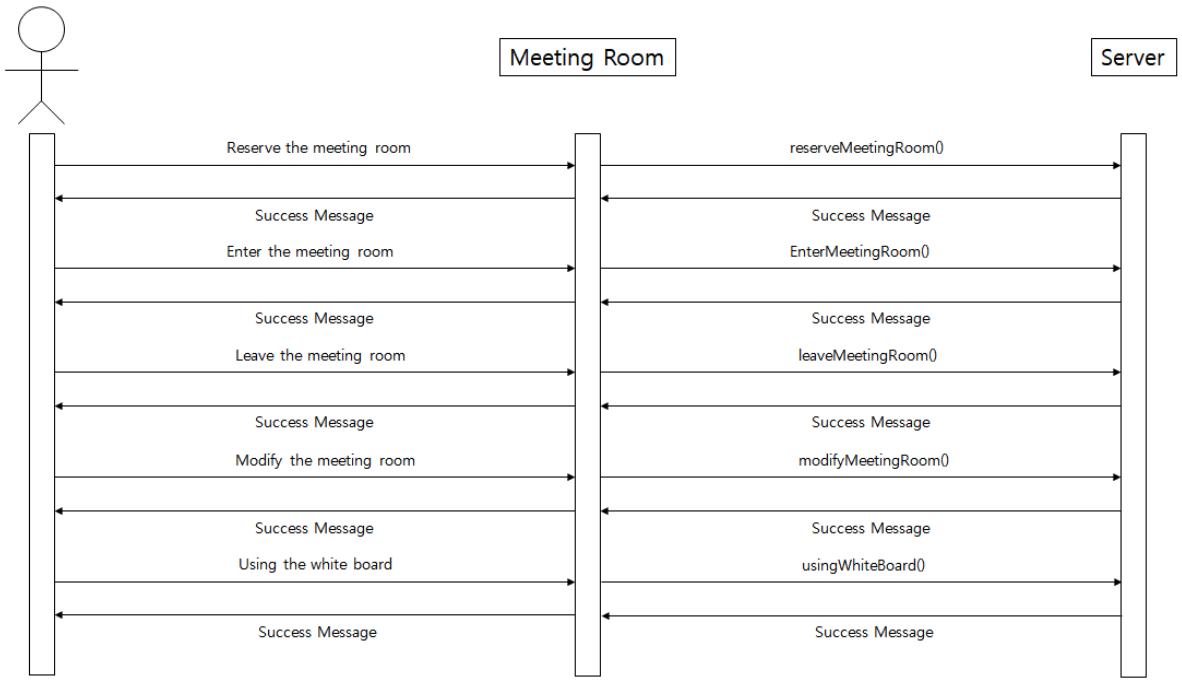
4.2.3.3 Class Diagram

Figure 9 Class diagram - Meeting Room



4.2.3.4 Sequence Diagram

Figure 10 Sequence diagram - Meeting Room



4.2.4 ChatterBot

The chatterbot is an interface that is created in the metaverse to interact with the user and provide useful information related to education, school life and academic-related inquiry

4.2.4.1. Attributes

- user_id: id of the user
- student_id: number of student id
- location: location of the user
- building_number : number of building

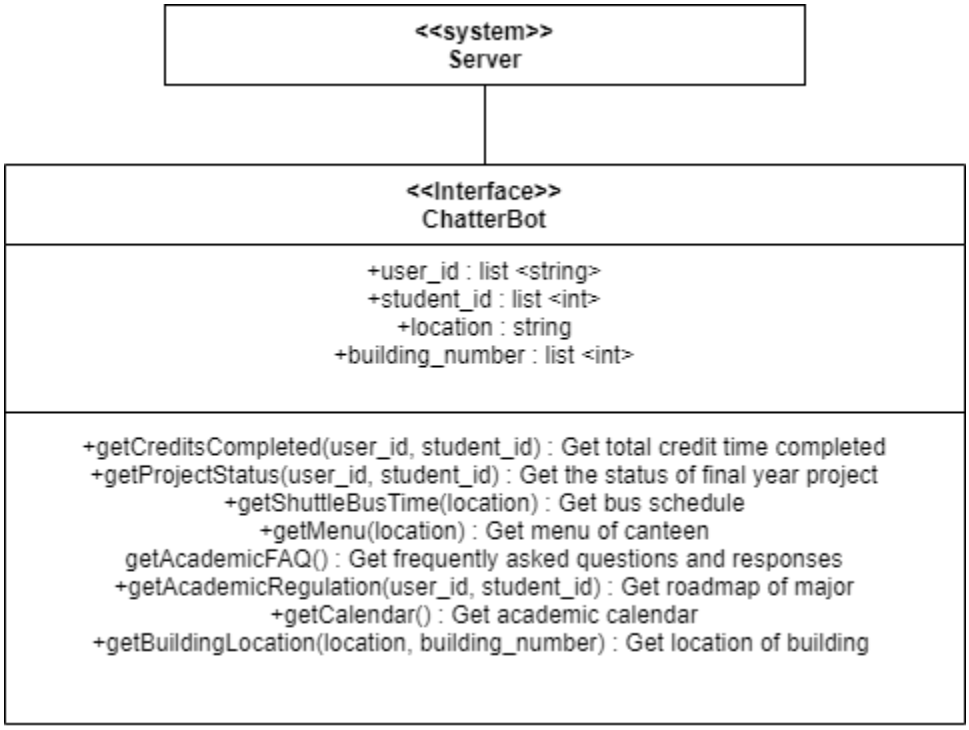
4.2.4.2 Methods

- getCreditsCompleted()
- getProjectStatus()

- getShuttleBusTime()
- getMenu()
- getAcademicFAQ()
- getAcademicRegulation
- getCalendar()
- getBuildingLocation()

