SKKU Online Reading Room

Software Requirement Specification

2021.04.25.

TEAM 8

Team Leader	장영재
Team Member	박윤진
Team Member	김지수
Team Member	박세연

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. Preface

이 장은 이 소프트웨어 디자인 문서에 대한 전반적인 소개를 포함하고 있다.

1.1. Readership

이 소프트웨어 디자인 명세서는 총 10장으로 구성되어 있다. 문서의 구성에 관한 세부적인 정보는 1.4에서 확인할 수 있다. 이 문서의 주요 독자는 소프트웨어공학 개론 수업의 8조이며, 수업의 교수님과 조교님 및 다른 수업의 학우들도 주요 독자가 될 수 있다.

1.2. Scope

이 문서는 Software Engineering 과정에서 성균관대학교 온라인 독서실 시스템을 구현하기 위해 디자인을 정의하는 목적으로 사용된다.

1.3. Objective

이 소프트웨어 디자인 명세서는 성균관대학교 온라인 독서실 시스템의 설계에 대한 기술적 설명을 제공한다. 이 문서는 온라인 독서실 구현을 위한 소프트웨어 구조와 디자인에 대해 설명하고 전체 시스템 구성에 대한 overview를 제공한다. 이 문서는 성균관대학교 온라인 독서실 시스템의 이해 관계자, 개발자, 디자이너를 대상으로 한다.

1.4. Document Structure

- **1. Preface**: 이 장에서는 예상 독자층, scope, 시스템의 목적, 이 명세서의 구조에 대해 설명한다.
- **2. Introduction**: 이 장에서는 이 명세서에 사용된 여러가지 diagram과 이

프로젝트의 목적에 대해 설명한다.

- **3. Overall System Architecture:** 이 장에서는 context diagram, sequence diagram, use case diagram을 통해 전체적인 시스템의 구조를 설명한다.
- **4. System Architecture – Frontend:** 이 장에서는 class diagram과 sequence diagram을 사용하여 Frontend 시스템의 구조를 설명한다.
- **5. System Architecture – Backend:** 이 장에서는 class diagram과 sequence diagram을 사용하여 Backend 시스템의 구조를 설명한다.
- **6. Protocol Design:** 이 장에서는 클라이언트와 서버의 통신에 사용되는 protocol 디자인을 설명한다.
- **7. Database Design:** 이 장에서는 ER diagram과 SQL DDL을 사용하여 데이터베이스의 디자인을 설명한다.
- **8. Testing Plan:** 이 장에서는 시스템의 테스트 계획을 설명한다.
- **9. Development Plan:** 이 장에서는 시스템을 개발하기 위해 사용되는 tool과 제약사항, 가정, 종속성을 설명한다.
- **10. Supporting Information:** 이 장에서는 이 문서의 기준과 history를 설명한다.

2. Introduction

이 프로젝트의 목적은 성균관대학교 학생들을 위한 온라인 독서실 모바일 애플리케이션을 개발하는 것이다. 이 애플리케이션은 수업 별 오픈 채팅 기능과 카메라

를 통한 온라인 독서실 기능을 지원해야 한다. 이 문서는 프로젝트를 구현하기 위해 사용되는 디자인을 설명한다.

2.1. Objectives

이 장에서는 이 명세서에 사용된 여러가지 diagram에 대해 설명한다.

2.2. Applied Diagrams

2.2.1. Use case Diagram

Use case diagram은 시스템의 high-level의 요구사항을 분석하기 위해 사용되며 시스템과 환경 간의 상호작용을 보여준다. 이것은 원래 요구사항 획득을 지원하기 위해 개발되었지만 현재는 UML(Unified Modeling Language)에 통합되었다. 각각의 Use case는 시스템과의 외부 상호 작용을 포함하는 개별적인 작업을 나타내며 기능적 요구사항, 행위자, 행위자와 Use case 사이의 관계로 구성되어 있다. Use case에서 행위자는 사람 또는 다른 시스템이 될 수 있다.

2.2.2. Sequence Diagram

Sequence diagram은 UML의 한 종류로 시스템에서 행위자와 object의 상호작용을 순차적으로 보여준다. 이 diagram에서 object는 상단에 가로로 배열되며 시간은 위에서 아래로 순차적으로 표현된다. 상호작용은 화살표로 표시된다.

2.2.3. Class Diagram

Class diagram은 시스템의 object class와 이 class들 사이의 관계를 보여주며 객체지향적 시스템을 개발할 때 사용된다. Class는 3개의 부분으로 나누어져 있는 직사각형으로 표현되며 맨 위의 부분에는 class 이름, 가운데 공간에는 class의 속성,

마지막 부분에는 class의 method 등이 표시된다. 각 class 사이의 관계는 직사각형들을 연결하는 연결선으로 표시된다.

2.2.4. Context Diagram

Context diagram은 가장 높은 level의 data flow diagram이며, 주로 프로젝트 초기에 사용된다. 이 diagram은 시스템과 외부 entity들 간의 데이터 흐름을 나타내며 일반적으로 요구사항 문서에 포함된다. 또한 시스템과 상호 작용할 수 있는 모든 외부 entity를 나타내며 전체 소프트웨어 시스템이 단일 프로세스로 표시된다.

2.2.5. Entity Relationship Diagram

Context diagram은 가장 높은 level의 data flow diagram이며, 주로 프로젝트 초기에 사용된다. 이 diagram은 시스템과 외부 entity들 간의 데이터 흐름을 나타내며 일반적으로 요구사항 문서에 포함된다. 또한 시스템과 상호 작용할 수 있는 모든 외부 entity를 나타내며 전체 소프트웨어 시스템이 단일 프로세스로 표시된다.

2.3. Project Scope

성균관대학교 온라인 독서실 시스템은 성균관대학교 학생들을 대상으로 카메라를 통한 온라인 독서실 서비스를 제공하고 수업 별 오픈 채팅 서비스를 제공한다. 이 시스템은 성균관대학교로부터 수업 정보를 불러오고 서비스에 등록한 학생들의 정보를 데이터베이스에 저장하여 학생들에게 유용한 서비스를 무료로 제공하는 것을 목적으로 한다.

2.4. References

- Team1, Software Design Document, SKKU. (2020)

- 이은석 교수님, System Modeling, SKKU 소프트웨어공학개론 강의자료. (2021).

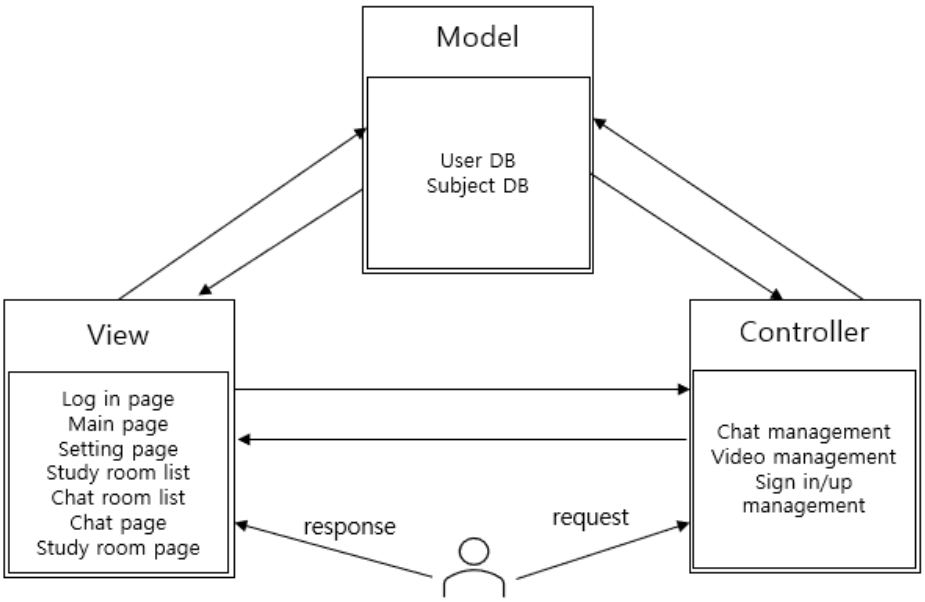
3. System Architecture – Overall

3.1. Objectives

프로젝트를 어플리케이션의 시스템 구성에 대해 frontend와 backend에 걸쳐서 설명한다.

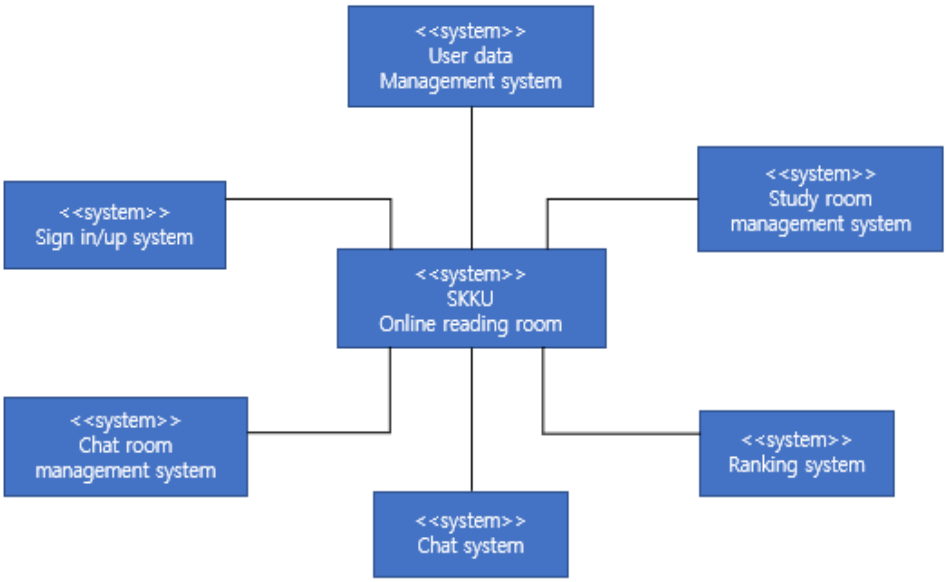
3.2. System Organization

이 서비스는 클라이언트-서버 모델을 적용하여 설계되었다. 사용자는 Front-end 를 통해 애플리케이션과 상호작용한다. 애플리케이션의 Front-end와 Back-end는 JSON 기반의 HTTP 통신을 통해 데이터를 주고 받는다. Back-end 는 설계 사용자의 요청을 Front-end에서 받아 컨트롤러로 배포한다. 컨트롤러는 데이터베이스에서 필요한 객체 정보를 얻어서 데이터베이스에서 처리한 후 JSON 형식으로 전달한다. 전달 받은 JSON 형식의 데이터에 따라 스터디나 채팅 목록을 사용자에게 응답으로 제공한다.



