



Project Highlight

Highlighting SKKU online campus life

2021.05.16

소프트웨어공학개론 41

Team 1

2017314510 안이은

2016311900 이성원

2017312115 양유진

2018310004 정세린

2018312179 박정인

2019310219 김지훈

목차

1. Preface.....	- 8 -
1.1. 독자들을 위하여···	- 8 -
1.2. 본 문서의 Scope(범주).....	- 8 -
1.3. 본 문서의 Objectives(목적).....	- 8 -
1.4. 본 문서의 Structures(구성).....	- 8 -
2. Introduction.....	- 9 -
2.1. 이 챕터에서는···	- 9 -
2.2. Diagrams.....	- 9 -
2.2.1. UML.....	- 9 -
2.2.2. Use Case Diagrams.....	- 10 -
2.2.3. Sequence Diagrams.....	- 10 -
2.2.4. Class Diagrams.....	- 10 -
2.2.5. Context Diagrams.....	- 10 -
2.2.6. Entity Relationship Diagrams.....	- 11 -
2.3. Applied Tools.....	- 11 -
2.3.1. Microsoft Word.....	- 11 -
2.3.2. Microsoft PowerPoint	- 11 -
2.3.3. draw.io.....	- 11 -
2.4. 프로젝트 Scope(범주).....	- 11 -
2.5. References.....	- 12 -
3. System Architecture – Overall.....	- 12 -
3.1. 이 챕터에서는···	- 12 -
3.2. 시스템 구성.....	- 12 -
3.2.1. Context Diagrams.....	- 13 -
3.2.2. Sequence Diagrams.....	- 13 -
3.2.3. Use Case Diagrams.....	- 17 -
4. System Architecture – Frontend.....	- 17 -
4.1. Objectives.....	- 17 -
4.2. Subcomponents.....	- 18 -

4.2.1. 로그인 및 회원가입.....	- 18 -
4.2.2. Attributes.....	- 18 -
4.2.3. Methods.....	- 18 -
4.2.4. 맞춤형 알림.....	- 20 -
4.2.5. 게시물 업로드 및 수정, 삭제.....	- 23 -
4.2.6. 게시물 열람 및 댓글 작성.....	- 26 -
4.2.7. 실시간 공유 문서 생성 및 참여.....	- 29 -
4.2.8. 강의 클립 생성 및 업로드.....	- 31 -
4.2.9. 강의 클립 열람.....	- 34 -
4.2.10. 강의 노트 공유.....	- 37 -
4.2.11. eBook.....	- 39 -
4.2.12. 채팅.....	- 41 -
5. System Architecture – Backend	- 44 -
5.1. Objectives.....	- 44 -
5.2. Overall Architecture.....	- 44 -
5.2.1. System.....	- 45 -
5.2.2. Handler.....	- 45 -
5.2.3. Controller.....	- 45 -
5.3. Subcomponents.....	- 45 -
5.3.1. Customized Alert System.....	- 45 -
5.3.2. Post System.....	- 46 -
5.3.3. Lecture System.....	- 47 -
5.3.4. Lecture Note System.....	- 48 -
5.3.5. Lecture Clip System.....	- 49 -
5.3.6. eBook System.....	- 50 -
5.3.7. Chat System.....	- 51 -
5.3.8. Real-time Document System.....	- 52 -
6. Protocol Design.....	- 54 -
6.1. Objective.....	- 54 -
6.2. 전달 형식.....	- 54 -
6.2.1. HTTP.....	- 54 -

6.2.2. MQTT	- 54 -
6.3. Authentication.....	- 54 -
6.3.1. 로그인 및 회원가입.....	- 54 -
6.3.2. 개인정보 수정.....	- 56 -
6.3.3. 맞춤형 알림.....	- 57 -
6.3.4. 게시글 업로드 및 열람.....	- 59 -
6.3.5. 강의 컨텐츠 업로드 및 열람.....	- 64 -
6.3.6. 실시간 공유 문서.....	- 65 -
6.3.7. 강의 클립화.....	- 68 -
6.3.8. 강의 노트 공유.....	- 70 -
6.3.9. eBook	- 71 -
6.3.10. 채팅.....	- 72 -
6.3.11. 권한부여.....	- 75 -
7. Database design.....	- 76 -
7.1. Objective.....	- 76 -
7.2. E-R diagram.....	- 76 -
7.2.1. Entities.....	- 77 -
7.3. Relational Schema.....	- 79 -
7.4. SQL DDL.....	- 80 -
7.4.1. User.....	- 80 -
7.4.2. Lecture	- 80 -
7.4.3. Lecture sign up.....	- 80 -
7.4.4. eBook	- 81 -
7.4.5. eBook subscribe.....	- 81 -
7.4.6. Lecture note	- 81 -
7.4.7. Lecture note star.....	- 82 -
8. Testing Plan.....	- 82 -
8.1. 이 챕터에서는.....	- 82 -
8.2. Testing Policy	- 82 -
8.2.1. Development Test.....	- 82 -
8.2.2. Release Test.....	- 84 -

8.2.3. User Test.....	- 84 -
8.2.4. Test Case.....	- 85 -
9. Development Plan.....	- 85 -
9.1. 이 챕터에서는···	- 85 -
9.2. Frontend Environment.....	- 85 -
9.2.1. Adobe Illustrator.....	- 85 -
9.2.2. Adobe Xd.....	- 86 -
9.2.3. Flutter	- 86 -
9.3. Backend Environment.....	- 86 -
9.3.1. Github.....	- 86 -
9.3.2. Jenkins.....	- 87 -
9.3.3. Firebase.....	- 87 -
9.3.4. Mosquitto.....	- 87 -
9.3.5. FFMPEG.....	- 88 -
9.4. Constraints.....	- 88 -
9.5. Assumption and Dependencies.....	- 89 -
10. 추가 정보.....	- 89 -
10.1. 서식.....	- 89 -
10.2. 문서 시간표.....	- 89 -