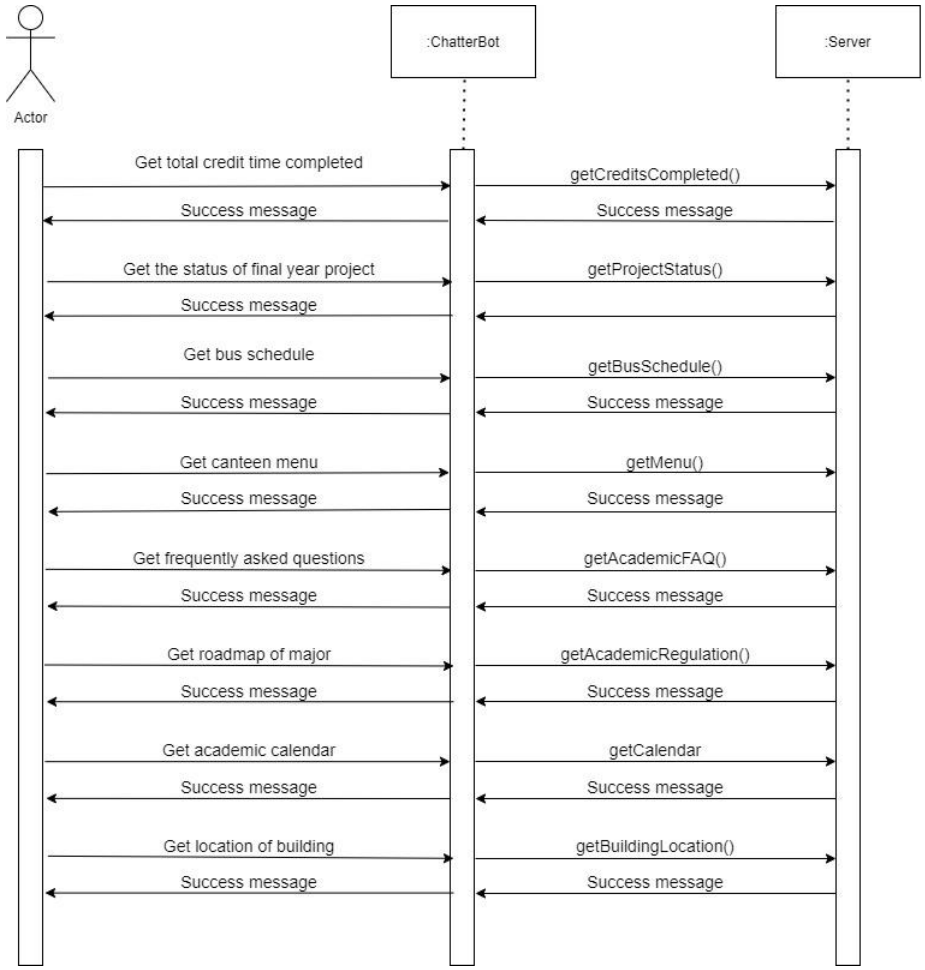
4.2.4.3 Class Diagram

Figure 11 Class Diagram - ChatterBot



4.2.4.4 Sequence Diagram

Figure 12 Sequence Diagram – ChatterBot



5. System Architecture - Backend

5.1. Objectives

This chapter describes the structure of the back-end system include search systems, reservation systems and DB. Each system interacts with user and DB to perform necessary functions on metaverse Library.

5.2. Overall Architecture

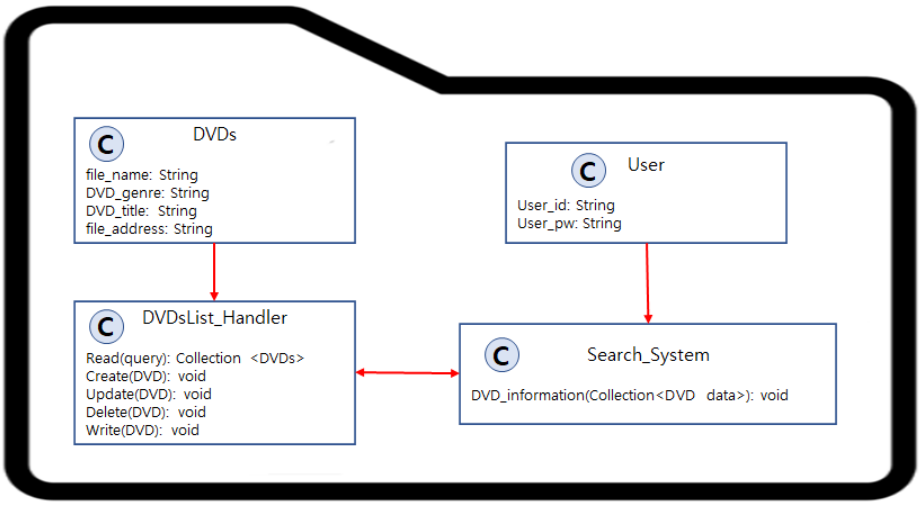
The overall architecture of the system is as follow: there are two kinds of systems: the search system and the reservation system. The DVD room and reading room will support users with a search system, and the meeting room and reading room will support the user with a reservation system. Each systems receives requests from user through keywords to search or reservation requests internally. Then, the system accesses on the List of DVDs or DB of E-books using handler to find appropriate material. Otherwise, the system accesses to list of room information or list of seats information to reserve selected place. After performing the necessary tasks in each system, the results are transmitted to the user.

5.3. Subcomponents

5.3.1. Search system of DVD room

5.3.1.1. Class Diagram

Figure 13 Class diagram – Search system of DVD room

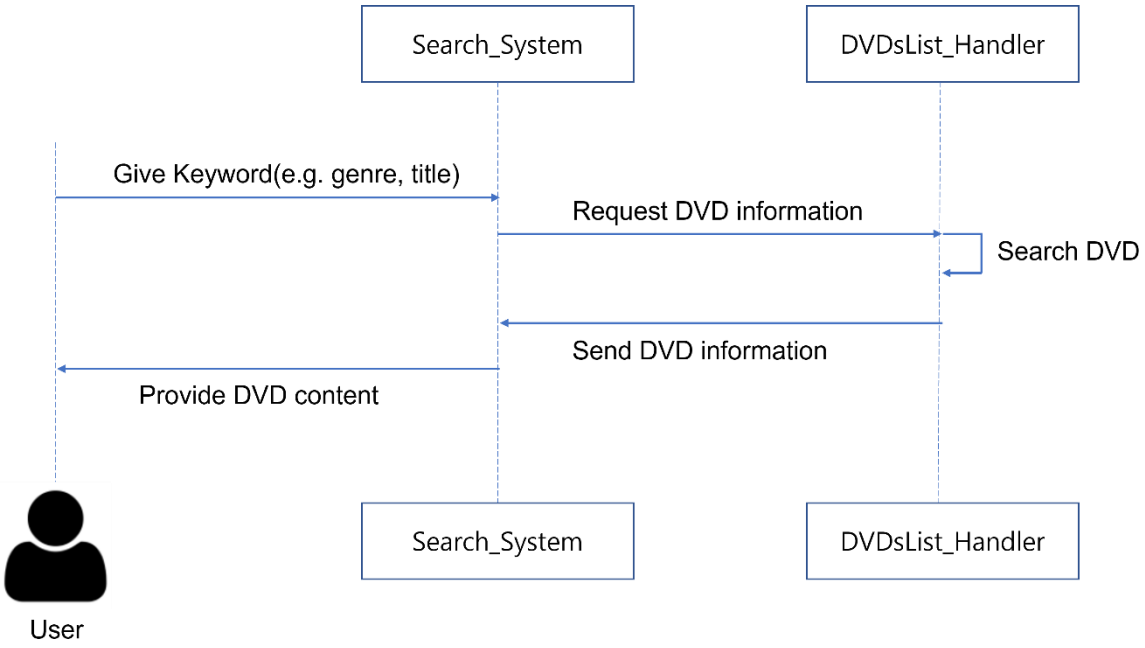


- Class description

- ✓ **Search system of DVD room:** This is the interface for search system of DVD room. A user inputs keywords of a DVD which includes genre and title information. The DVDsList contains every information and content for DVDs. After user requests about DVD information, search system starts function for searching the genre and title of DVD by accessing to DVDsList. When the searching is complete, search system responds to the user with searched DVD content.