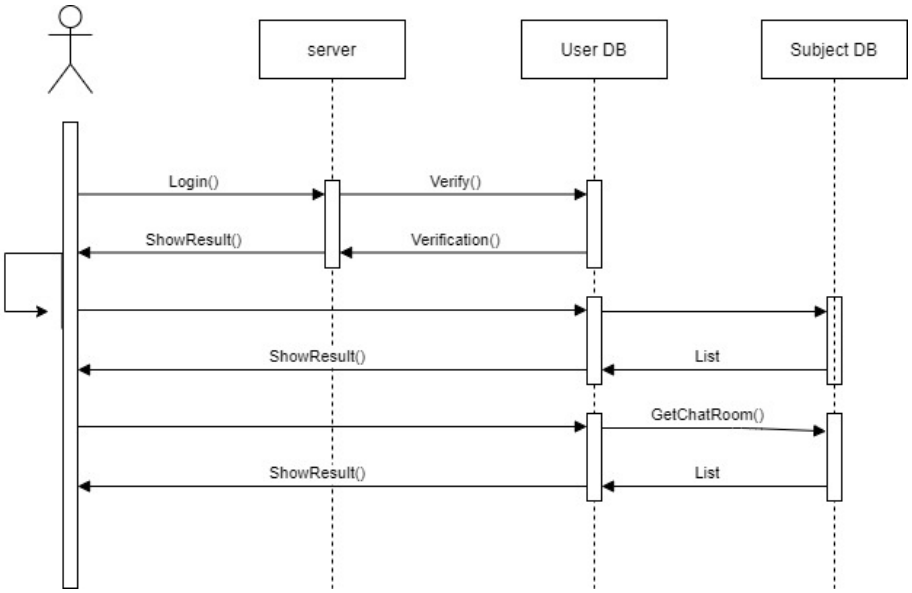
[Figure 1] Overall system architecture

3.2.1. Context Diagram



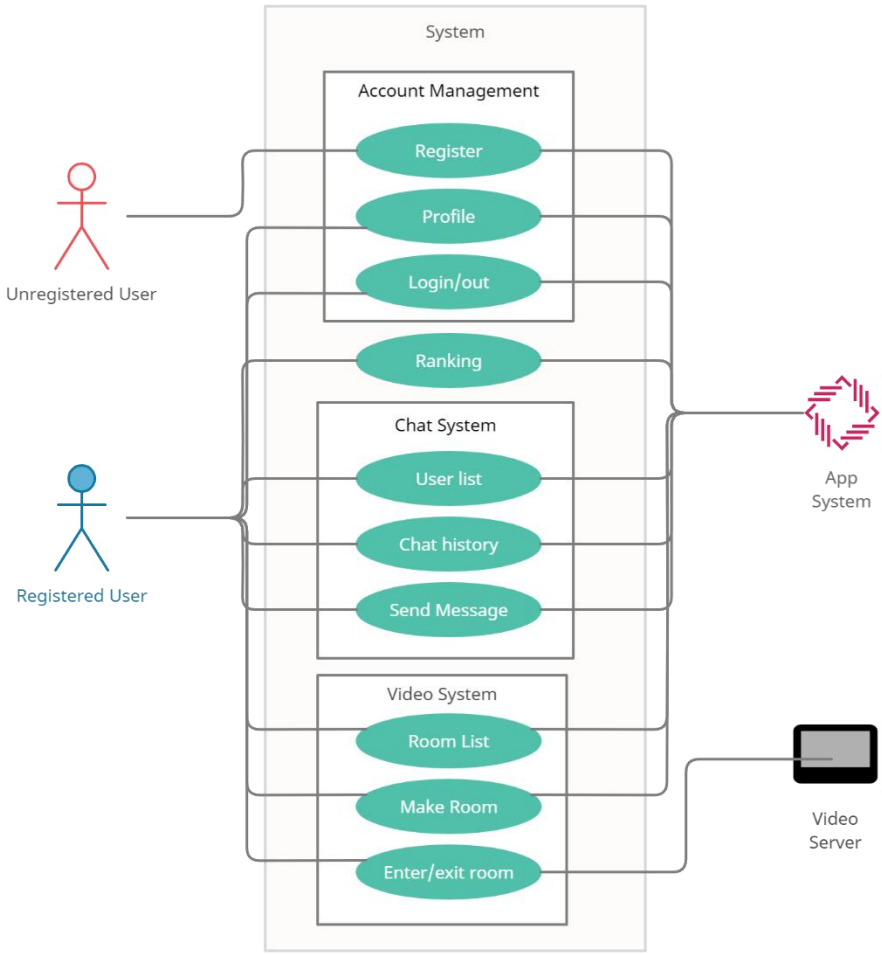
[Figure 2] Overall context diagram

3.2.2. Sequence Diagram



[Figure 3] Overall sequence diagram

3.2.3. Use Case Diagram



[Figure 4] Use case diagram

4. System Architecture – Frontend

4.1. Objectives

이 장에서는 Front-end의 시스템 구조, 특성 및 기능을 설명하고 각 구성 요소의 관계를 설명한다.

4.2. Subcomponents

4.2.1. Profile

Profile 클래스는 사용자 정보를 처리한다. 사용자 등록 후 사용자는 프로필 정보를 입력해야 한다. 사용자는 로그인 후 프로필을 수정할 수 있다.

4.2.1.1. Attributes

+ user_id : 사용자가 로그인할 때 사용하는 id

+ major_name : 사용자의 전공

+ ranking : 공부시간에 따른 사용자 순위

+ name : 사용자 실제 이름

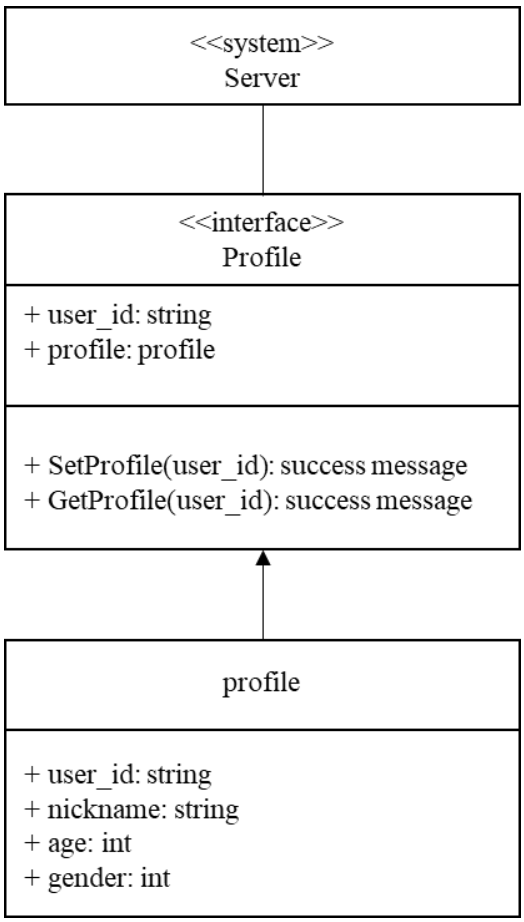
+ is_online : 사용자 접속 여부

4.2.1.2. Methods

+ SetProfile() : 사용자 프로필을 등록한다.

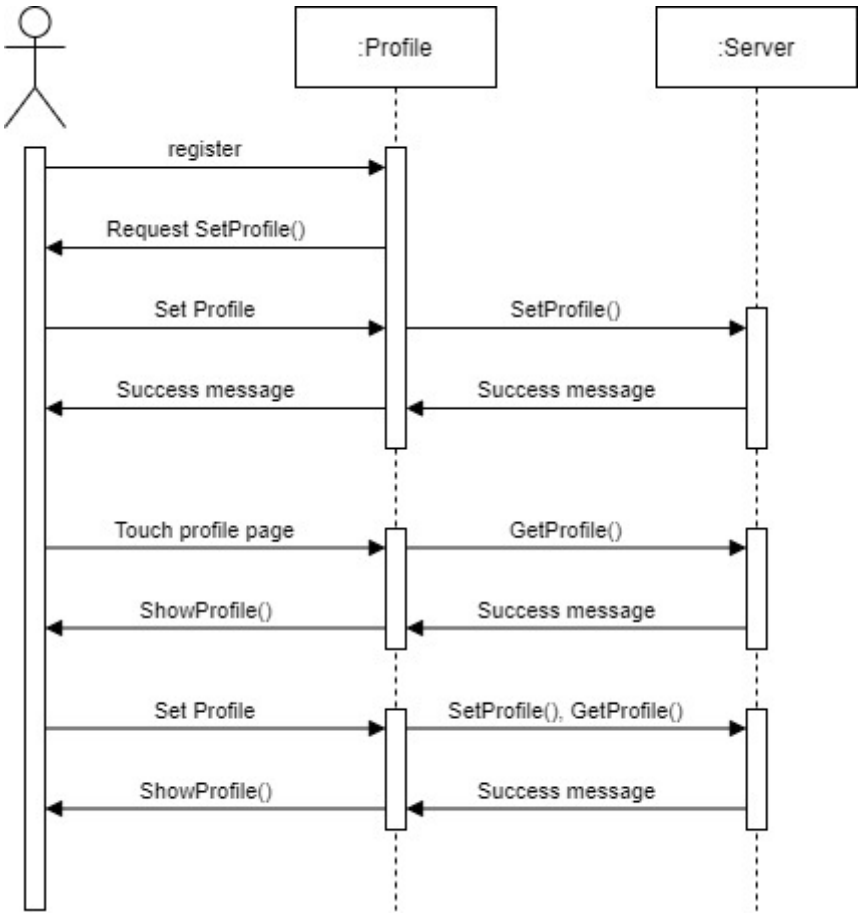
+ GetProfile() : 사용자 프로필 정보를 불러온다.

4.2.1.3. Class Diagram



[Figure 5] Class diagram – Profile

4.2.1.4. Sequence Diagram



[Figure 6] Sequence diagram – Profile

4.2.2. Chat

Chat 클래스는 사용자 간의 채팅 프로그램을 제공한다. 사용자 아이디 확인을 통해 로그인한 사용자 간에 메시지를 보내고 받을 수 있다.