그림목차

그림 1 OVERALL SYSTEM ARCHITECTURE.....	- 12 -
그림 2 CONTEXT DIAGRAM.....	- 13 -
그림 3 가입/로그인 SEQUENCE DIAGRAM.....	- 13 -
그림 4 맞춤형 알림 SEQUENCE DIAGRAM.....	- 14 -
그림 5 강의 클립화 SEQUENCE DIAGRAM.....	- 14 -
그림 6 강의 노트 공유 SEQUENCE DIAGRAM.....	- 15 -
그림 7 EBOOK DIAGRAM.....	- 15 -
그림 8 실시간 공유 문서 SEQUENCE DIAGRAM.....	- 16 -
그림 9 USE CASE DIAGRAM.....	- 17 -
그림 10 CLASS DIAGRAM – 로그인 및 회원가입.....	- 19 -
그림 11 SEQUENCE DIAGRAM – 로그인 및 회원가입.....	- 20 -
그림 12 CLASS DIAGRAM – 맞춤형 알림.....	- 21 -
그림 13 SEQUENCE DIAGRAM – 맞춤형 알림.....	- 22 -
그림 14 CLASS DIAGRAM – 게시글 업로드 및 수정, 삭제.....	- 24 -
그림 15 SEQUENCE DIAGRAM – 게시글 업로드 및 수정, 삭제.....	- 25 -
그림 16 CLASS DIAGRAM – 게시글 열람 및 댓글 작성.....	- 27 -
그림 17 SEQUENCE DIAGRAM – 게시글 열람 및 댓글 작성.....	- 28 -
그림 18 CLASS DIAGRAM – 실시간 공유 문서 생성 및 참여.....	- 29 -
그림 19 SEQUENCE DIAGRAM – 실시간 공유 문서 생성 및 참여.....	- 30 -
그림 20 CLASS DIAGRAM – 강의 클립 생성 및 업로드.....	- 32 -
그림 21 SEQUENCE DIAGRAM – 강의 클립 생성 및 업로드.....	- 33 -
그림 22 CLASS DIAGRAM – 강의 클립 열람.....	- 35 -
그림 23 SEQUENCE DIAGRAM – 강의 클립 열람.....	- 36 -
그림 24 CLASS DIAGRAM – 강의 노트 공유.....	- 38 -
그림 25 SEQUENCE DIAGRAM – 강의 노트 공유.....	- 39 -
그림 26 CLASS DIAGRAM – EBOOK.....	- 40 -
그림 27 SEQUENCE DIAGRAM – EBOOK.....	- 40 -
그림 28 CLASS DIAGRAM – 채팅.....	- 42 -
그림 29 SEQUENCE DIAGRAM – 채팅.....	- 43 -
그림 30 OVERALL ARCHITECTURE.....	- 44 -
그림 31 CLASS DIAGRAM – CUSTOMIZED ALERT SYSTEM.....	- 45 -
그림 32 SEQUENCE DIAGRAM – CUSTOMIZED ALERT SYSTEM.....	- 46 -
그림 33 CLASS DIAGRAM – POST SYSTEM.....	- 46 -
그림 34 SEQUENCE DIAGRAM – POST SYSTEM.....	- 47 -
그림 35 CLASS DIAGRAM – LECTURE SYSTEM.....	- 47 -
그림 36 SEQUENCE DIAGRAM – LECTURE SYSTEM.....	- 48 -
그림 37 CLASS DIAGRAM – LECTURE NOTE SYSTEM.....	- 48 -
그림 38 SEQUENCE DIAGRAM – LECTURE NOTE SYSTEM.....	- 49 -
그림 39 CLASS DIAGRAM – LECTURE CLIP SYSTEM.....	- 49 -
그림 40 SEQUENCE DIAGRAM – LECTURE CLIP SYSTEM.....	- 50 -
그림 41 CLASS DIAGRAM – EBOOK SYSTEM.....	- 50 -

그림 42 SEQUENCE DIAGRAM – EBOOK SYSTEM.....	- 51 -
그림 43 CLASS DIAGRAM – CHAT SYSTEM.....	- 51 -
그림 44 SEQUENCE DIAGRAM – CHAT SYSTEM.....	- 52 -
그림 45 CLASS DIAGRAM – REAL-TIME DOCUMENT SYSTEM.....	- 52 -
그림 46 SEQUENCE DIAGRAM – REAL-TIME DOCUMENT SYSTEM.....	- 53 -
그림 47 OBJECT ORIENTED FUCNTION.....	- 53 -
그림 47 E-R 다이어그램.....	- 76 -
그림 48 USER ENTITY.....	- 77 -
그림 49 LECTURE ENTITY.....	- 77 -
그림 50 EBOOK ENTITY.....	- 78 -
그림 51 LECTURE NOTE ENTITY.....	- 78 -
그림 52 RELATIONAL SCHEMA.....	- 79 -
그림 53 소프트웨어 배포 생명 주기.....	- 84 -
그림 54 ADOBE ILLUSTRATOR 로고.....	- 85 -
그림 55 ADOBE XD 로고.....	- 86 -
그림 56 FLUTTER 로고.....	- 86 -
그림 57 GITHUB 로고.....	- 86 -
그림 58 JENKINS 로고.....	- 87 -
그림 59 FIREBASE 로고.....	- 87 -
그림 60 MOSQUITTO 로고.....	- 87 -
그림 61 FFmpeg 로고.....	- 88 -

1. Preface

이 챕터에서는 본 문서의 독자들을 위한 간략한 정보들, 개요, 본 문서의 범주, 목적을 기술하고 있으며, Project Highlight 를 위한 소프트웨어 디자인 관련 구성물을 소개한다.

1.1. 독자들을 위하여...

본 디자인 명세서는 Highlight 의 디자인 설계 요소에 따라 챕터가 구분되어있다. 소프트웨어 디자인 명세서의 구조는 본 단락 아래의 [1.4. 본 문서의 Structure]에 기술된 바에 따라 찾을 수 있다. 본 문서는 2021 1 학기 소프트웨어 공학 개론 Team 1 을 주요 독자로 삼고 있으며, 추가적으로 해당 수업의 교수, 조교 및 여타 수강생들 또한 주요 독자가 될 수 있다.

1.2. 본 문서의 Scope(범주)

본 디자인 명세서는 성균관대학교의 차세대 쌍방향 자기주도형 온라인 강의 플랫폼인 Highlight 의 소프트웨어 공학(Software Engineering)과 소프트웨어 품질 공학(Software Quality Engineering)에 쓰일 예정이다.

1.3. 본 문서의 Objectives(목적)

본 디자인 명세서의 주요 목적은 강의 플랫폼 Highlight 의 개발에 필요한 디자인 요소를 묘사하는 데에 있다. 본 문서에는 Highlight 를 직접 개발하는 데에 필요한 소프트웨어 아키텍처(Software Architecture)와 소프트웨어 디자인 결정 요소(Software design decision)가 묘사되어 있다. 또한 본 문서에는 Highlight 시스템을 여러 다른 관점에서 묘사하기 위한 아키텍처에 대한 전체적 조망이 담겨있다. 본 문서를 통해 Team1 은 이전에 작성된 요구사항 명세서에서 논의된 여러 모듈 들의 디자인과 구조를 더욱 명확히 하고, 몇몇 use case 들을 실제 개발팀이 특정 모듈을 구현하는데에 필요한 class diagram 을 포함한 activity diagram, sequential diagram 을 통해 보여준다. 본 문서가 주요 대상으로 삼는 독자층은 본 시스템의 이해관계자(성균관대학교 학사운영팀 측, 교수진, 학생 대표), 시스템 개발자, 시스템 디자이너, 소프트웨어 테스터이며, 경우에 따라 더욱 늘어날 수 있다.

1.4. 본 문서의 Structures(구성)

1. Preface: 이 챕터에는 본 문서의 독자들을 위한 정보, 본 문서의 적용 범주, Highlight 의 목적, 본 문서의 구조가 기술되어 있다
2. Introduction: 본격적으로 본 디자인 명세서에 기술될 정보들의 특성이 설명되고, 본 프로젝트가 목표로 삼고 있는 바에 대해 기술한다.
3. System Architecture – Overall: Highlight 의 개괄적인 아키텍처를 use case, sequence, context diagram 을 통해 기술한다.

4. System Architecture – Frontend: Highlight 의 프론트엔드 파트에 대해 기술한다.
5. System Architecture – Backend: Highlight 의 백엔드 파트에 대해 기술한다.
6. Protocol Design: Highlight 가 정보를 통신하는 프로토콜에 대해 기술한다.
7. Database Design: Highlight 의 데이터베이스 디자인에 대해 기술한다.
8. Testing Plan: Highlight 의 테스트 계획에 대해 기술한다.
9. Developing Plan: Highlight 의 개발에 사용된 툴, Highlight 에 대한 가정들과 의존성 및 제약사항에 대해 기술한다.
10. Supporting Information: 본 문서의 수정기록을 기록한다.