5.3.1.2. Sequence Diagram

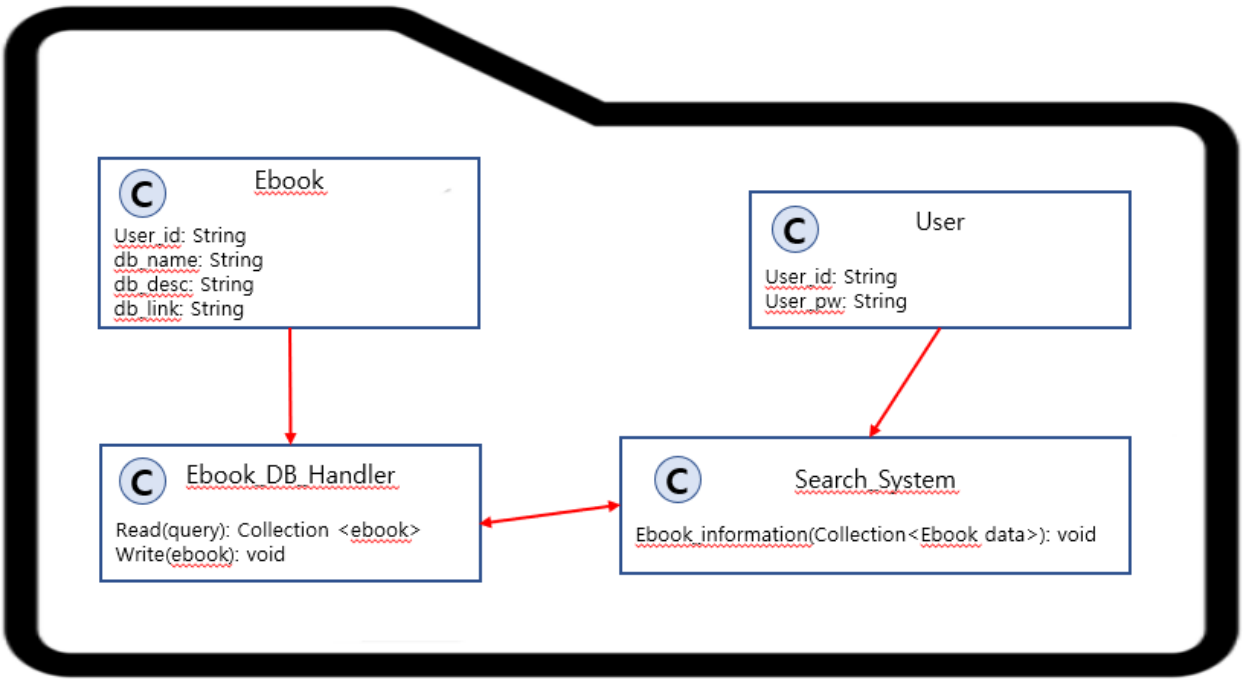
Figure 14 Sequence diagram- Search system of DVD room



5.3.2. Search system of reading room

5.3.2.1. Class Diagram

Figure 15 Class diagram – Search system of reading room

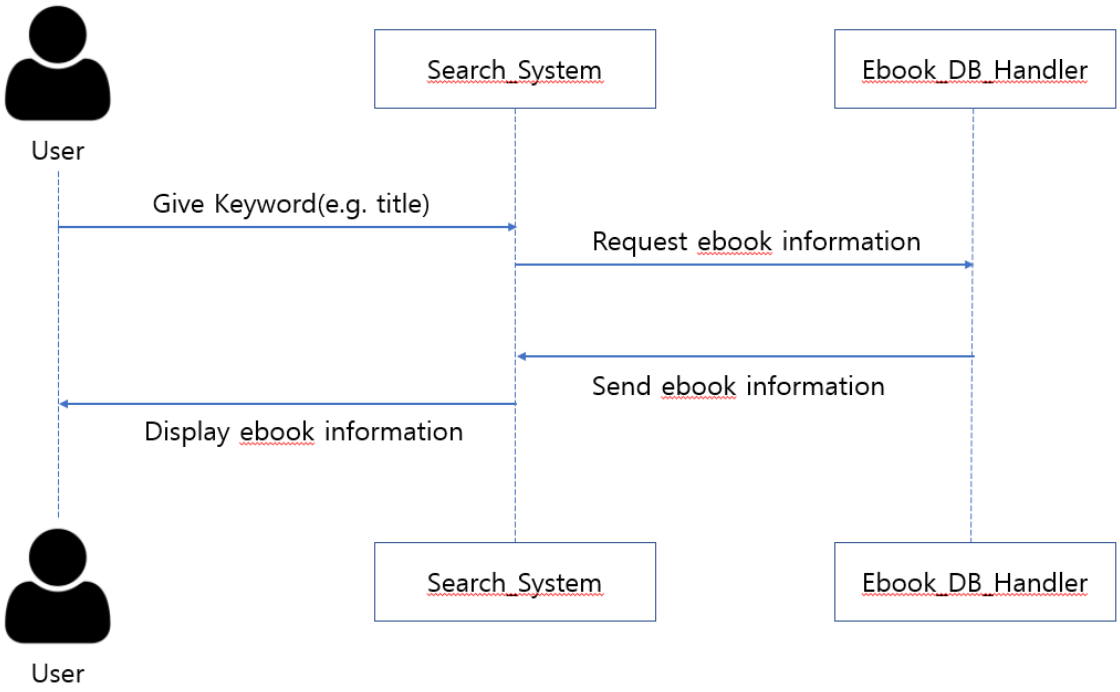


- Class description

  ✓ **Search system of Reading room:** This is the interface for search system of reading room. User input keyword to search such as title of book, and then search system receives it and request search results to e-book DB handler. E-book DB handler makes search results and send it to search system.

5.3.2.2. Sequence Diagram

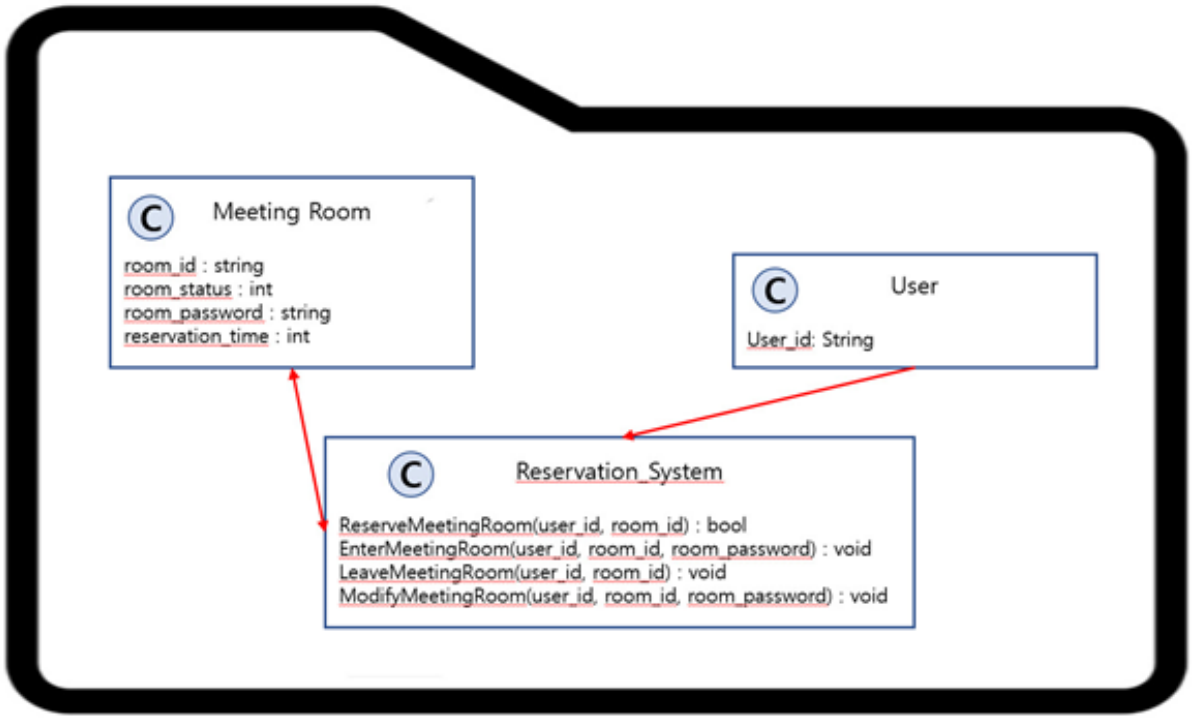
Figure 16 Sequence diagram- Search system of reading room