4.2.2.1. Attribute

+host : 로컬 IP

+port : 포트번호

+user_id : 사용자 id

+message : 전송 메시지

4.2.2.2. Methods

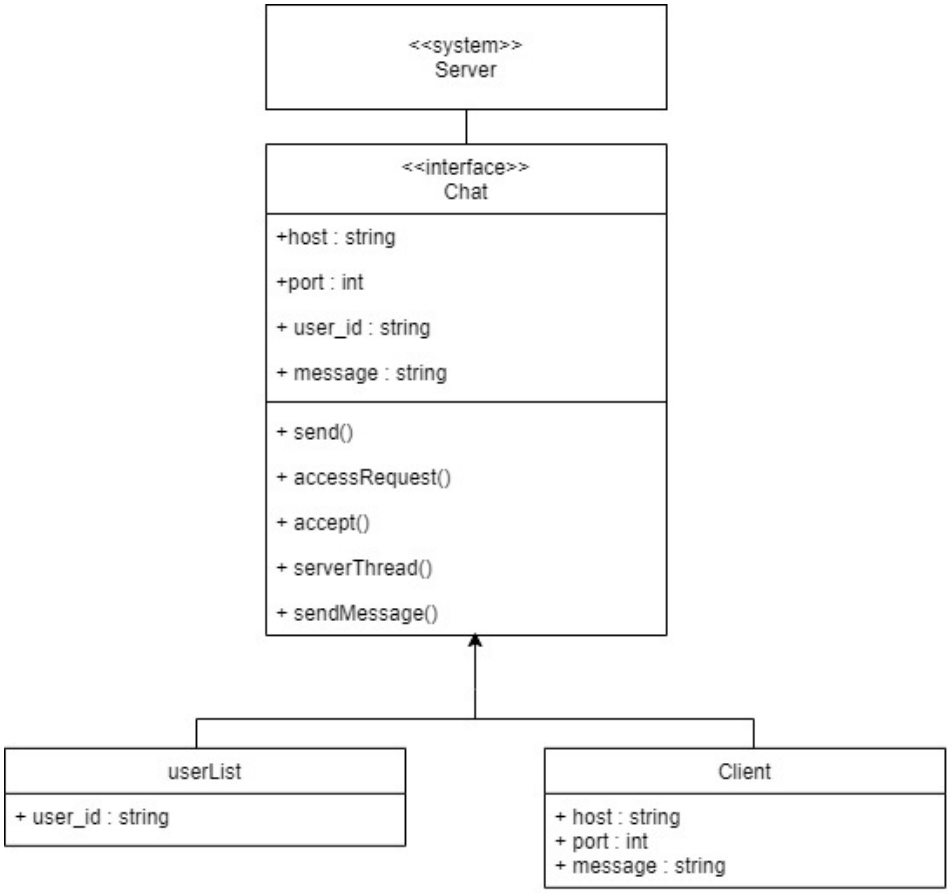
+accessRequest() : 메시지 전송 요청

+accept() : 요청 허용

+serverThread() : 서버 쓰레드 생성

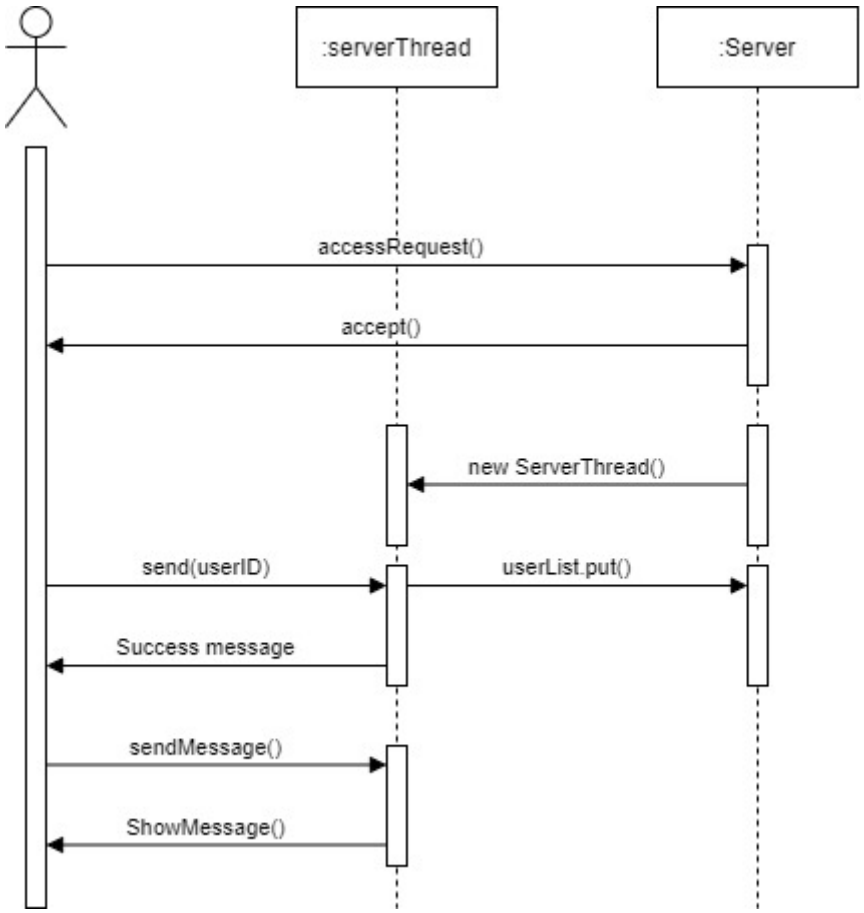
+sendMessage() : 메시지 전송

4.2.2.3. Class Diagram



[Figure 7] Class diagram – Chat

4.2.2.4. Sequence Diagram



[Figure 8] Sequence diagram – Chat

4.2.3. Streaming

Streaming 클래스는 실시간 영상을 서버에 제공하고 서버는 다시 그 영상을, 서버 접속자에게 제공한다.

4.2.3.1. Attributes

4.2.3.2. Methods

smServerInterface

+ Run()

+ AcceptThread()

+ RequestThread()

+MsgHndlr()

smClientInterface

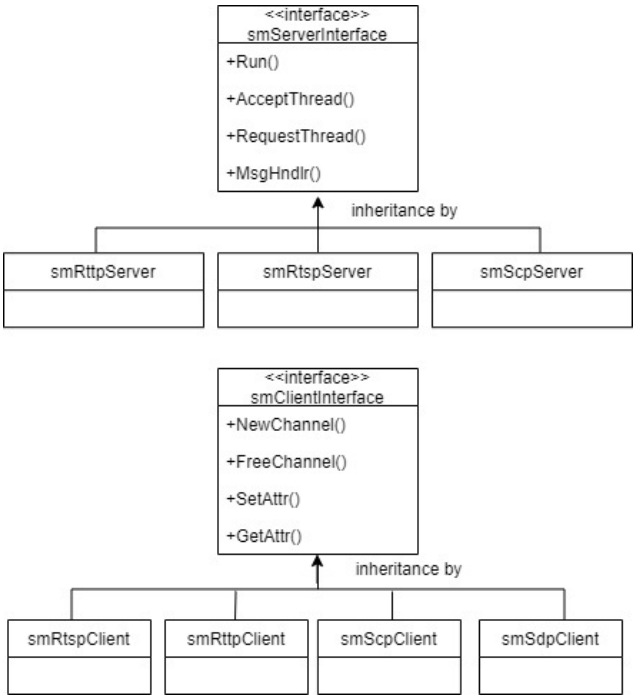
+NewChannel()

+FreeChannel()

+SetAttr()

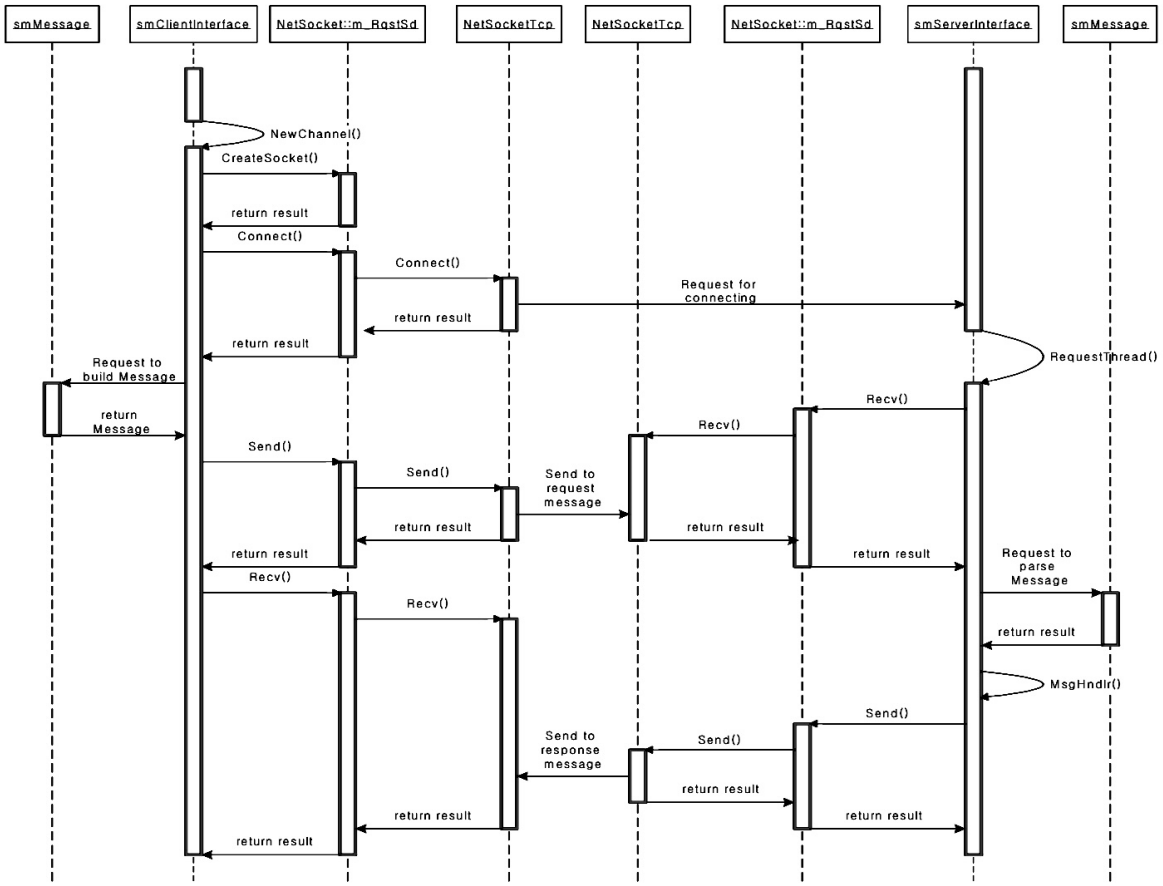
+GetAttr()

4.2.3.3. Class Diagram



[Figure 9] Class diagram – Streaming

4.2.3.4. Sequence Diagram



[Figure 10] Sequence diagram – Streaming

4.2.4. Study Room

StudyRoom 클래스는 사용자의 스터디룸 목록을 불러와서 리스트로 보여준다. 스터디룸을 만들 시에는 서버의 스터디 목록에 추가한다.

4.2.4.1. Attributes

+ user_id : 사용자 id

+ major_name : 사용자 전공

+ studyroom : 사용자 스터디룸 리스트

+ time : 사용자 접속 시간

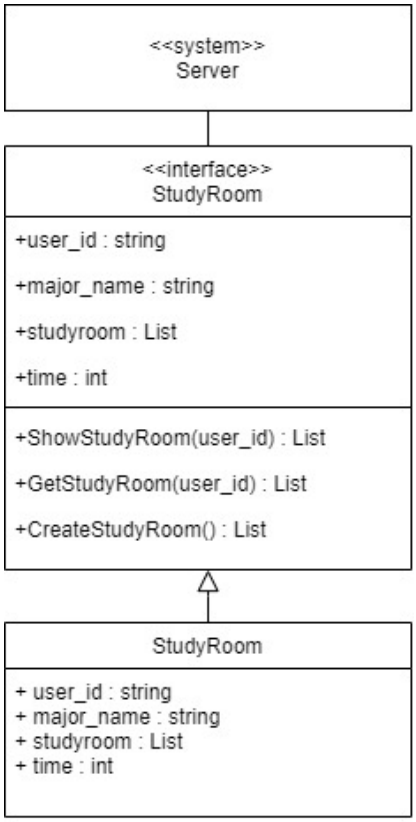
4.2.4.2. Methods

+ ShowStudyRoom() : 스터디룸 리스트를 보여준다.