2. Introduction

Highlight 는 Team1 에서 기획한 모바일 어플리케이션이자 자기주도형 쌍방향 강의 플랫폼으로 성균관대학교 학생 및 교수진이 강의 전, 중, 후로 더욱 학습에 집중할 수 있는 환경을 제공하는 데에 그 목적이 있다. Highlight 는 교수로 하여금 외부의 플랫폼을 사용하지 않고 실시간 강의를 진행할 수 있도록 해주며, 학생들에게는 이전처럼 일방적인 녹화 강의의 시청이 아닌, 실시간으로 질의응답을 할 수 있도록 해준다. 또한 교수로 하여금 학생의 학업 성취도에 대한 평가 및 강의 관리를 더욱 수월하게 해주고, 학생들에게는 자기주도적으로 온라인 미팅을 진행, 여러 알림에 대한 개인화를 통해 강의시간 외에도 바람직한 학습환경이 만들어질 수 있도록 한다. 본 디자인 명세서에는 Highlight 프로젝트에 직/간접적으로 사용되는 여러 디자인 구조가 묘사되어 있다. 묘사된 디자인 구조는 이전에 작성된 요구사항 명세서에서 명시된 요구사항들에 따라 설계되어 있다.

2.1. 이 챕터에서는...

이 챕터에는 본 프로젝트의 디자인 단계에서 적용한 여러가지 툴 및 다이어그램들이 설명되어 있다.

2.2. Diagrams

2.2.1. UML

UML 은 통합 모델링 언어(Unified Modeling Language)의 약자로, 소프트웨어를 모델링 하는데 표준으로 사용되는 일종의 약속 체계를 의미한다. UML 을 통해 시스템의 디자인에 있어서 혼선 없이 개발팀 전체가 같은 이해를 갖고 개발에 이해할 수 있으며, 개발 관점 이전의 비즈니스 관점에서도 시스템의 진행 과정에 대해 어느정도 이해할 수 있도록 돋는다. 또한 UML 은 설계된 디자인, 비즈니스 프로세스, 구조 및 구성 요소 간의 역학을 문서화하는 데에 굉장히 유리한 언어이다. UML 은 유통, 금융, 제약, 건축 등 여러 산업 분야에서 그들의 비즈니스 프로세스를 표현하는데 이용될 수 있으며,

2.2.2. Use Case Diagrams

Use Case Diagram 은 본 어플리케이션의 주요 사용자 층인 교수, 조교, 학생의 요구사항에 따른 사용 예시에 대한 Diagram 이다. 시스템의 설계에 있어 요구사항이 가장 기초에 있듯, Use Case Diagram 은 시스템의 존재 목적에 대한 요약이라고 볼 수 있다. Use Case Diagram 에 있어서 우리가 중요하게 보는 요소는 Actor, Scope 와 그에 속하는 Use Case 이다. Actor 는 시스템을 실제로 사용하는 소비자로서, 본 어플리케이션의 경우에는 교수, 학생, 조교, 관리자가 이에 해당할 것이다. Scope 는 우리가 이 프로그램의 주요 기능으로 여기는 갈래들이며, Scope 에 속한 Use Case 는 실제로 사용자가 기능을 사용할 때 마주하는 상황들을 의미한다. 이를테면 eBook 을 열람하는 기능에는 세부적으로 업로더가 eBook 을 업로드하는 Use Case, 학생인 사용자가 eBook 을 구독 및 열람하는 Use Case 가 있을 수 있다.

2.2.3. Sequence Diagrams

Sequence Diagram 은 어떤 작업의 과정을 나타내는 UML Diagram 으로 시간에 따른 순차적인 진행 과정을 보이는 모든 산업에서 중요하게 쓰이는 Diagram 이다. Sequence Diagram 을 통해 어느 작업을 나타냄에 있어 글로 장황하게 나열하거나, 일관성 없는 그림이나 표로 같은 작업이 다른 형태로 나타나는 등의 혼선 없이 UML 을 이용하여 표준화된 작업 과정을 보여줄 수 있다. 우리에게 있어 Sequence Diagram 은 Highlight 가 갖는 여러가지 Scope 들과 그들이 갖는 Use Case 가 실제로 사용자, 기기, 서버에서 어떤 상호작용을 통해 작동하는지 보여주는 주요한 도구가 될 것이다.

2.2.4. Class Diagrams

Class Diagram 은 프로그램의 아키텍처를 더욱 직접적으로 보여주는 Diagram 이다. Class Diagram 에서는 시스템을 구성하는 객체와, 그 객체의 attribute, method, 그리고 객체 간의 관계를 보여준다. 프로그램은 컴포넌트로 이루어져 있고, 현대의 소프트웨어는 객체 지향 프로그래밍으로 인해 컴포넌트는 대체로 객체로 이루어져 있다. 즉, Class Diagram 은 시스템의 조합을 보여주는 Diagram 이라고 할 수 있다. 이 Diagram 에서 하나의 Class 는 3 개의 행과 1 개의 열을 갖는 사각형으로 묘사된다. 맨 윗줄에는 Class 의 이름이, 가운데 줄에는 Class 의 attribute, 맨 아랫줄에는 Class 의 method 가 표시된다.

2.2.5. Context Diagrams

Context Diagram 은 시스템의 데이터 플로우를 표현하는 가장 높은 레벨의 Diagram 이다. 이 Diagram 에선 전체 시스템을 대표하고 시스템의 맥락과 범위를 만드는 가장 큰 process 만이 표현된다. 즉, 시스템과 사용자 간의 데이터의 큰 흐름만 표현하는 것이다. Context Diagram 은 보통 요구사항 명세서에 기술된다. 이 Diagram 은 반드시 모든 이해관계자에게 읽혀야 하며, 그렇기 때문에 전문 용어가 아닌, 누구라도 이해할 수 있는 용어로 기술되어야 한다. 왜냐하면, 모든 이해관계자가 시스템의 역학구조를 알아야, 그 과정에서 생기는 제약사항, 또는 요구사항을 충분히 알 수 있기 때문이다. 시스템 전체를 하나의 프로세스로 보여준다는 점이 다른 Diagram 과의 차이점이라고 할 수 있다. 그 모습은 정 가운데에 시스템을 표시하고, 주변을 둘러싼 사용자 및 그 외 영향을 미치는 요소들이 시스템과 교류하는 형태를 띤다.

2.2.6. Entity Relationship Diagrams

Entity Relationship diagram(이하 ER Diagram)은 데이터베이스의 관점에서 시스템을 분석하는 Diagram이다. 따라서 entity 와 그들의 relationship 을 주로 나타내게 된다. 하나의 Entity 에는 여러 attribute 가 있을 수 있고, 어떤 attribute 가 다른 entity 에 종속되거나, entity 자체가 상속될 수도 있고, 이러한 역학 관계들이 그림으로 표현되는 것이 ER diagram 이다. 작성된 ER Diagram 은 데이터베이스를 구축하는 데에 사용될 것이다.

2.3. Applied Tools

2.3.1. Microsoft Word

이 툴은 본 프로젝트의 요구사항 명세서, 디자인 명세서를 작성하는 데에 사용되었으며, 추후 테스트 계획서를 작성하는 데에도 이용될 수 있다. 팀원들 간의 서로 다른 문서 편집환경에서 제일 공통된 툴이고, 작성된 파일에 대해서도 가능한 한 많은 곳에서 편집이 가능하게 해주는 툴이므로 Team1은 MS Word 를 기본적으로 사용하기로 한다.