5.3.3. Reservation system of meeting room

5.3.3.1. Class Diagram

Figure 17 Class diagram - Reservation System of Meeting Room

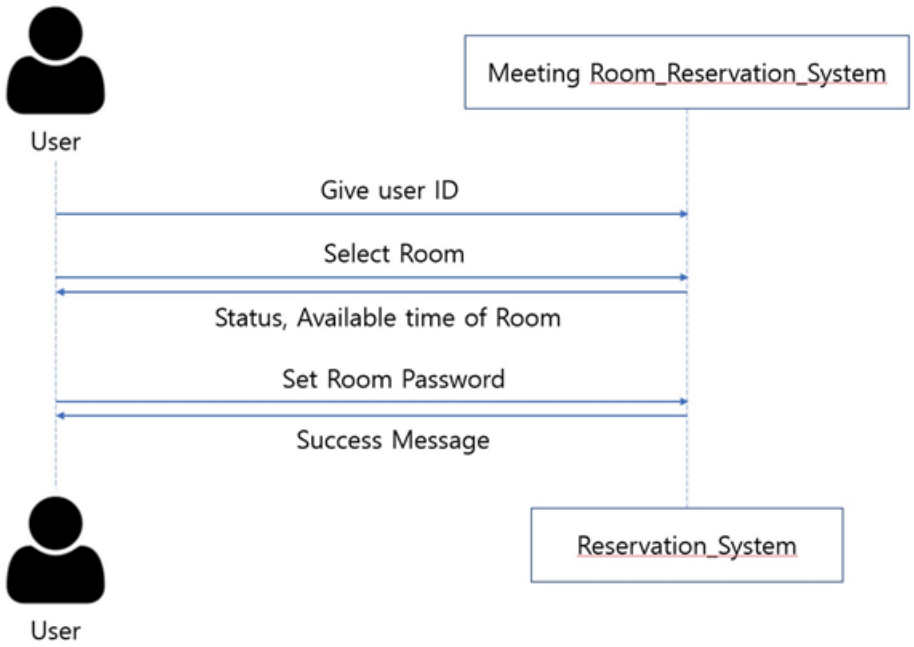


- Class description

  ✓ **Reservation system of Meeting Room:** This is the interface for reservation system of meeting room. When user request room id that he searches, the system request status and available time of that room. User set room password when he reserves the room. Also user can enter, modify, and leave each room.

5.3.3.2. Sequence Diagram

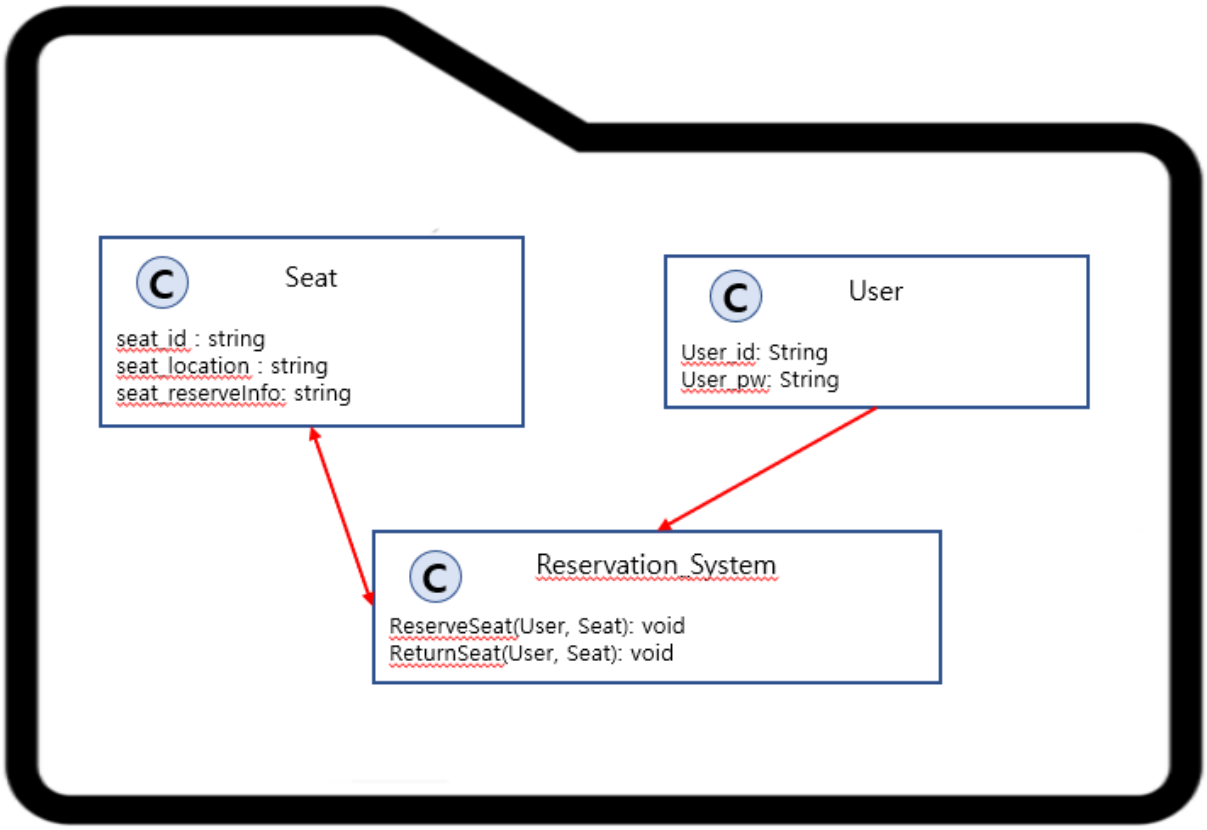
Figure 18 Sequence diagram - **Reservation** System **of Meeting Room**