+ GetStudyRoom() : 스터디룸 리스트를 서버에서 가져온다.

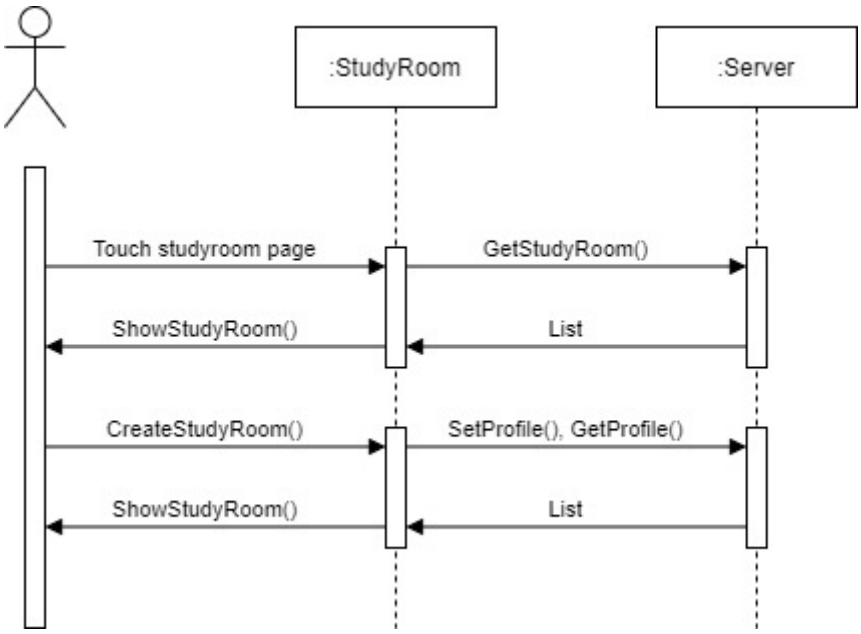
+ CreateStudyRoom() : 새로운 스터디룸을 등록한다.

4.2.4.3. Class Diagram



[Figure 11] Class diagram – Study Room

4.2.4.4. Sequence Diagram



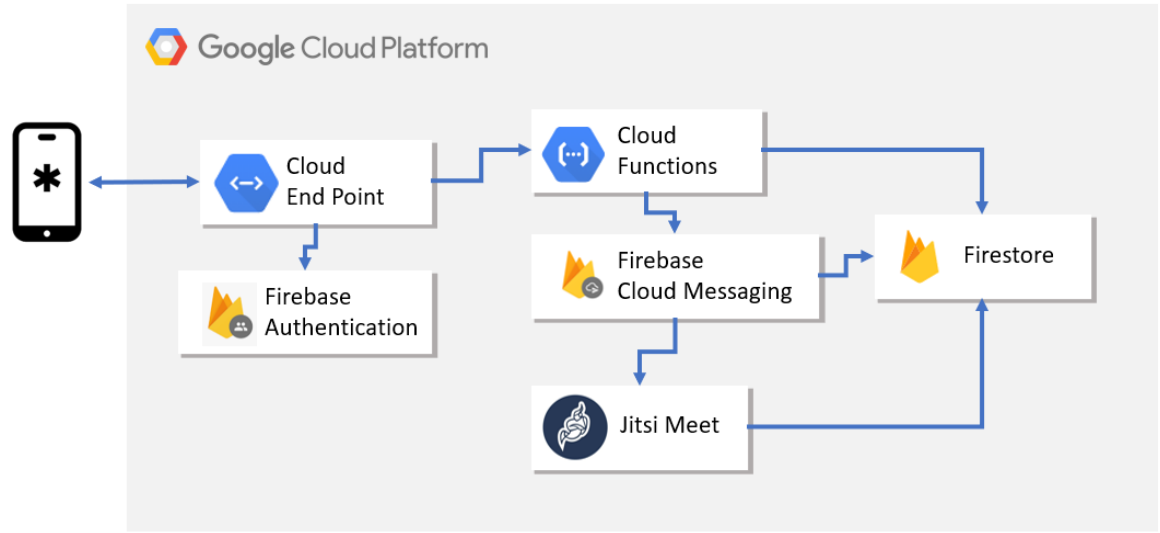
[Figure 12] Sequence diagram – Study Room

5. System Architecture – Backend

5.1. Objectives

이 챕터는 백엔드 시스템의 아키텍처, 즉 DB와 클라우드 API들에 대하여 기술한다.

5.2. Overall Architecture

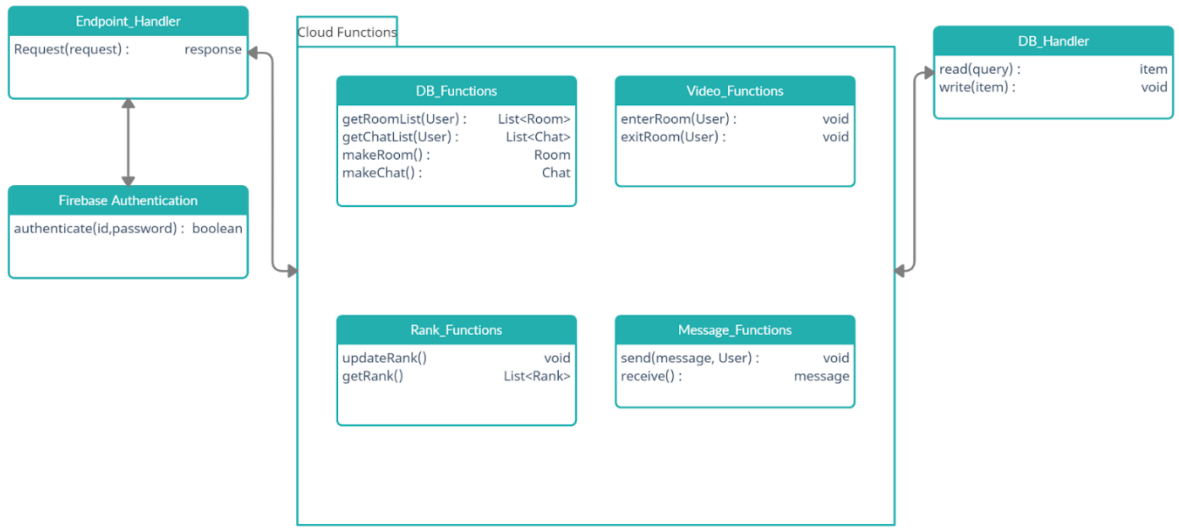


[Figure 13] Overall architecture

시스템의 전반적인 아키텍처는 다음과 같다. 클라우드 엔드 포인트에서 시스템의 프런트 엔드로부터 API 콜을 받아서 클라우드 컨트롤러로 넘겨주게 된다. 컨트롤러에서 API콜에 해당하는 클라우드 API를 사용해서 요청을 수행하게 된다. 어텐티케이션 같은 경우는 외부 API를 사용하는 형식이며, 이는 파이어베이스에서 제공하는 기능을 사용하게 된다. 시스템의 주요 기능들은 실시간 채팅과 온라인 독서실이며, 실시간 채팅 같은 경우에는 파이어베이스 클라우드 메세징 기능으로 구현이 되며, 온라인 독서실은 "Jitsi"라는 오픈소스 화상회의 서버를 사용해서 구성되게 된다.

5.3. Subcomponents

5.3.1. Cloud Function



[Figure 14] Class diagram – Cloud functions

5.3.1.1. Endpoint Handler Class

API 게이트웨이 역할을 하며, 프런트 엔드로부터 오는 request들을 처리한다.

5.3.1.2. Firebase Authentication Class

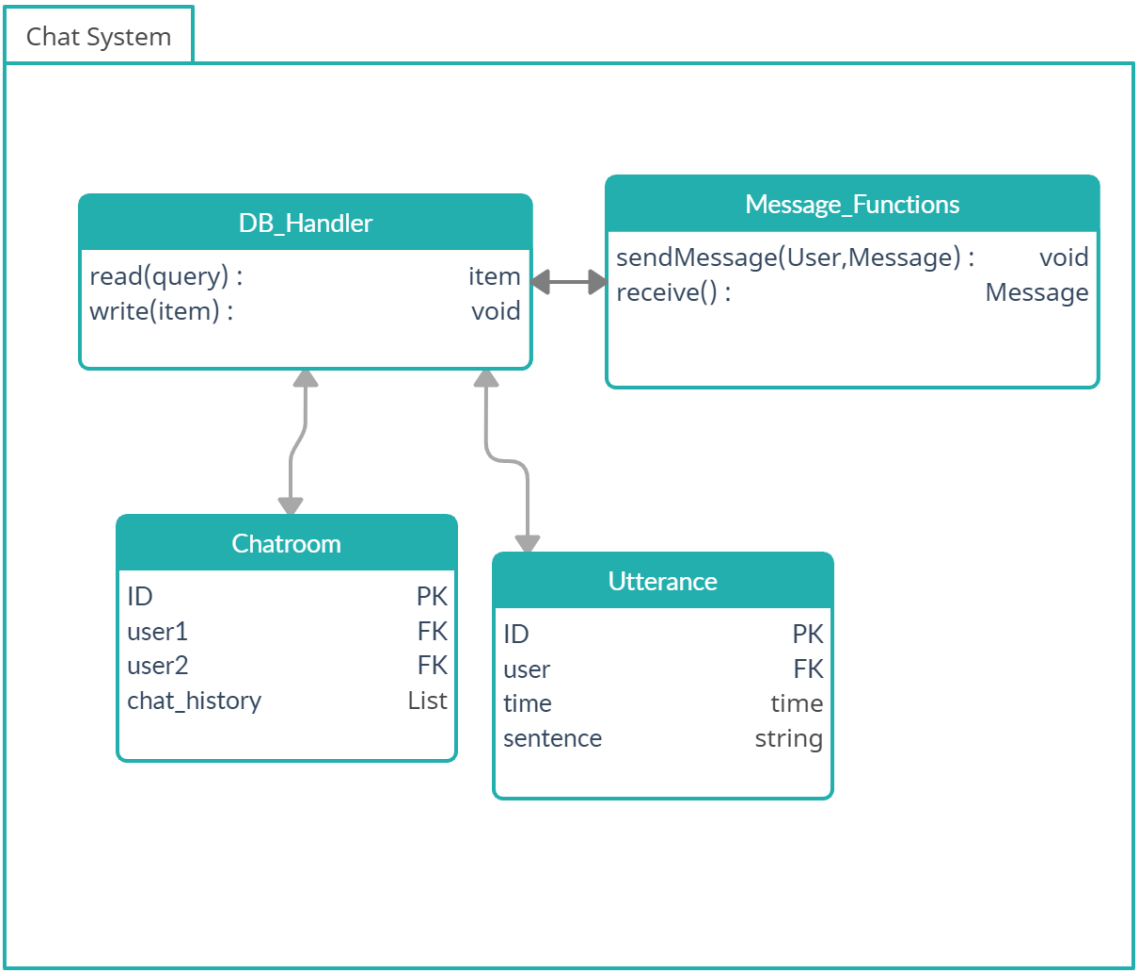
Firebase에서 제공해주는 Authentication 기능을 사용하게 된다.