2.3.2. Microsoft PowerPoint

이 툴은 문서 내 UML 이외의 수단으로 작성된 그림을 만드는 데에 사용된 툴이다. MS Word 와 같은 포맷을 지원하기 때문에 완성된 그림에 대해 높은 호환성을 갖고 문서를 작성하는 데에 유리한 입지를 점할 수 있게 한다.

2.3.3. draw.io

draw.io 는 플로우 차트를 작성하는 툴로, 자체적으로 UML 을 통한 Diagram 작성은 지원하기 때문에 본 프로젝트에서 사용하는 수많은 Diagram 의 작성에 큰 도움이 된다. 또한 웹 브라우저에서도 작성은 지원하기 때문에 별도의 설치가 필요 없고 네트워크가 연결된 환경이라면 어디서든 Diagram 의 작성은 할 수 있게 한다.

2.4. 프로젝트 Scope(범주)

쌍방향 자기주도형 강의 플랫폼 Highlight 는 변화하는 학습 환경에 따른 온라인 수업의 확장에 대한 요구와 그에 대해 모자란 기존의 아이캠퍼스 어플리케이션에 대한 개선 요구에 의해 도입된 프로젝트이다. 기본적으로 본 프로젝트는 사용자와 사용자 간의 실시간 교류에 목적을 두고 있으므로, 이를 위한 데이터 중계 서버가 동시에 구비되어야 할 것이고, 자기주도형 학습 어플리케이션이라는 이름에 걸맞게 강의 컨텐츠 알림에 대한 개인화를 블록 프로그래밍을 통해 지원할 것이다. 이 외에도 가능한 한 사용자에게 친화적인 환경의 제공을 통해 사용자가 스스로 학습에 흥미를 갖고 임할 수 있도록 Team1 은 노력할 것이다.

2.5. References

- Team 1, 2020 Spring. Software Design Document, SKKU

3. System Architecture – Overall

3.1. 이 챕터에서는...

이 챕터에는 시스템의 프론트엔드부터 백엔드까지의 디자인 구성이 묘사되어 있다.

3.2. 시스템 구성

Highlight는 실시간 쌍방향 화상 강의 서비스 및 사용자 간 실시간 채팅과 문서 공유 서비스를 제공한다. 이는 Highlight가 라이브 서비스 어플리케이션의 성향을 강하게 뛴다는 것을 의미한다. 따라서, Highlight는 서버-클라이언트 모델 형식으로 구현을 할 필요가 있다. 이를 MVC 디자인 패턴을 통해 설명하면, 사용자가 프론트엔드에 해당하는 기기 내 View의 조작을 통해 Controller에 조절을 요청하고, Controller는 요청받은 작업에 대해 백엔드에 해당하는 Model의 데이터를 변경, 같은 내용을 View에 반영하여 사용자에게 보여준다. 이 과정에서 Controller와 View, Model이 데이터를 주고받는 형태는 JSON의 포맷으로 HTTP 프로토콜에 따라 진행된다. 이렇게 Model로부터 얻은 데이터는 View에서 UI에 따라 최대한 원본과 유사하게 표시되고, 이 과정에서 오픈소스 API를 통해 동영상, 텍스트, 사진, pdf 등이 각각에 맞게 표시된다. 또한 사용자는 자신의 기기에서 강의 노트, Controller를 통해 제작한 클립들을 Model에 업로드 요청할 수 있고, 같은 사용자 혹은 다른 사용자가 해당 데이터에 대한 접근을 요청할 시, 업로드 된 데이터가 View에 표시된다.