5.3.4. Reservation system of reading room

5.3.4.1. Class Diagram

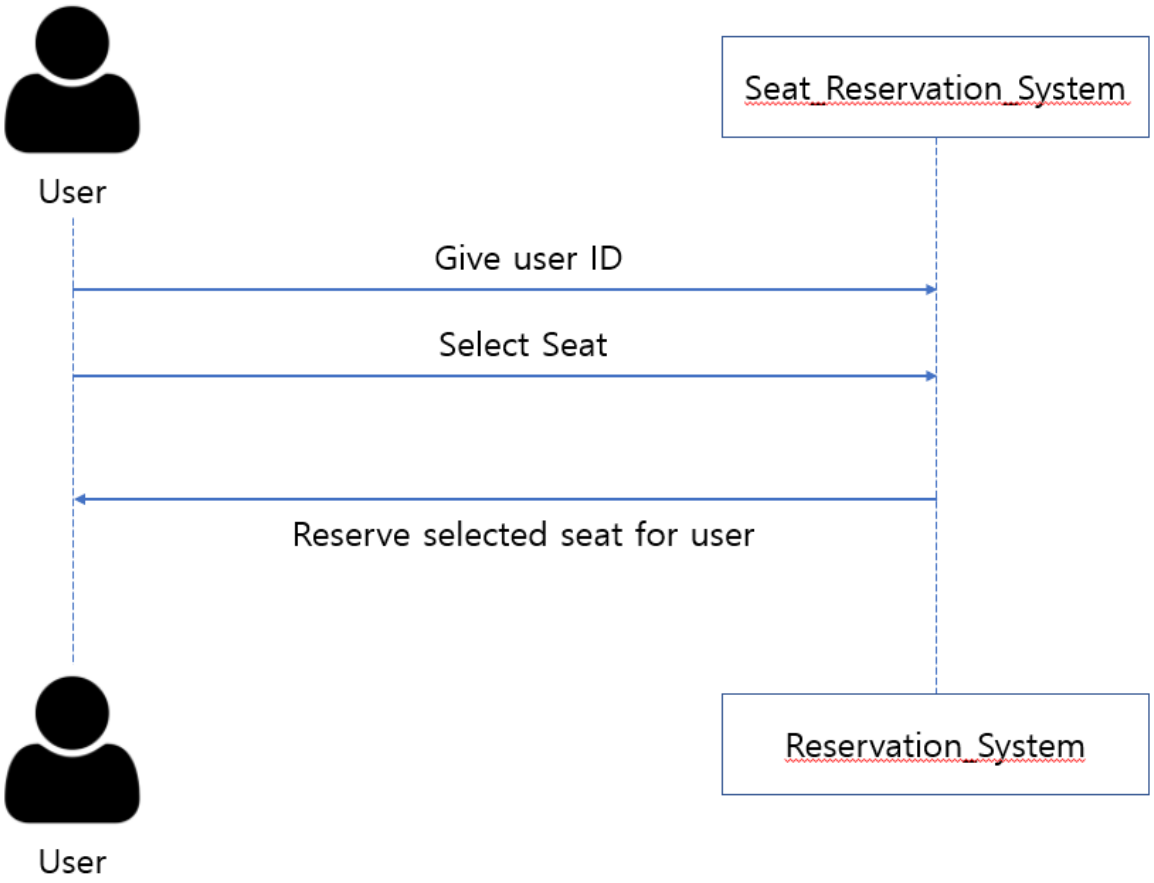
Figure 19 Class diagram – Reservation system of reading room



- Class description
  - ✓ **Reservation system of Reading room:** This is the interface for reservation system of reading room. User select seat which is available, make request to reserve, and then reservation system receives it. Reservation system check data of seat classes, and if user can reserve seat, reflect changes to data. Send result to requested user.

5.3.4.2. Sequence Diagram

Figure 20 Sequence diagram- Reservation system of reading room



6. Protocol Design

6.1. Objectives

This chapter show the protocols design of each subsystem.

6.2. JSON

JavaScript Object Notation (JSON) is a standard text-based format for representing structured data based on JavaScript object syntax. It is commonly used for transmitting data in web applications

6.4. DVD Room

6.4.1. Enter to DVD Room

- Request

Table 1 Enter to DVD Room request

Attribute	Detail	
URI	/user/:id/profile	
Method	GET	
Parameter	User	User information
Header	Authorization	User authentication

- Response

Table 2 Enter to DVD Room response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.4.2. Select seat

- Request

Table 3 Select DVD room seat request

Attribute	Detail	
URI	/DVD_room/seat	
Method	GET	
Parameter	User	Basic User Information
	Seat	Identification key of seat
Header	Authorization	User authentication

- Response

Table 4 Select DVD room seat response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.4.3. Select DVD

- Request

Table 5 Select DVD request

Attribute	Detail
URI	/DVD_room/DVD

Method	GET	
Parameter	Genre	Category of the DVDs
	DVD	Identification key of seat
Header	Authorization	User authentication

- Response

Table 6 Select DVD response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.4.4. Search Video

- Request

Table 7 Search Video request

Attribute	Detail	
URI	/DVD_room/Video	
Method	GET	
Parameter	URL	Web address of the Video
Header	Authorization	User authentication

- Response

Table 8 Search Video response

Attribute	Detail

Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.5. Meeting Room

6.5.1. Reservation

- Request

Table 9 Reserve a meeting room request

Attribute	Detail	
URI	/meeting_room/room_status	
Method	GET	
Parameter	User	Basic User Information
	Time	Desired Time
	Password	Meeting Room Password
Header	Authorization	User authentication

- Response

Table 10 Reserve a meeting room response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 403 (Forbidden)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access

	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.5.2. Enter to Meeting Room

- Request

Table 11 Enter to meeting room request

Attribute	Detail	
URI	/user/:id/profile	
Method	GET	
Parameter	Password	Meeting Room Password
Header	Authorization	User authentication

- Response

Table 12 Enter to meeting room response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.5.3. Using the White Board

Request

Table 13 Using the white board request

Attribute	Detail

URI	/user/:id/profile	
Method	GET	
Parameter	Number	Board Number
Header	Authorization	User authentication

- Response

Table 14 Using the white board response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.6. Reading Room

6.6.1. Enter to Reading room

- Request

Table 15 Enter to reading room request

Attribute	Detail	
URI	/user/:id/profile	
Method	GET	
Parameter	User	User information
Header	Authorization	User authentication

- Response

Table 16 Enter to reading room response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.6.2. Put up certain book

- Request

Table 17 Put up certain book request

Attribute	Detail	
URI	/user/:id/profile	
Method	GET	
Parameter	User	User’s location
	Book	ID of book
Header	Authorization	User authentication

- Response

Table 18 Put up certain book response

Attribute	Detail
Success Code	HTTP 200 OK
Failure Code	HTTP 400 (Bad request, overlap)

	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.6.3. Reserve seat

- Request

Table 19 Reserve seat request

Attribute	Detail	
URI	/user/:id/profile	
Method	GET	
Parameter	User	User information
	Seat	ID of seat
Header	Authorization	User authentication

- Response

Table 20 Reserve seat response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.7. TCP

Transmission Control Protocol or also known as TCP, is a network layer protocol that provides a logical communication between application processes running on different hosts. The protocol offers a reliable and in-order delivery of frames during data transmissions over networks. By applying socket programming with TCP, a client-server connection can be established, creating an uplink for a server to talk with multiple clients. In this context, this project will be using socket programming to implement a simple chatbot as a medium to provide information on a variety of topics related to education, school life, and other academic-related inquiries.