5.3.1.3. DB_Handler Class

DB 입출력을 관리하는 클래스. 특정 정보를 가져오기도 하고, 전체 오브젝트들을 리스트로 가져오기도 한다. DB 저장도 DB 핸들러를 사용해서 하게된다.

5.3.2. Chat System

5.3.2.1. Class Diagram

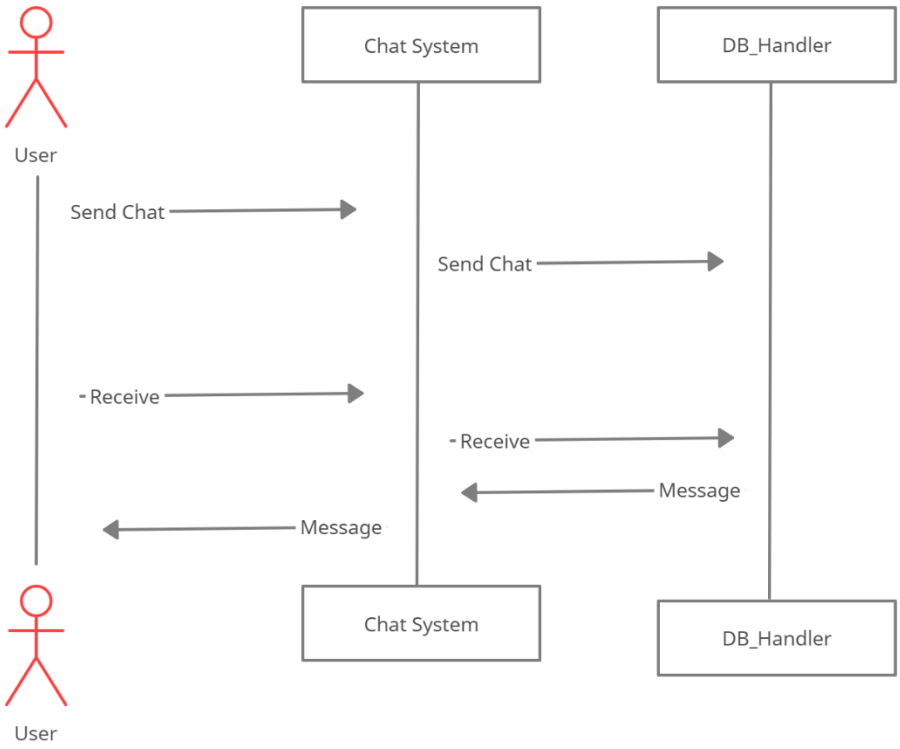


[Figure 15] Class diagram – Chat system

- Class Description

  ✓ **Chat Class**: 채팅 API들을 담고 있는 클래스. sendMessage와 receive를 DB_Handler로 요청해서 필요한 정보들을 DB로부터 read, write 한다.

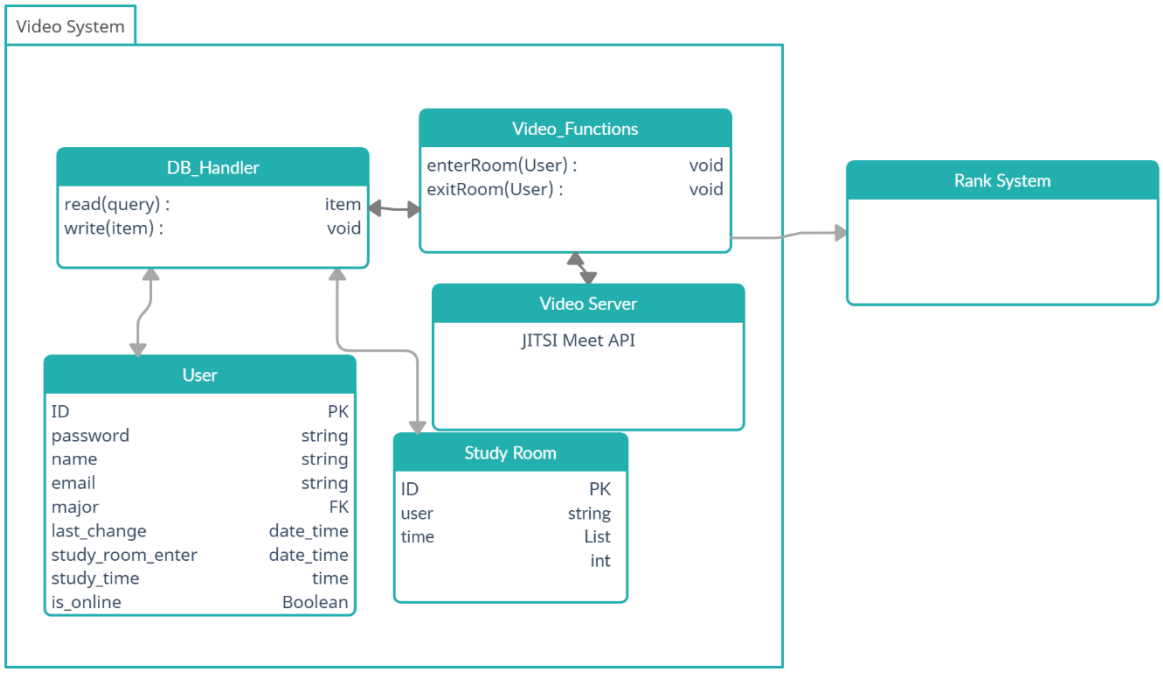
5.3.2.2. Sequence Diagram



[Figure 16] Sequence diagram – Chat system

5.3.3. Video Conference System

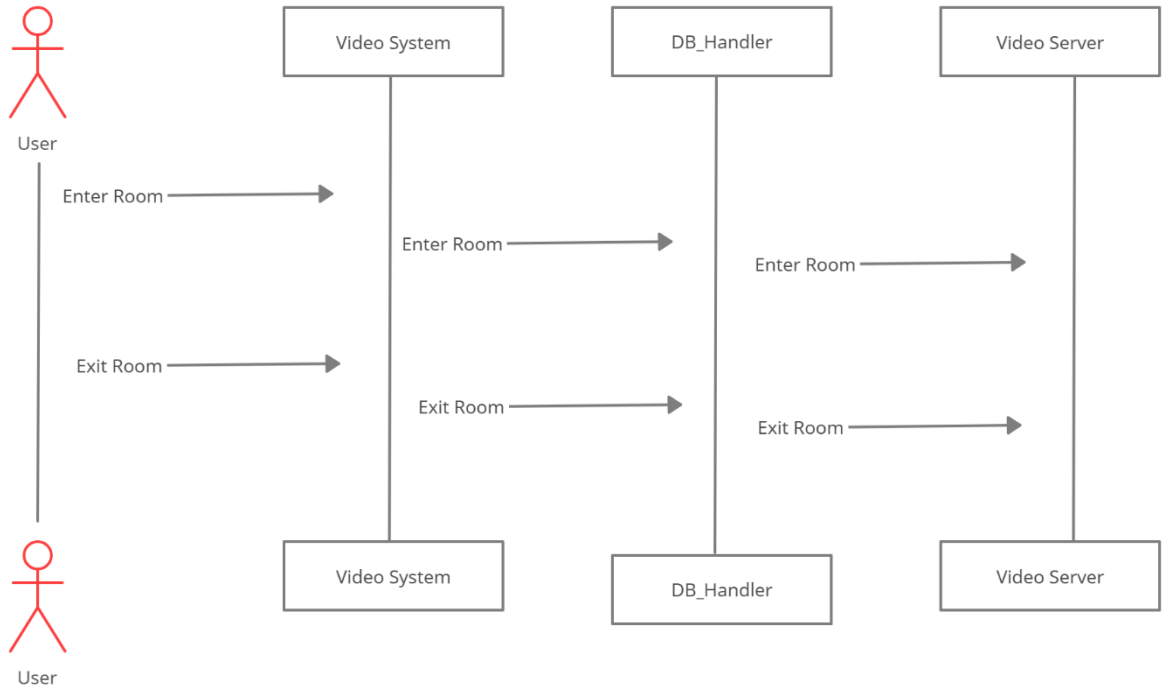
5.3.3.1. Class Diagram



[Figure 17] Class diagram – Video Conference system

- Class Description
  - ✓ **Video Conference Class**: 온라인 독서실 API들을 담고 있는 클래스. Jitsi Meet Server에 접속하기 위한 정보들을 담고 있으며, 추가적으로 Rank System을 위해서 개개인의 공부시간을 구해서 rank를 업데이트 하는