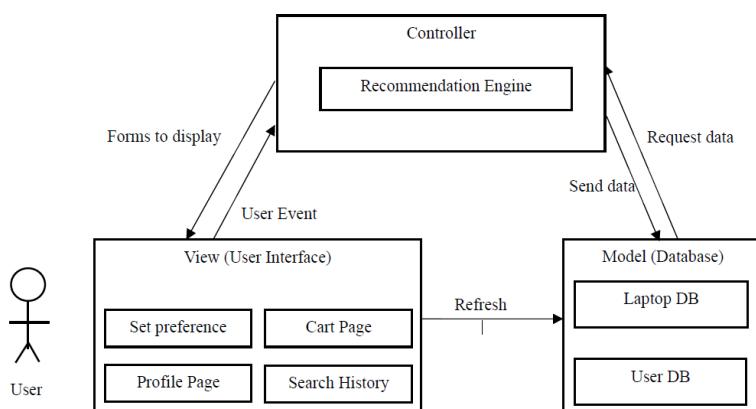


그림 1 Overall System Architecture

3.2.1. Context Diagrams

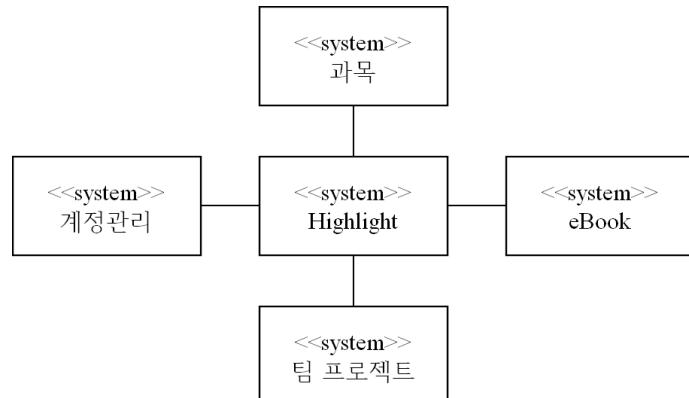


그림 2 Context Diagram

3.2.2. Sequence Diagrams

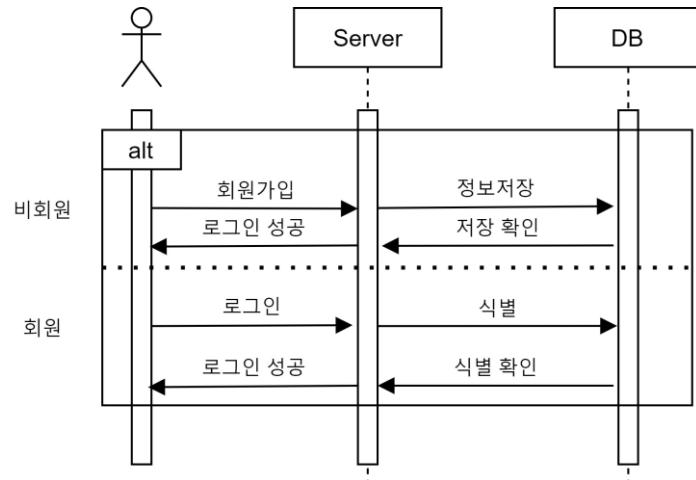


그림 3 가입/로그인 Sequence Diagram

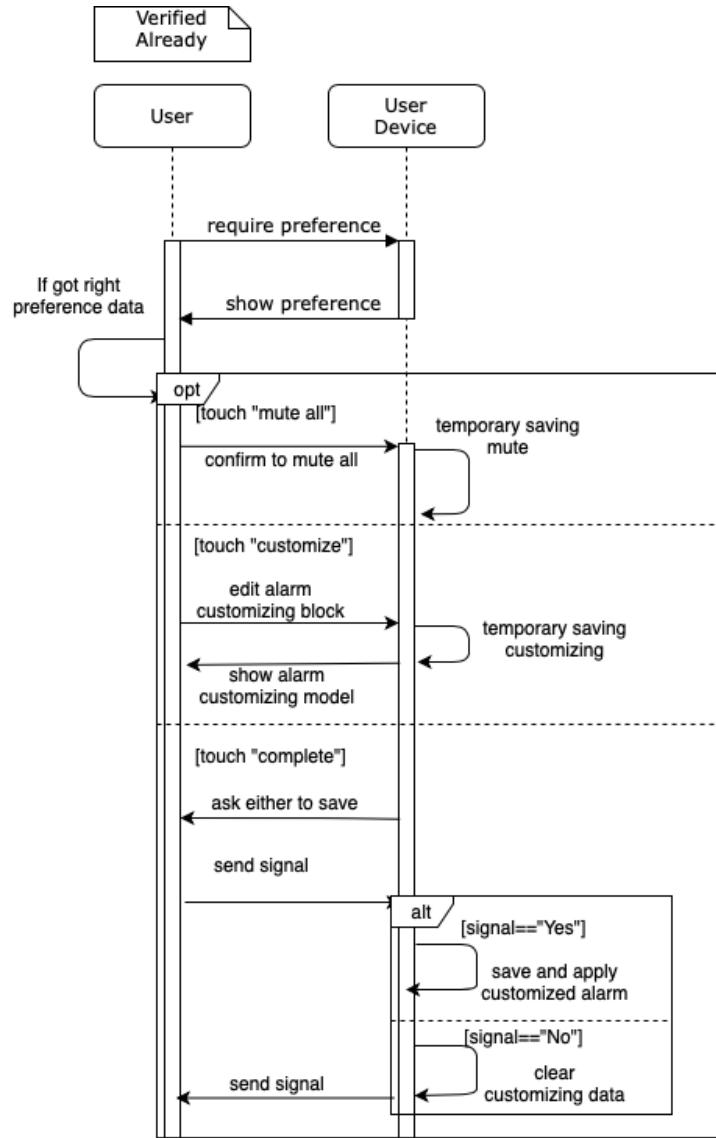


그림 4 맞춤형 알림 Sequence Diagram

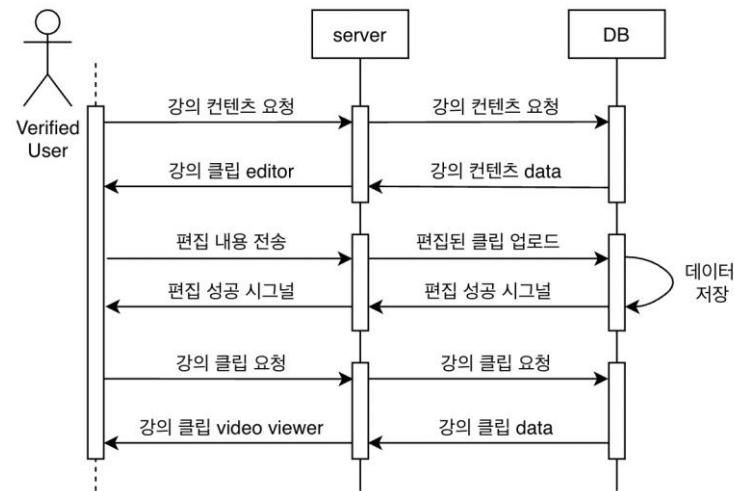


그림 5 강의 클립화 Sequence Diagram

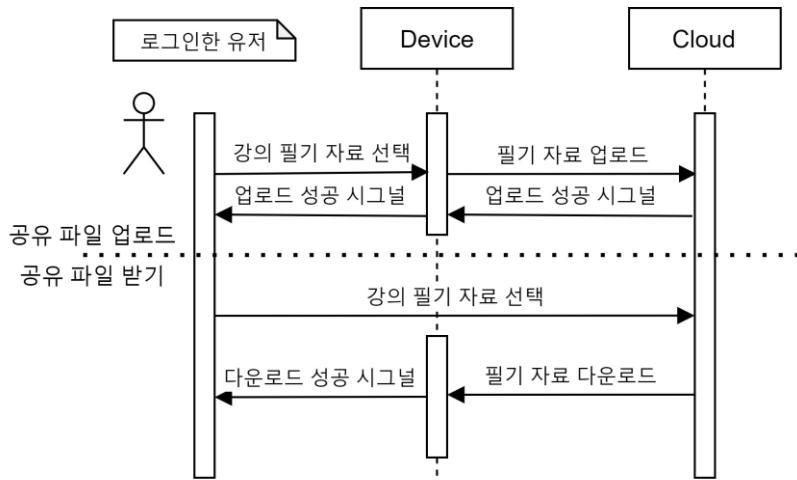


그림 6 강의 노트 공유 Sequence Diagram

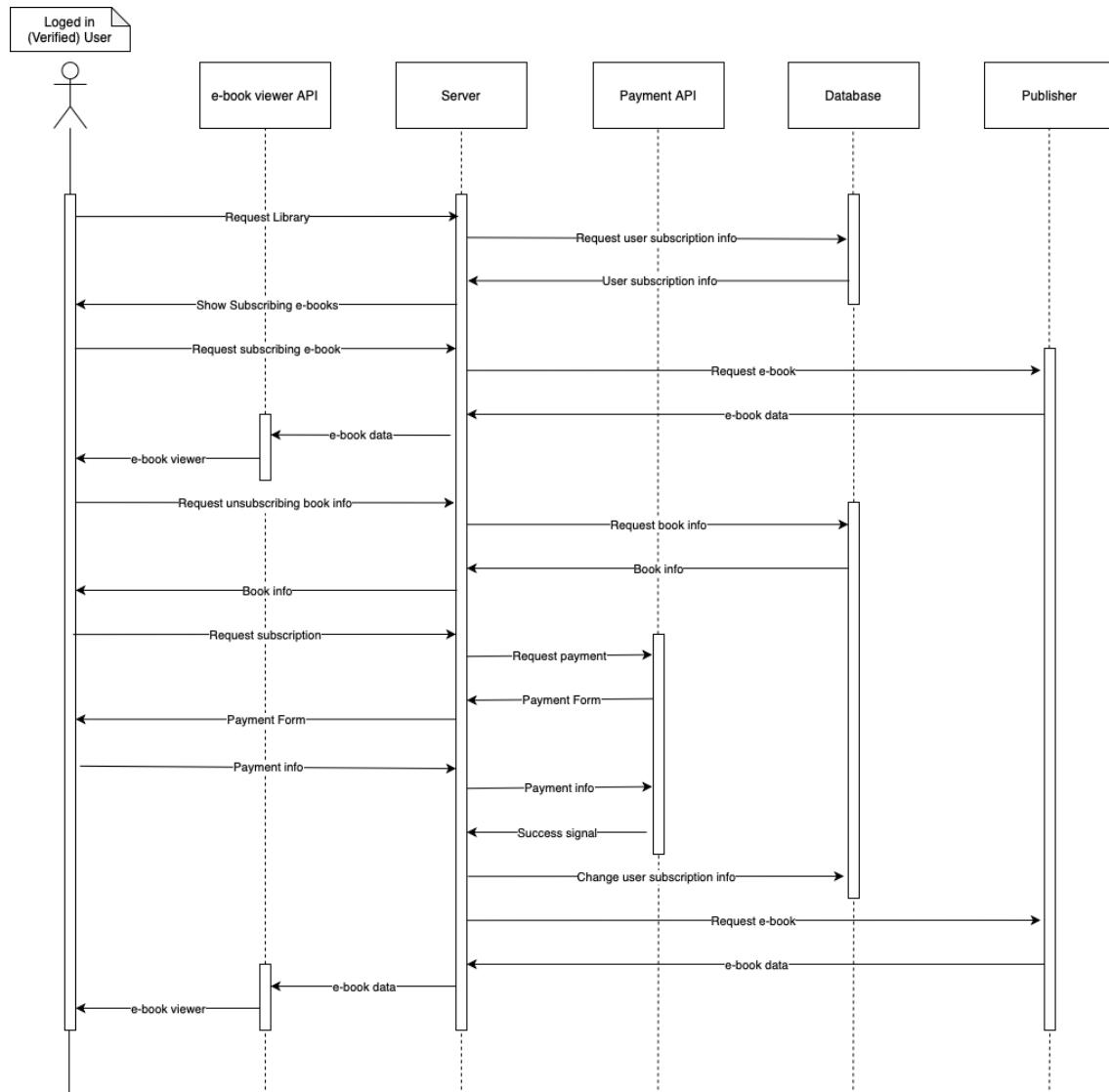


그림 7 eBook Diagram