6.8. ChatterBot

6.8.1 Establish Connection with ChatterBot

- Request

Table 21 Get chatterbot request

Attribute	Detail	
URI	/chatterbot	
Method	GET	
Parameter	User	Basic User Information
	Port	Port Number of Server
Header	Authorization	User authentication

- Response

Table 22 Get chatterbot response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access

	List Box	GUI frame for messages and responses
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.8.2 Graduation Requirement

- Request

Table 23 Get graduation requirement request

Attribute	Detail	
URI	/school_life/bus_schedule	
Method	GET	
Parameter	User	Basic User Information
	Credits	User’s Completed Credits
	Project	User’s Final Year Project Status
Header	Authorization	User authentication

- Response

Table 24 Get graduation requirement response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Graduation Status	Views the required criteria to graduate
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.8.3 Bus Schedule

- Request

Table 25 Get shuttle bus schedule request

Attribute	Detail	
URI	/school_life/bus_schedule	
Method	GET	
Parameter	User	Basic User Information
	Location	User's location
Header	Authorization	User authentication

- Response

Table 26 Get shuttle bus schedule response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Time	Returns schedule of SKKU shuttle bus
	Message	Message: "Access success"
Failure response body	Message	Message: "Access fail"

6.8.4 Canteen Menu

- Request

Table 27 Get canteen menu request

Attribute	Detail
URI	/school_life/canteen_menu

Method	GET	
Parameter	User	Basic User Information
	Location	User's location
Header	Authorization	User authentication

- Response

Table 28 Get canteen menu response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Menu	Returns canteen menu of the day
	Message	Message: "Access success"
Failure response body	Message	Message: "Access fail"

6.8.5 Academic Information

- Request

Table 29 Get academic information request

Attribute	Detail	
URI	/academic_faq	
Method	GET	
Parameter	User	Basic User Information
Header	Authorization	User authentication

- Response

Table 30 Get academic information response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Grade	Returns current grade
	Scholarship	Returns scholarship status
	Leave of Absence	Shows status of leave of absence
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.8.6 GLS

- Request

Table 31 Get GLS request

Attribute	Detail	
URI	/gls	
Method	GET	
Parameter	User	Basic User Information
Header	Authorization	User authentication

- Response

Table 32 Get GLS response

Attribute	Detail
Success Code	HTTP 200 OK
Failure Code	HTTP 400 (Bad request, overlap)
	HTTP 404 (Not found)

Success response body	Access Token	Token for access
	GLS	Returns GLS page
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

6.8.7 Campus Information

- Request

Table 33 Get campus info request

Attribute	Detail	
URI	/campus_info	
Method	GET	
Parameter	User	Basic User Information
	Location	User’s Location
Header	Authorization	User authentication

- Response

Table 34 Get campus info response

Attribute	Detail	
Success Code	HTTP 200 OK	
Failure Code	HTTP 400 (Bad request, overlap)	
	HTTP 404 (Not found)	
Success response body	Access Token	Token for access
	Classroom Location	Returns location of empty classroom
	Building Location	Returns location of building
	Map	View map
	Message	Message: “Access success”
Failure response body	Message	Message: “Access fail”

7. Database Design

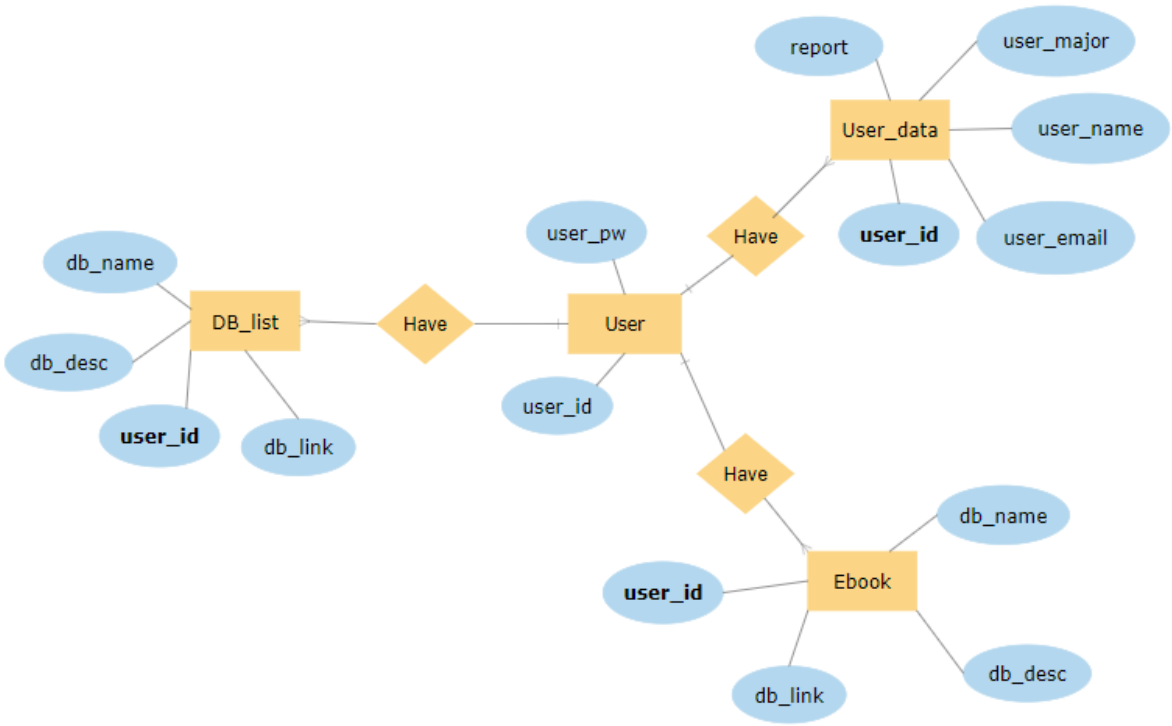
7.1. Objectives

This part describes the database and how the system data structures will be represented in the system. Starting with creating an ER-diagram (Entity Relationship diagram), it will then be break down into entities and how it relates to different data sets. Then, the Relational Schema and SQL DDL (Data Description Language) specifications will be described and identified.

7.2. ER Diagram

The system consists of four main entities: User, User_data, Ebook, and DB_list. The rectangular shape represents the entities while the diamond shape represents the relations. The primary key data are bolded in the diagram. When an entity has multiple relationships with another entity, trident (three line) is used to indicate it. When an entity has just one relationship with another entity, the cross (two line) is used to indicate it.