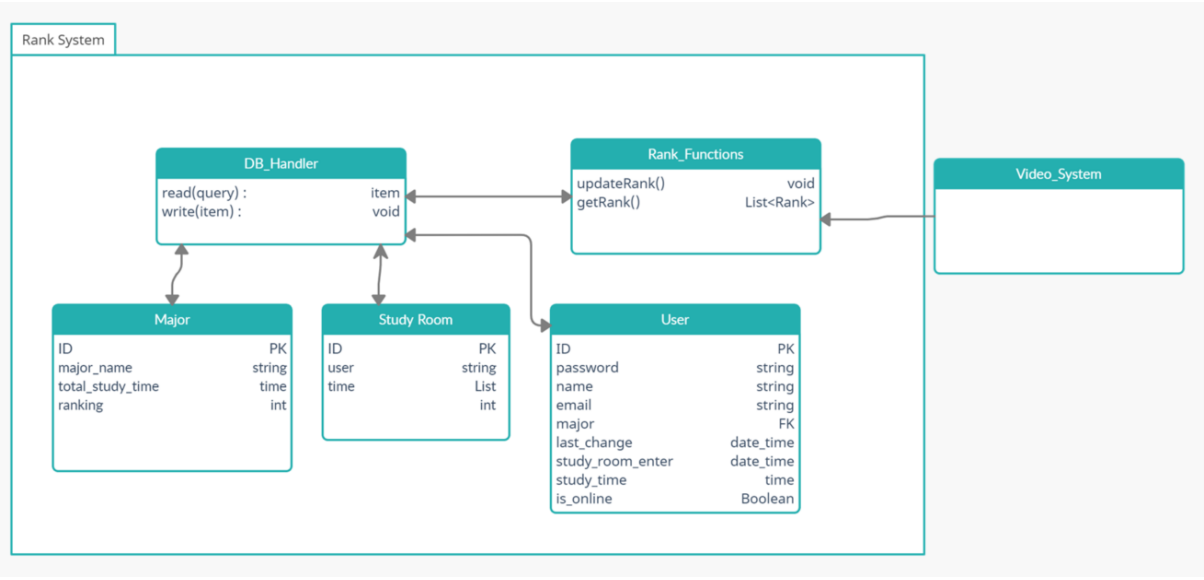
5.3.3.2. Sequence Diagram



[Figure 18] Sequence diagram – Video Conference system

5.3.4. Rank System

5.3.4.1. Class Diagram

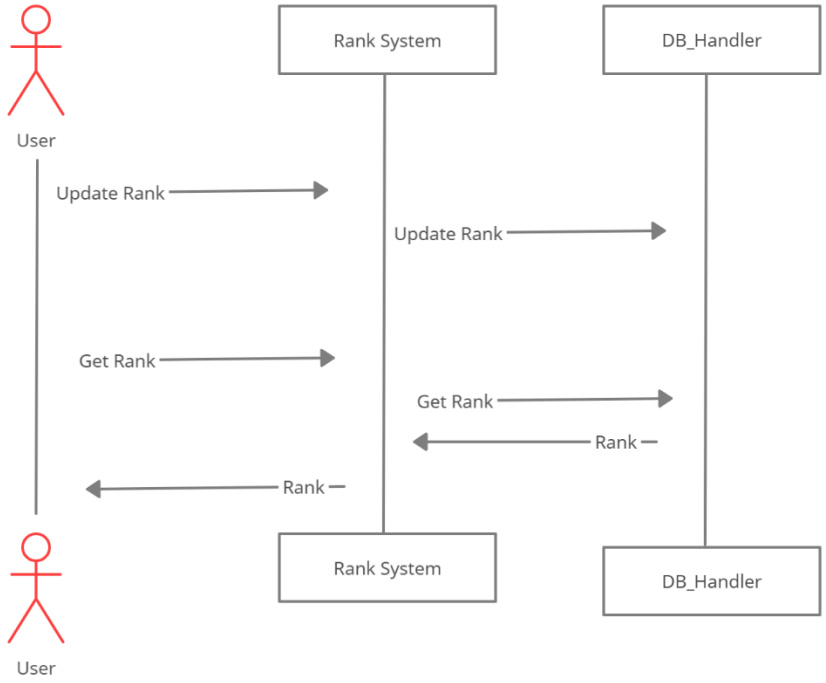


[Figure 19] Class diagram - Rank system

- Class Description

  ✓ **Rank System Class**: 온라인 독서실을 참여했을 시, 공부시간을 기록하고, 전공별 학생들의 공부시간을 전부 취합해서 가장 많이 공부를 하는 전공을 뽑을 수 있는 API로 이루어져있다.

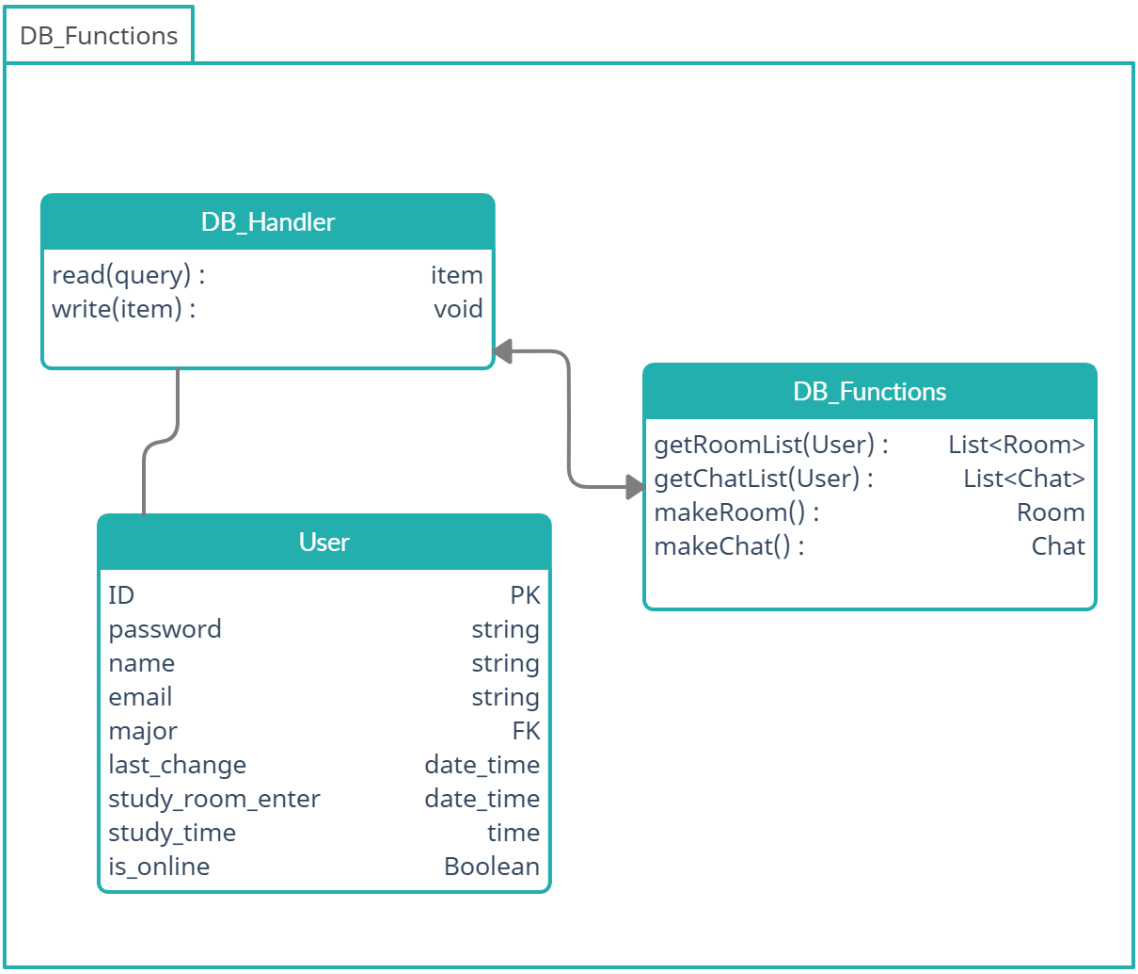
5.3.4.2. Sequence Diagram



[Figure 20] Sequence diagram – Recommendation system

5.3.5. DB System

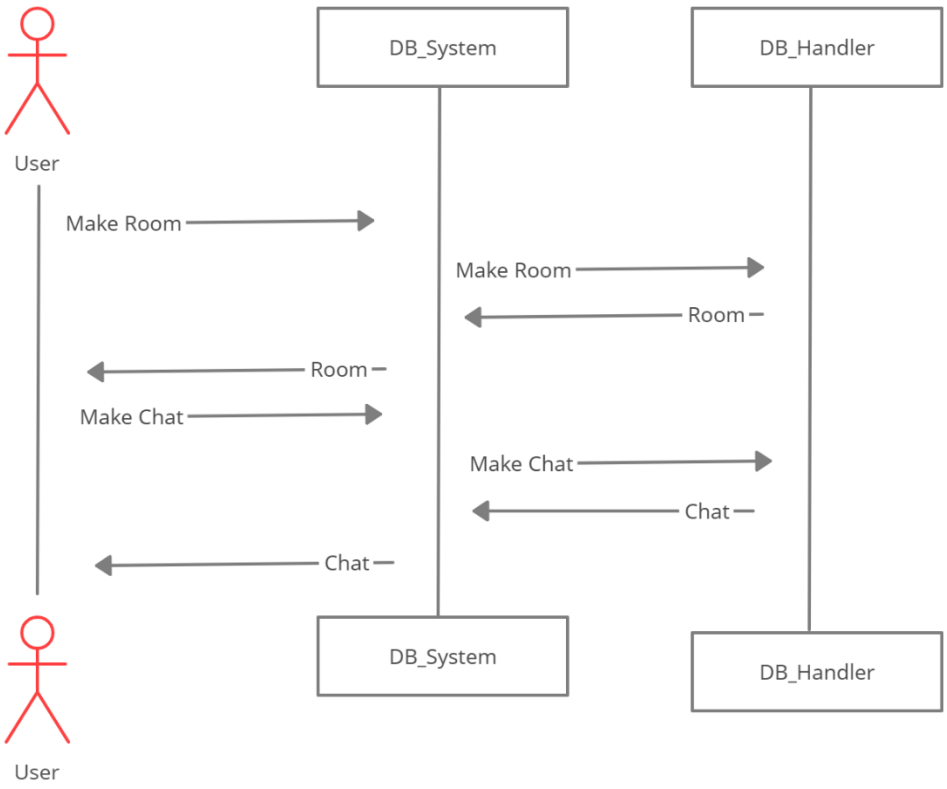
5.3.5.1. Class Diagram



[Figure 21] Class diagram – DB system

- Class Description

  ✓ **DB System Class**: 온라인 독서실 객체와 채팅 객체를 만들고 현재 DB 에 있는 객체의 목록을 조회할 수 있는 기능들이 포함된 클래스이다.



[Figure 22] Sequence Diagram – DB system

6. Protocol Design

6.1. Objectives

이 장에서는 각 subsystem의 상호작용에 사용된 프로토콜에 대해 설명하고 각 인터페이스가 어떻게 정의되었는지 설명한다.

6.2. JSON

JSON은 개방형 표준 파일 형식 및 데이터 교환 형식으로 텍스트를 사용하여 데이터 object를 저장하고 전송한다. JSON에서 사용되는 데이터 object는 속성-값 쌍, 배열 등으로 구성되어 있다. Firebase 실시간 데이터베이스 데이터는 모두 JSON 객체로 저장되며 SQL 데이터베이스와 달리 table이나 record가 존재하지 않는다.