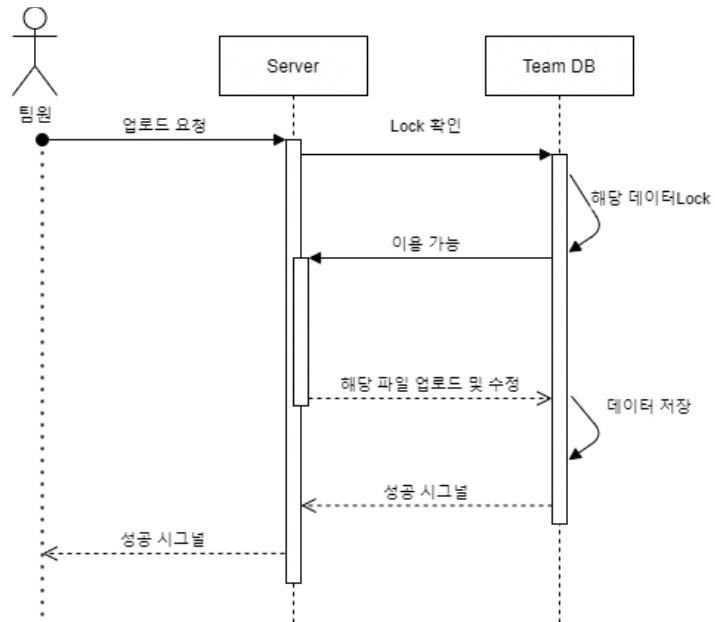
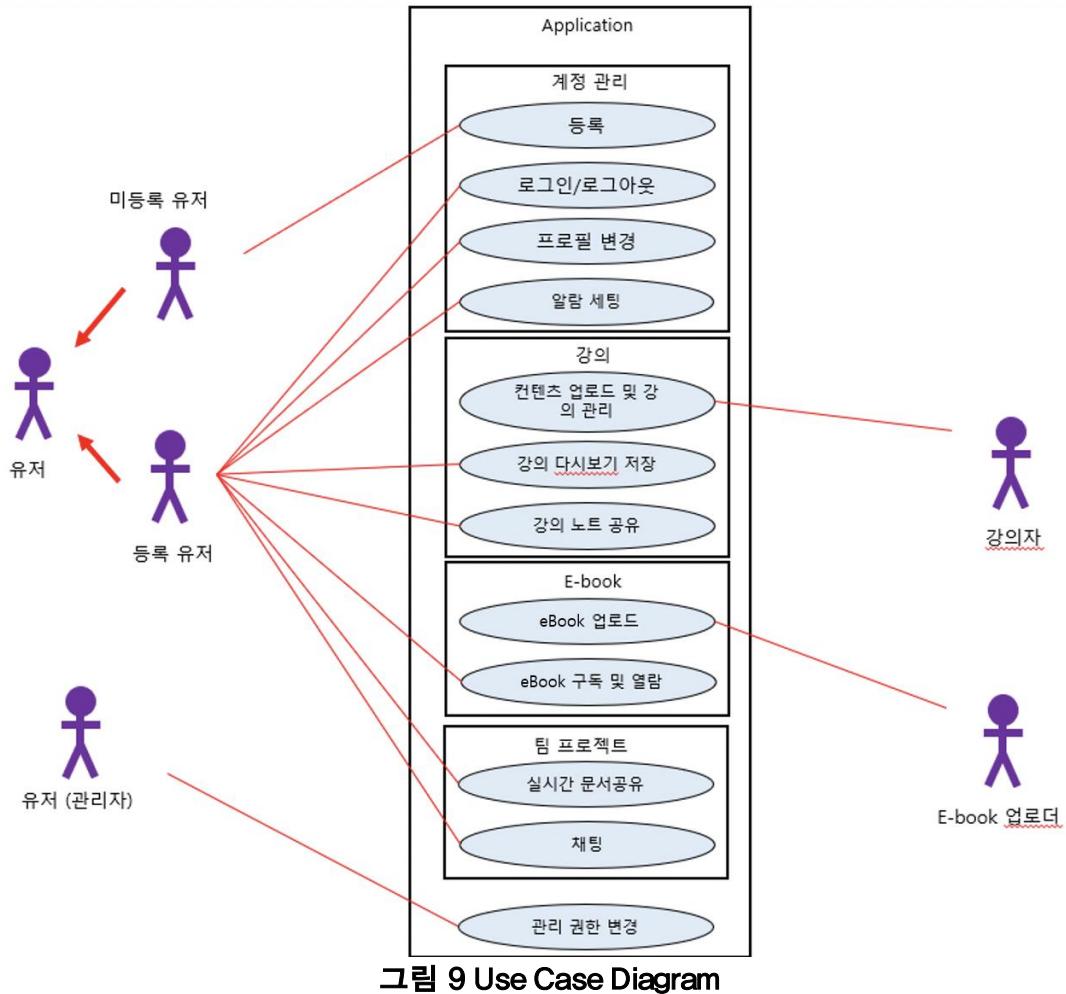


그림 8 실시간 공유 문서 Sequence Diagram

3.2.3. Use Case Diagrams



4. System Architecture – Frontend

4.1. Objectives

System Architecture에서 사용자 인터페이스에 해당하는 Frontend 시스템을 이루는 Component들의 구성을 Class Diagram과 Sequence Diagram으로 도식화 및 설명한다.

4.2. Subcomponents

4.2.1. 로그인 및 회원가입

4.2.2. Attributes

해당 profile object 가 가지는 attribute 는 다음과 같다.

- User id: 로그인시 사용되는 사용자의 id 이다.
- Name: 사용자의 본명이다.
- Student id: 사용자의 학번이다

4.2.3. Methods

해당 profile object 가 가지는 attribute 는 다음과 같다.