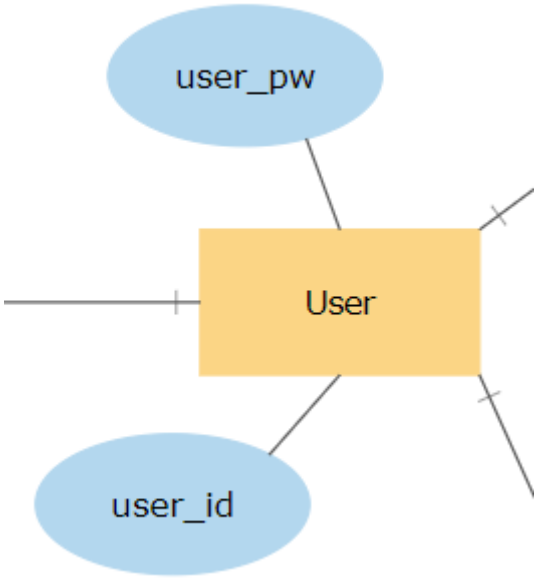
Figure 21 ER-Diagram



7.2.1 Entities

7.2.1.1. User

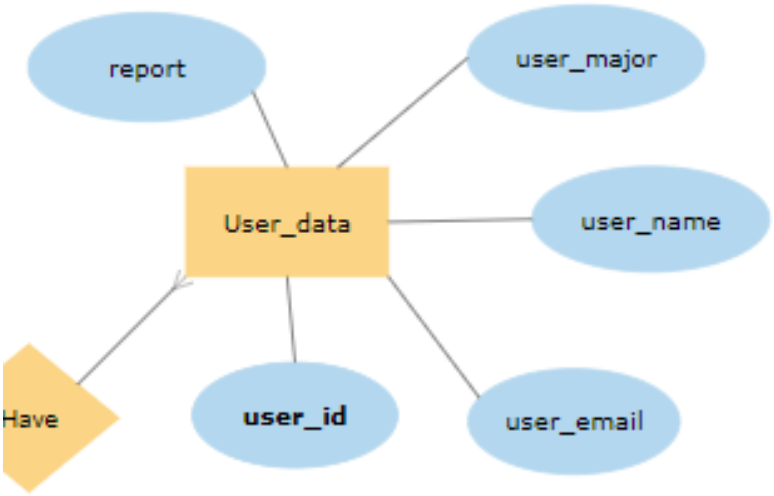
Figure 22 ER diagram, Entity, User



User entity represents the basic user login information. These data is essential to access the database that is available only for Sungkyunkwan University students. This entity consists of 2 basic login information which are user_id and user_pw. The login ID of student will be stored in user_pw and the password will be stored in user_pw.

7.2.1.2. User_data

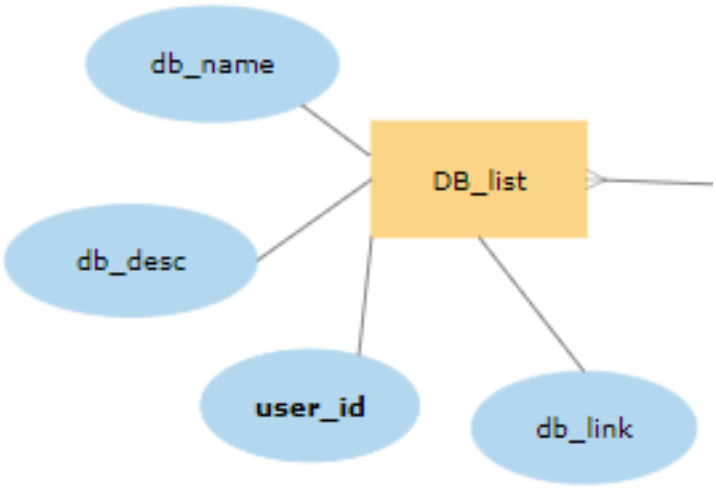
Figure 23 ER diagram, Entity, User_data



User_data represents the basic information of the students. It consists of user_id, user_email, user_name, user_major and report. The user_id is the primary key to access the User_data. If there is any issue that the user want to report, those data will be stored in report and will be directly linked to each user.

7.2.1.3. DB_list

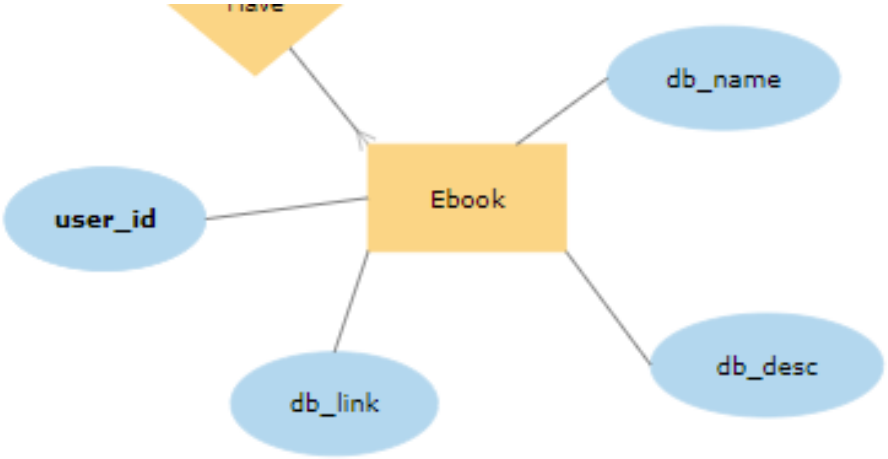
Figure 24 ER diagram, Entity, DB_list



DB_list represents all the database available to be access by the users. Among the attributes are user_id, db_name, db_desc, and db_link. Similar to Used_data, the user_id will be the primary key in order to access the databases available. db_desc will be a short description or summary about the database and db_link will be the redirection link through SKKU library proxy to access the database.

7.2.1.4. Ebook

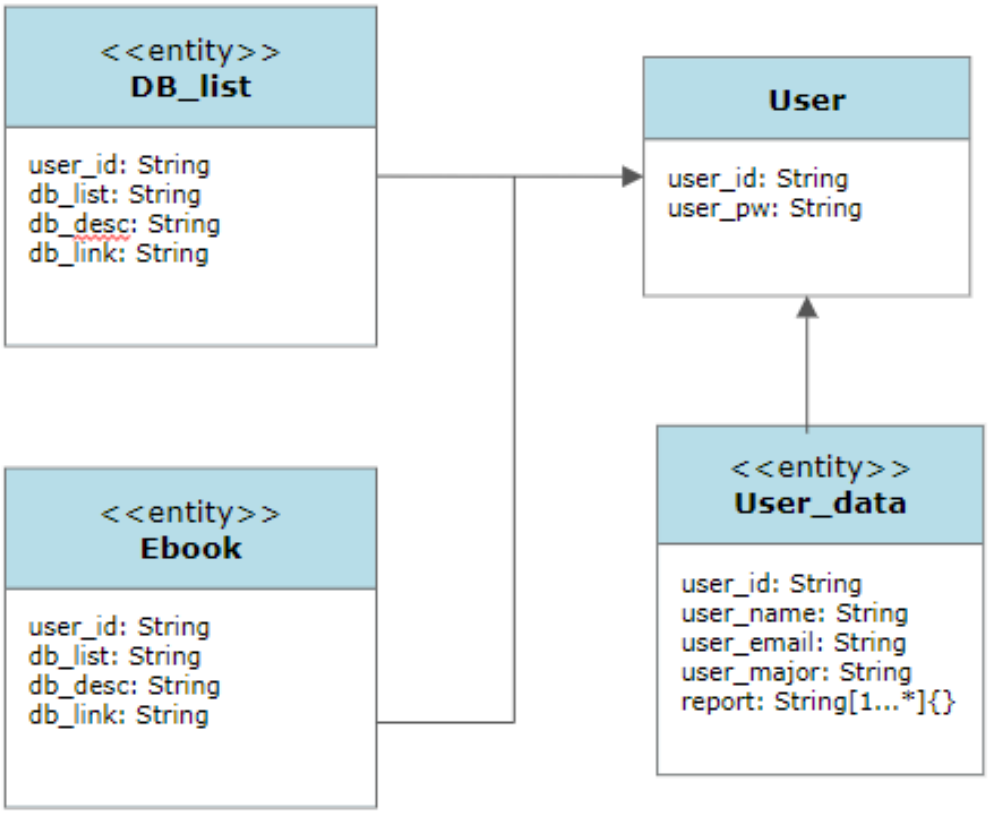
Figure 25 ER diagram, Entity, Ebook



As what is done in DB_list, all the attributes and data of the Ebook database are stores in Ebook. Since the Ebook has a slightly different information as DB_list and for the purpose of categorization, it needs to be stored in a different entity.