6.3. OAuth

OAuth는 액세스 위임을 위한 개방형 표준으로 어떤 웹 사이트나 애플리케이션이 다른 웹 사이트의 정보에 액세스 하는 것을 허용하기 위해 사용된다

6.4. Authentication

6.4.1. Register

- Request

[Table 1] Table of register request

속성	세부사항	
Protocol	OAuth	
Request Body	Email	User email
	Request Token	Token for OAuth
	User	Basic information of the user

- Response

[Table 2] Table of register response

속성	세부사항	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Access Token	Token for access
	Message	Success message
Failure Response Body	Message	Fail message

6.4.2. Log In

- Request

[Table 3] Table of register request

속성	세부사항	
Protocol	OAuth	
Request Body	Email	User email
	Request Token	Token for OAuth
	User	Basic information of the user

- Response

[Table 4] Table of register response

속성	세부사항	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Access Token	Token for access
	Message	Success message
Failure Response Body	Message	Fail message

6.5. Profile

6.5.1. Set Profile

- Request

[Table 5] Table of set profile request

속성	세부사항	
Method	POST	
URI	/user/id/profile	
Parameter	User	Basic information of the user
Header	Authorization	User authentication

- Response

[Table 6] Table of set profile response

속성	세부사항	
Success Code	200 OK	
Failure Code	HTTP error code = 403 (Forbidden)	
Success Response Body	Message	Success message
Failure Response Body	Message	Failure message

6.5.2. Get Profile

- Request

[Table 7] Table of get profile request

속성	세부사항	
Method	GET	
URI	/user/id/profile	
Header	Authorization	User authentication

- Response

[Table 8] Table of get profile response

속성	세부사항	
Success Code	200 OK	
Failure Code	HTTP error code = 404 (Not Found)	
Success Response Body	User	User objects

6.6. Chat

6.6.1. Enter the chat room

- Request

[Table 9] Table of laptop recommendation request

속성	세부사항	
Method	GET	
URI	/chat/room/number	
Header	Authorization	User authentication

6.6.2. Make the chat room

- Request

[Table 10] Table of make the chat room request

속성	세부사항	
Method	GET	
URI	/chat/room/number	
Header	Authorization	User authentication

- Response

[Table 11] Table of make the chat room response

속성	세부사항	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Message	Success message
Failure Response Body	Message	Failure message

6.7. Study room

6.7.1. Enter the study room

- Request

[Table 12] Table of enter the study room request

속성	세부사항	
Method	GET	
URI	/studyroom/number	
Header	Authorization	User authentication

- Response

[Table 13] Table of enter the study room response

속성	세부사항	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Room_information	입장한 Study room에 대한 정보
Failure Response Body	Message	Fail message

6.7.2. Make the study room

- Request

[Table 14] Table of make the study room request

속성	세부사항	
Method	GET	
URI	/studyroom/number	
Header	Authorization	User authentication

- Response

[Table 15] Table of make the study room response

속성	세부사항
Success Code	200 OK

속성	세부사항	
Failure Code	HTTP error code = 400	
Success Response Body	Message	Success message
Failure Response Body	Message	Failure message

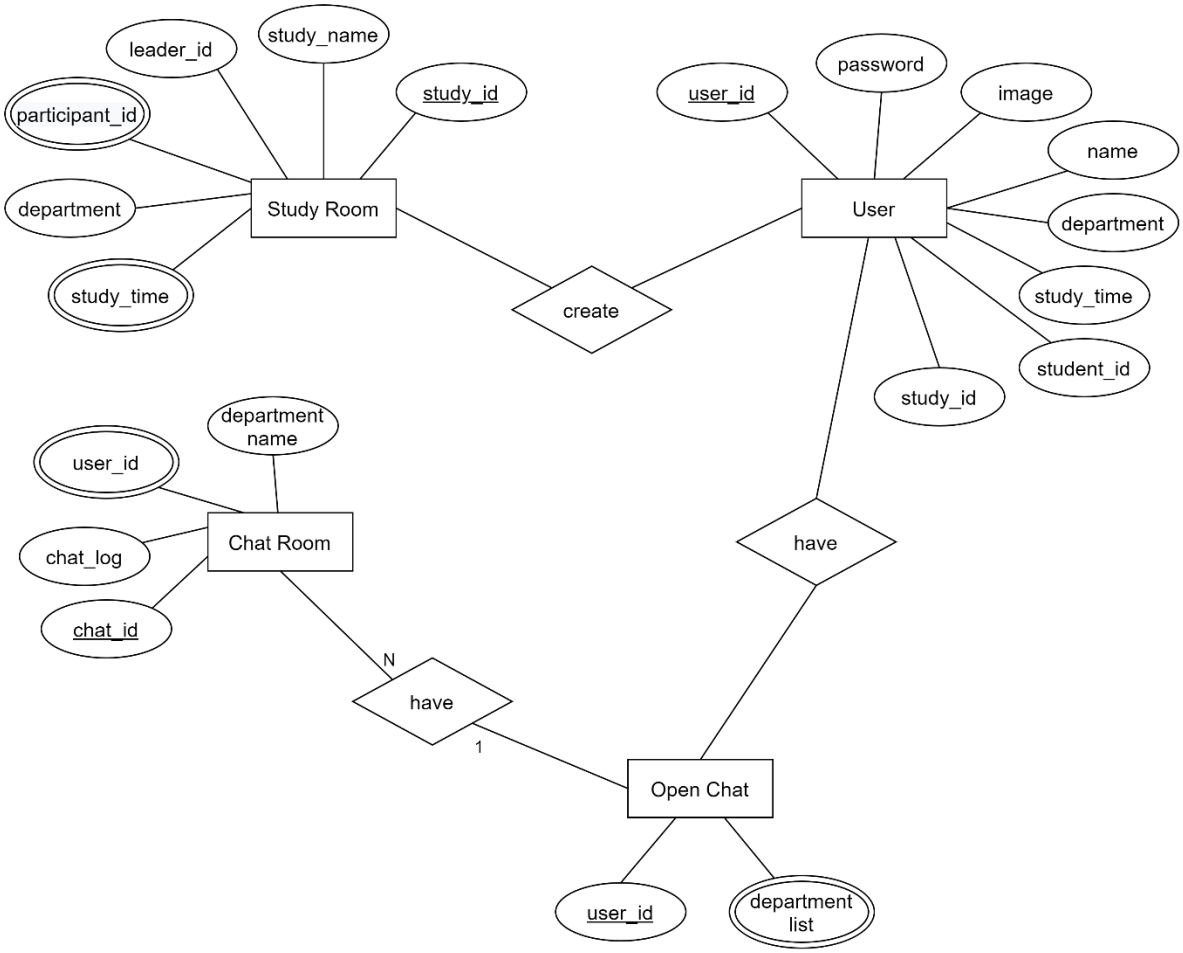
7. Database Design

7.1. Purpose

이 장에서는 시스템의 데이터 구조를 설명하고 데이터가 데이터베이스에 어떻게 저장되는지를 설명한다. ER 다이어그램과 Relational Schema를 이용해 데이터베이스 구조를 설명하고, SQL DDL을 정의한다.

7.2. ER Diagram

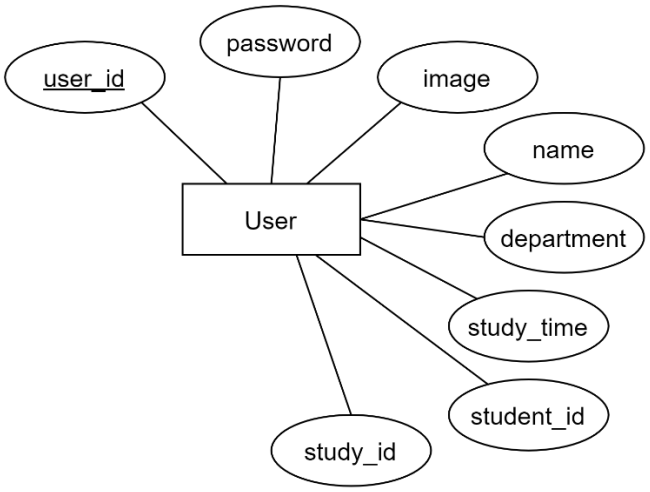
해당 시스템은 User, Study Room, Open Chat, Chat Room의 4가지 entity로 구성된다. 각 entity는 사각형으로, entity가 포함하는 속성은 타원으로 표현한다. 여기서 이중 실선으로 표현된 타원은 값을 여러 개 가질 수 있는 속성을 의미한다. 밑줄이 쳐진 속성 요소는 해당 entity를 유일하게 구분할 수 있는 속성을 의미한다.



[Figure 23] ER-diagram

7.2.1. Entities

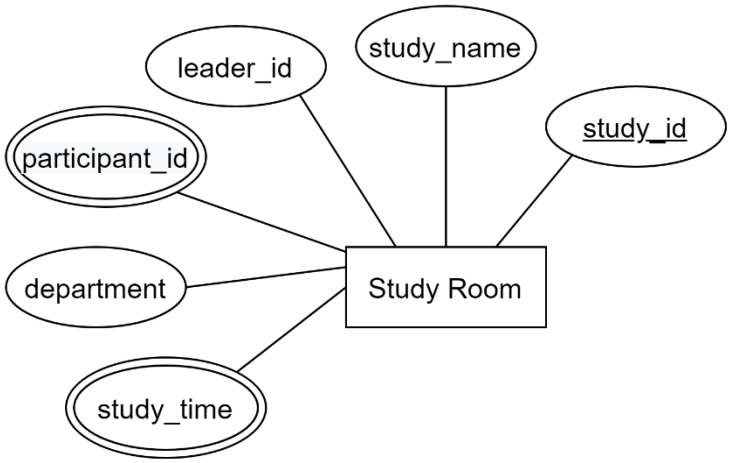
7.2.1.1. User



[Figure 24] ER-diagram, User

User 엔터티는 성균관대학교의 학생 인증을 마치고 'SKKU Online Reading Room' 서비스에 가입한 회원을 의미한다. User 엔터티는 user_id, password, image, name, department, study_time, student_id, study_id로 구성된다. 여기서 image, name, department, student_id 값은 성균관대학교 학생 정보 시스템에 저장된 내용을 얻어서 저장한다. user_id는 해당 서비스에 가입할 때 사용한 id를 의미하고, user 엔터티의 primary key이다. 각 user는 study room을 생성하거나 study room에 참여할 수 있다.

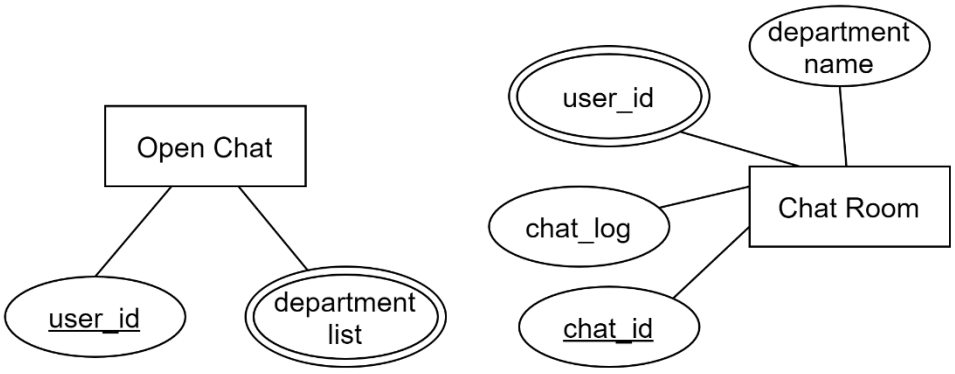
7.2.1.2. Study Room



[Figure 25] ER-diagram, Study Room

Study Room 엔터티는 user가 생성한 스터디를 의미한다. 해당 엔터티는 primary key인 study_id로 각 스터디를 유일하게 구분한다. 그 외에 속성으로 study_name, leader_id, participant_id, department, study_time을 갖는다. study_name은 서비스에서 보여지는 스터디의 이름이고, leader_id는 해당 스터디를 생성한 user의 user_id이다. department는 해당 스터디가 속한 학과명을 의미한다. 스터디에는 leader가 설정한 인원수만큼 user가 참여할 수 있으므로, participant_id에는 해당 스터디에 참여하고 있는 user들의 user_id를 저장한다.