- SetProfile()
- GetProfile()

4.2.3.1. Class Diagram

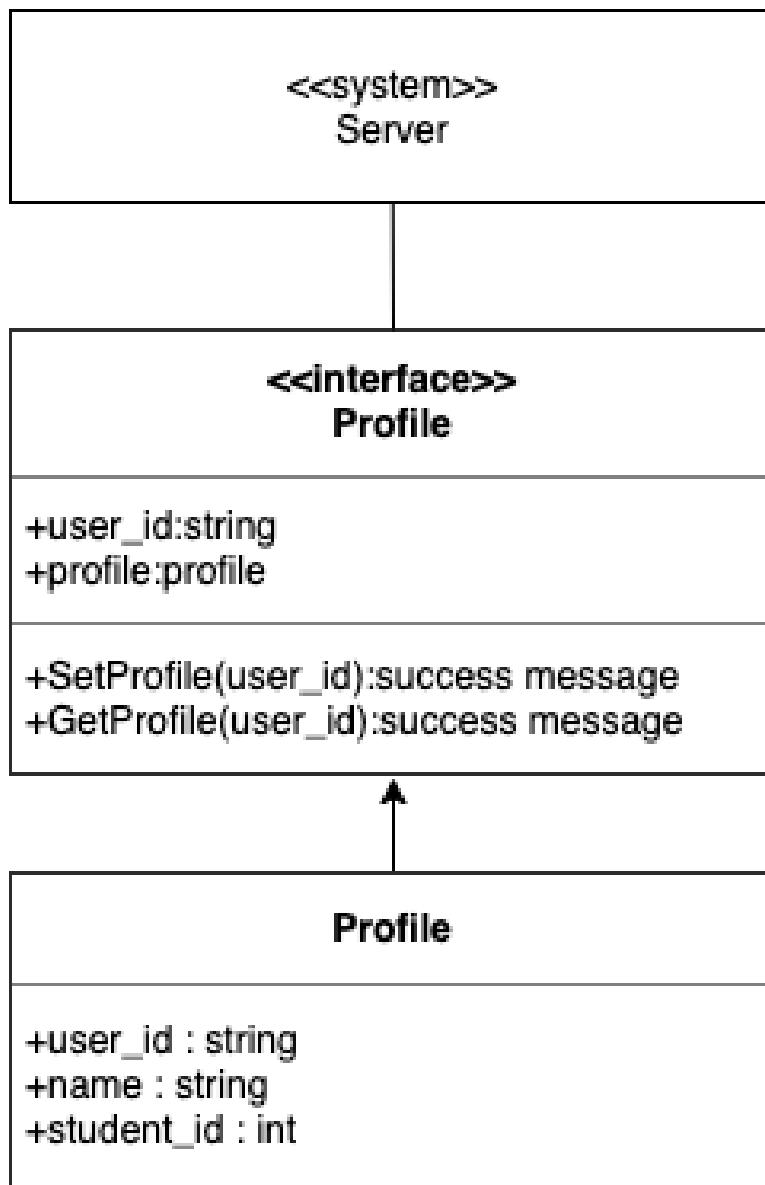


그림 10 Class Diagram – 로그인 및 회원가입

4.2.3.2. Sequence Diagram

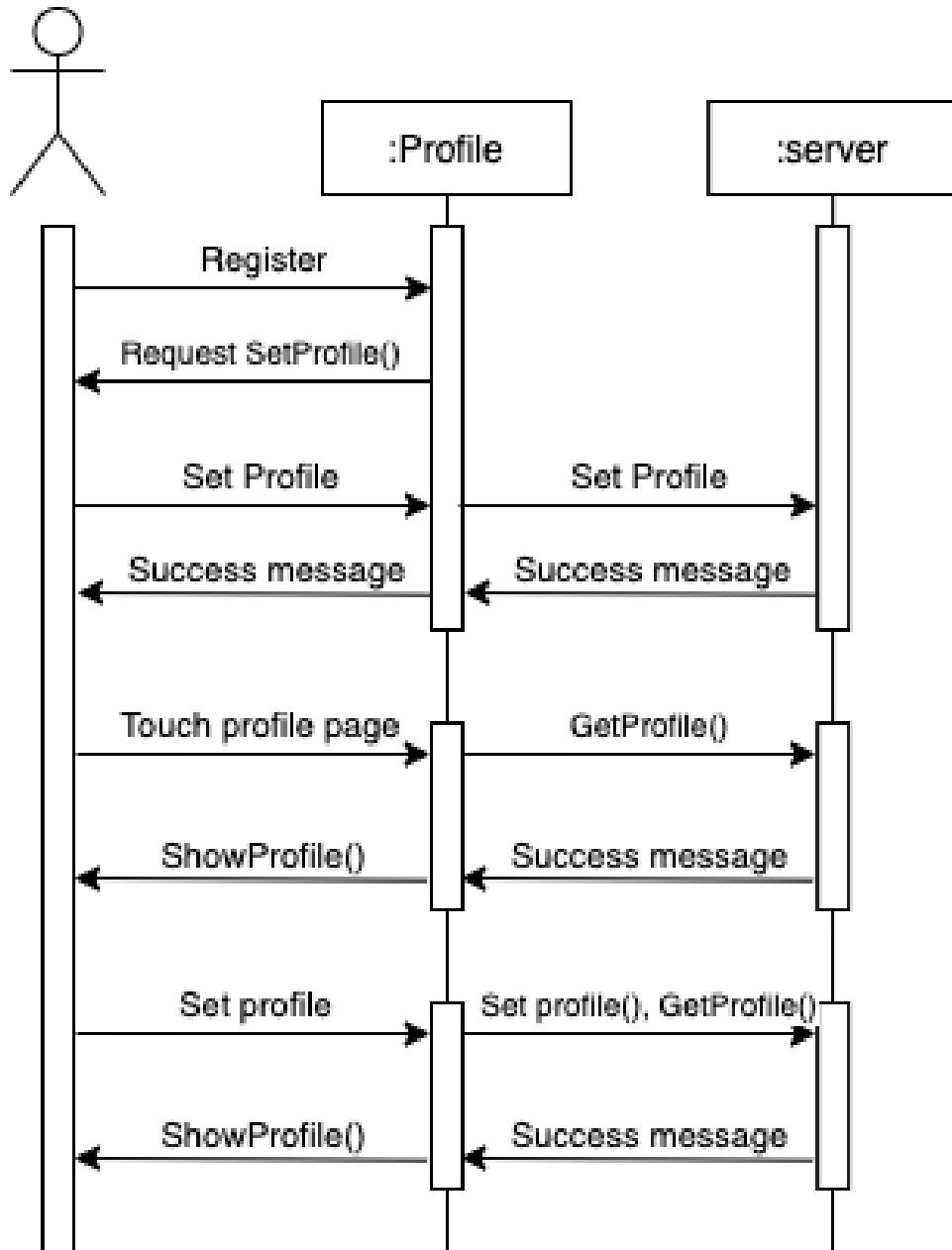


그림 11 Sequence diagram – 로그인 및 회원가입

4.2.4. 맞춤형 알림

4.2.4.1. Attributes

해당 customized alert object 가 가지는 attribute 는 다음과 같다.

- userId: 현재 사용자의 id 이다.
- alertId: 알림의 id 이다.
- alertList: 현재 사용자가 생성한 알림 목록이다.
- alertName: 사용자가 생성시에 설정한 알림의 이름 (별칭)이다.

- content: 생성한 알림을 xml 형태로 저장한 내용이다.

4.2.4.2. Methods

해당 customized alert object 가 가진 attribute 는 다음과 같다.