7.3. Relational Schema

Figure 26 Relational Schema



7.4. SQL DDL

7.4.1 User

Figure 27 User

```
CREATE TABLE User
(
user_id VARCHAR NOT NULL
user_pw VARCHAR NOT NULL
);
```

7.4.2 User_data

Figure 28 User_data

```
CREATE TABLE User_data
(
user_id VARCHAR NOT NULL
user_id VARCHAR NOT NULL
user_email VARCHAR NOT NULL
user_major VARCHAR NOT NULL
report ARRAY[]
);
```

7.4.3 DB_list

Figure 29 Table DB_list

```
CREATE TABLE DB_list
(
user_id VARCHAR NOT NULL
db_list VARCHAR NOT NULL
db_desc VARCHAR
db_desc VARCHAR NOT NULL
);
```

7.4.4 Ebook

Figure 30 Table Ebook

```
CREATE TABLE Ebook
(
user_id VARCHAR NOT NULL
db_list VARCHAR NOT NULL
db_desc VARCHAR
db_desc VARCHAR NOT NULL
);
```

8. Testing Plan

8.1. Objectives

In this chapter describes plans for VRCHAT World. Development testing, release testing and user testing are included. First tests need to fine error in function or design of application. After that, judge whether world is appropriate for upload and release.

8.2. Testing Policy

8.2.1. Development Testing

Development testing is a software development process that involves synchronized application of a broad spectrum of defect prevention and detection strategies in order to reduce software development risks, time, and costs.

Development testing might include static code analysis, data flow analysis, metrics analysis, peer code reviews, unit testing, code coverage analysis, traceability, and other software verification practices. Development testing helps to reduce the effect of software errors, and it speed the delivery of new features and bug fixes to customers

8.2.1.1. Performance

Performance testing is a form of software testing that focuses on how a system running the system performs under a particular load. In this project, due to the nature of the platform used, it differs according to the user's individual computer performance, and in terms of functionality, it does not show much difference in any structure. Only simple network requests are used in server, so performance of main function will decide entire system performance.

8.2.1.2. Reliability

Software reliability is the probability of failure-free operation of a computer program for a

specified period in a specified environment. Reliability is a customer-oriented view of software quality. It relates to operation rather than design of the program, and hence it is dynamic rather than static. In this project, VRCHAT platforms reliability is critical for entire reliability of system.

8.2.1.3. Security

Software security is the application of techniques that assess, mitigate, and protect software systems from vulnerabilities. These techniques ensure that software continues to function and are safe from attacks. Developing secure software involves considering security at every stage of the life cycle. The major goal is to identify flaws and defects as early as possible. In our system should pay attention to security, not only our school students but also all user in VRCHAT . our system using VRCHAT platform, this point make advantage for system security. In VRCHAT platform has issues about security, so this point makes changes of policies.

8.2.2. Release Testing

Release testing refers to coding practices and test strategies that give teams confidence that a software release candidate is ready for users. Release testing aims to find and eliminate errors and bugs from a software release so that it can be released to users. In this project, main product is “World for VRCHAT”. Our file should be checked for uploading regulation of VRCHAT platform. After uploading, checking whether world in VRCHAT is operating as intended is main goal of release testing.

8.2.3. User Testing

User testing is the process through which the interface and functions of a website, app, product, or service are tested by real users who perform specific tasks in realistic conditions. The purpose of this process is to evaluate the usability of that website or app and to decide whether the product is ready to be launched for real users. We need to focus on main functionality of systems. There are four subsystem for users. In user testing, check the system is operating well and which system is suitable for virtual system or not.

8.2.4. Testing Case

Test cases are setting for checking functionality, performance, and security. Set up 5 test cases on each subsystem, test the entire system, and make the validation sheet.

9. Development Plan

9.1. Objectives

This chapter illustrates the environment for the development of the product. For frontend environment Unity and VRCHAT SDK are used for UI/UX Design and functions. For backend Flask server and MySQL DBMS are used for data communication.

9.2. Frontend Environment

9.2.1. UNITY (UI/UX Design)

Figure 31 Unity logo

Unity is a cross-platform game engine developed by Unity Technologies. The engine can be used to create 3D and 2D games, as well as interactive simulations and other experiences. The engine offers a primary scripting API in C#, for both the Unity editor in the form of plugins, and games themselves, as well as drag and drop functionality.

9.2.2. VRCHAT SDK3 for UNITY(UI/UX Design)

Figure 32 VRCHAT logo

VRCHAT is an online virtual platform. The platform allows users to interact with others with user-created 3D avatars and worlds.

The VRCSDK3-Avatars package comes with Avatars 3.0, the latest avatar framework we offer for creation of both basic and advanced avatars with full customization. Check out Avatars 3.0 to learn more.

VRCSDK3-Worlds comes pre-packaged with VRChat Udon for programming advanced actions. To learn more about Udon, check out our Udon section in this documentation.

9.3. Backend Environment

9.3.1. Github

Figure 33 Github logo

Github is a provider of Internet hosting for software development and version control usting Git. It offers the It offers the distributed version control and source code management (SCM) functionality of Git, plus its own features. It provides access control and several collaboration features

9.3.2. MySQL(DBMS)

Figure 34 MySQL logo

MySQL is an open-source relational database management system. MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often, MySQL is used with other programs to implement applications that need relational database capability.

9.3.3. Flask (SERVER)

Figure 35 Flask logo

Flask is a micro web framework written in Python. It is classified as
a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

9.4. Constraints

The system will be designed by Unity development tool and operating on VRCHAT platform. There are some constraints for these points. VRCHAT is being updated steadily. Changes in policy and the availability of functions accordingly should be checked.

- Consider the VRCHAT's limitation for security.
- Use open source functions for Unity.
- Develop environment is different with operation environment(Unity-VRCHAT)
- Develop with VRCHAT SDK3
- Develop with Unity version (2019.4.30f1 (64-bit))

9.5. Assumptions and Dependencies

The system will be designed by Unity development tool and operating on VRCHAT platform for PC. For VRCHAT, use Unity version (2019.4.30f1), programmed with C#.

10. Supporting Information

10.1. Software Design Specification

This document is following basic form of IEEE Recommended Practice for Software Design Description.

10.2. Document History

Table 35 Document History

Date	Version	Description	Writer
2021/10/25	1.0	Make document	Keunha Kim
2021/10/28	1.1	Addition of 8, 9	Keunha Kim
2021/10/29	1.2	Adding Database Design	Fatdzirul Izzat
2021/10/29	1.3	Adding descriptions about meeting room component	Minsu Kim
2021/10/30	1.4	Adding descriptions about reading room component	Ukcheol Choe
2021/10/30	1.5	Addition of 1,2	Keunha Kim
2021/10/30	1.6	Add descriptions about DVD room component	Dasol Lee
2021/10/30	1.7	Adding overall system architecture	Gyeonghyeon Cho
2021/10/31	2.0	Contents table organization	Keunha Kim