7.2.1.3. Cart

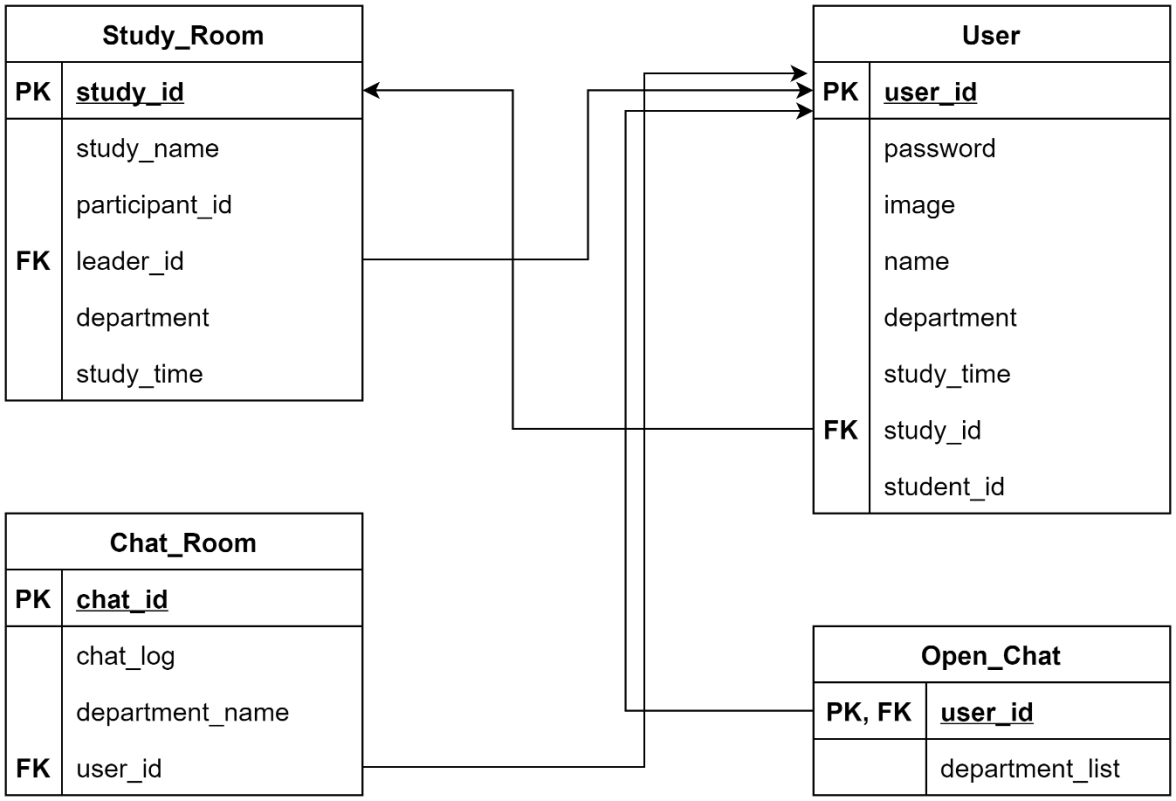


[Figure 26] ER-diagram, Open chat, Chat Room

Open Chat 엔터티는 user의 학생정보를 받아 현재 수강 중인 수업 목록을 받는다. 해당 수업 목록을 department list 속성에 저장한다.

Chat Room은 각 오픈 채팅방의 정보를 저장하는 엔터티로, 속성은 채팅에 참여하는 사용자들의 id인 user_id, 채팅 내역을 저장한 chat_log, 해당 채팅방의 과목명을 저장한 department name, 그리고 엔터티의 primary key인 chat_id로 이루어진다.

7.3. Relational Schema



[Figure 27] Relational Schema

7.4. SQL DDL

7.4.1. User

```
CREATE TABLE User
(
    user_id INT NOT NULL,
    password VARCHAR NOT NULL,
    image BLOB NOT NULL,
    name VARCHAR NOT NULL,
    department VARCHAR NOT NULL,
    study_time TIME,
    study_id INT,
    student_id INT NOT NULL,
    PRIMARY KEY(user_id)
);
```

[Figure 28] User Table DDL

7.4.2. Study Room

```
CREATE TABLE Study_Room
(
    study_id INT NOT NULL,
    study_name VARCHAR NOT NULL,
    participant_id INT NOT NULL,
    leader_id INT NOT NULL,
    department VARCHAR NOT NULL,
    study_time TIME,
    FOREIGN KEY (leader_id) REFERENCE User (user_id),
    PRIMARY KEY (study_id)
);
```

[Figure 29] Study Room Table DDL

7.4.3. Open Chat

```
CREATE TABLE Open_Chat
(
    user_id INT NOT NULL,
    department_list VARCHAR NOT NULL,
    FOREIGN KEY (user_id) REFERENCE User (user_id),
)
```

[Figure 30] Open Chat Table DDL

7.4.4. Chat Room

```
CREATE TABLE Chat_Room
(
    chat_id INT NOT NULL,
    chat_log VARCHAR NOT NULL,
    department_name VARCHAR NOT NULL,
    user_id INT NOT NULL,
    FOREIGN KEY (user_id) REFERENCE User (user_id),
    PRIMARY KEY (chat_id)
);
```

[Figure 31] Chat Room Table DDL

8. Testing Plan

8.1. Objectives

이 장은 테스트 계획을 development testing, release testing, user testing으로 나누어 설명한다.

8.2. Testing Policy

8.2.1. Development Testing

이 단계는 주로 소프트웨어 개발에서 발생할 수 있는 위험 요소를 줄이고 시간과 비용을 절약하기 위해 수행된다. 이 단계에서는 소프트웨어가 충분한 테스트를 거치지 않았기 때문에 불안정할 수 있으며 구성요소가 충돌할 수 있으므로 코드 분석 및 검토, data flow 분석, unit testing을 수행한다. 이 testing을 수행함으로써 소프트웨어의 성능과 신뢰성을 향상시키는 것을 목적으로 한다

8.2.1.1. Performance

이 시스템에서 스터디 룸을 생성하고 검색하는 서비스는 빠른 응답 시간을 가져야 하며 각 목록은 5초마다 업데이트 되어야 한다. 또한 사용자 간의 채팅 시스템은 즉시 메시지를 주고받을 수 있어야 한다. 이것을 시험할 수 있는 다양한 테스트 케이스를 준비하여 응답 속도를 평가하고 좋은 성능이 나올 수 있도록 코드를 분석하고 업데이트한다.

8.2.1.2. Reliability

시스템의 reliability를 높이기 위해서 unit testing을 수행하고 각 unit이 통합되는 동안 시스템에 오류가 발생하는지 반복적으로 확인한다.

8.2.1.3. Security

시스템은 사용자의 정보를 보호해야 한다. 타사에서 제공하는 모바일 애플리케이션 보안 테스트 서비스와 코드 분석을 통해 보안을 검토한다.

8.2.2. Release Testing

One 이 단계에서는 소프트웨어의 새로운 버전을 출시하기 전에 미리 테스트하고 결함 여부를 확인한다. Alpha 버전부터 테스트를 진행하여 시스템 오류를 수정하고 사용자를 포함한 테스트 진행을 위해 Beta 버전을 출시한다. Beta 버전에서는 사용자에게 usability, 디자인, 성능과 관한 의견을 얻는다.

8.2.3. Testing Case

신뢰성, 성능, 보안 3부분으로 나누어 각 부분마다 5개의 테스트케이스를 생성하고 시스템에 대한 테스트를 진행하고 평가한다..

9. Development Plan

9.1. Objectives

이 섹션에서는 서비스 개발 환경과 기술에 대한 소개를 한다. 사용할 기술 스택과 개발 환경에 대해 설명한다.

9.2. Frontend Environment

9.2.1. Figma

[Figure 32] Figma logo

Figma 를 사용해서 어플리케이션의 UI/UX 를 디자인하고 각 요소의 위치, 크기 등을 정의한다. 동일한 페이지를 팀원과 함께 공유할 수 있기 때문에 협업에 적합하다.

9.2.2. GitHub

[Figure 33] GitHub logo

깃 허브를 사용하여 개발하는 서비스의 버전관리, 형상관리 뿐 아니라 여러 라이브러리를 참고하여 사용한다. 버튼이나 디자인 등을 깃 허브에 있는 라이브러리를 사용하고자 한다.

9.2.3. Android Studio

[Figure 34] Android Studio logo

안드로이드 스튜디오를 사용하여 어플리케이션을 개발한다. 프론트엔드 개발에서 전체적인 어플리케이션의 구조와 디자인을 안드로이드 스튜디오를 사용하여 개발하고, 에뮬레이터를 이용해서 어플을 테스트한다.

9.3. Backend Environment

9.3.1. Firebase

[Figure 35] Firebase logo

firebase 는 구글에서 서비스하는 모바일 앱 개발에 적합한 기능을 제공하는 백엔드 플랫폼으로 데이터베이스 서버를 구축할 필요가 없다는 장점이 있다. 따라서 firebase 를 사용하여 데이터베이스를 관리하도록 한다.

9.3.2. Android Studio

[Figure 36] Android Studio logo

프론트엔드 부분과 백엔드 부분의 연결을 담당하는 것 또한 안드로이드 스튜디오를 사용한다. 어플리케이션의 데이터베이스 서버와 연결을 하고, 데이터를 관리하기 위해 안드로이드 스튜디오를 사용한다.

9.4. Constraints

우리가 개발하고자 하는 어플리케이션의 설계와 구현은 해당 디자인 명세서에 작성한 내용을 토대로 한다.

-
- 변경되거나 수정된 부분에 대해서는 각 팀원들과 형상 관리 툴을 사용해 소통한다.
- 개발을 진행하는 윈도우 환경은 Windows 10 환경이다.
- 최소 안드로이드 버전은 6.0(API 23)이고, 타겟 버전은 29이다.
- 에뮬레이터는 Android 버전 10(API 29) 기기를 사용한다.

9.5. Assumptions and Dependencies

이 문서에 작성된 설계와 구현에 관한 내용은 모두 안드로이드 환경에서 진행된다. 안드로이드는 버전 6.0, API 23을 기준으로 하고, 다른 환경이나 버전에서는 적용되지 않을 수 있다.

10. Supporting Information

10.1. Software Design Specification

해당 소프트웨어 디자인 명세서는 'IEEE Recommended Practice for Software Design Description'을 참고하여 작성했다.

10.2. Document History

[Table 16] Document History

Date	Version	Description	Writer
2021/5/14	0.1	Section 1, 2, 6, 8 추가	박윤진
2021/5/14	0.2	Section 3, 4 추가	김지수
2021/5/14	0.3	Section 3, 5 추가	장영재
2021/5/14	0.4	Section 7, 9, 10 추가	박세연
2021/5/16	1.0	전체 문서 취합, 정리	