- uploadAlert()
- createAlert()
- editAlert()
- deleteAlert()
- getAlertList()
- getAlert()

4.2.4.3. Class Diagram

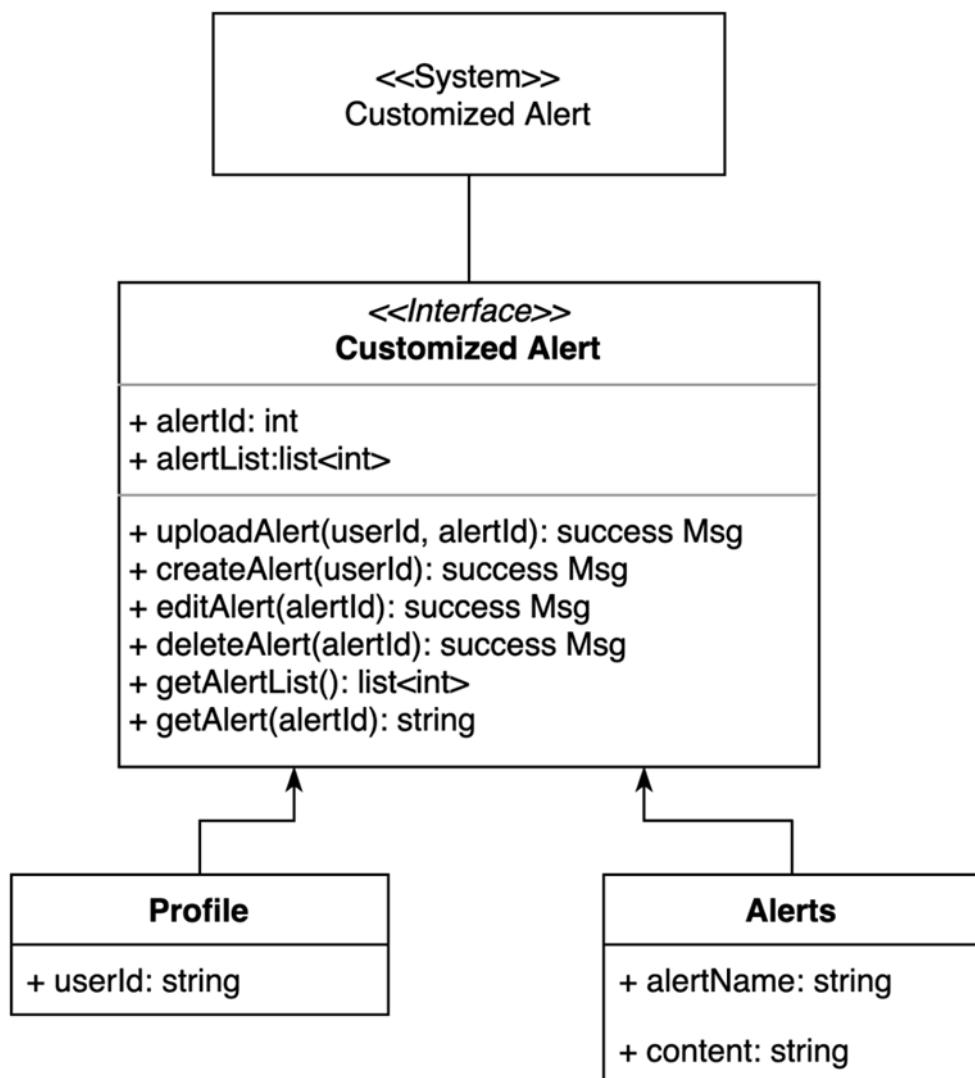


그림 12 Class Diagram – 맞춤형 알림

4.2.4.4. Sequence Diagram

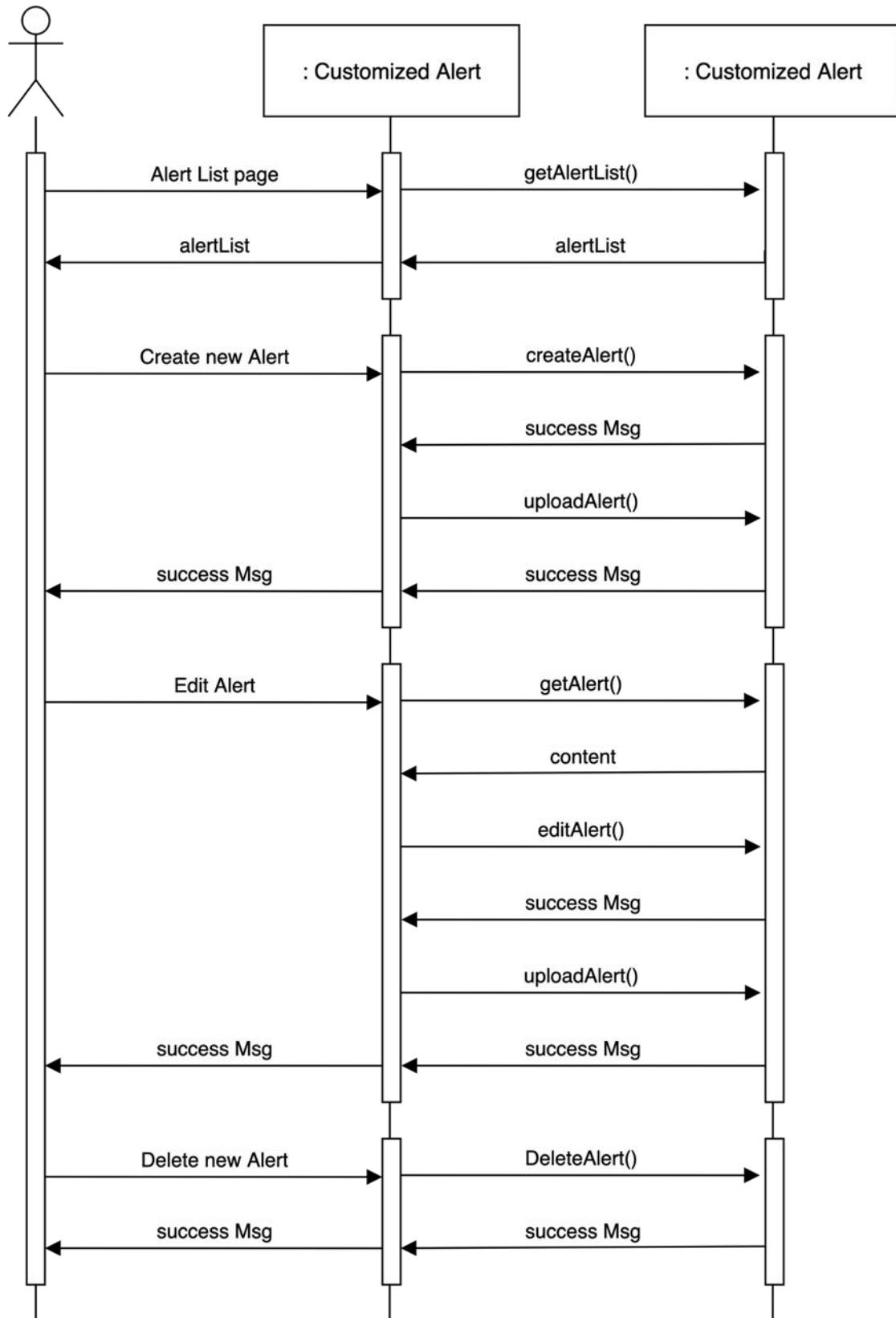


그림 13 Sequence diagram – 맞춤형 알림

4.2.5. 게시물 업로드 및 수정, 삭제

4.2.5.1. Attributes

해당 object 가 가진 attribute 는 다음과 같다.

- post: 수정/삭제하려는 대상 게시물이다.
- Upload Permission: 게시물을 해당 게시판에 업로드할 수 있는 권한 여부를 나타낸다.
- Edit Permission: 해당 게시물을 수정/삭제할 수 있는 권한 여부를 나타낸다.
- Content: 게시물을 업로드/수정 시 게시물의 내용이다.

4.2.5.2. Methods

해당 object 가 가진 methods 는 다음과 같다.

- GetPost()
- GetPermission()
- CreatePost()
- UploadPost()
- EditPost()
- DeletePost()

4.2.5.3. Class Diagram

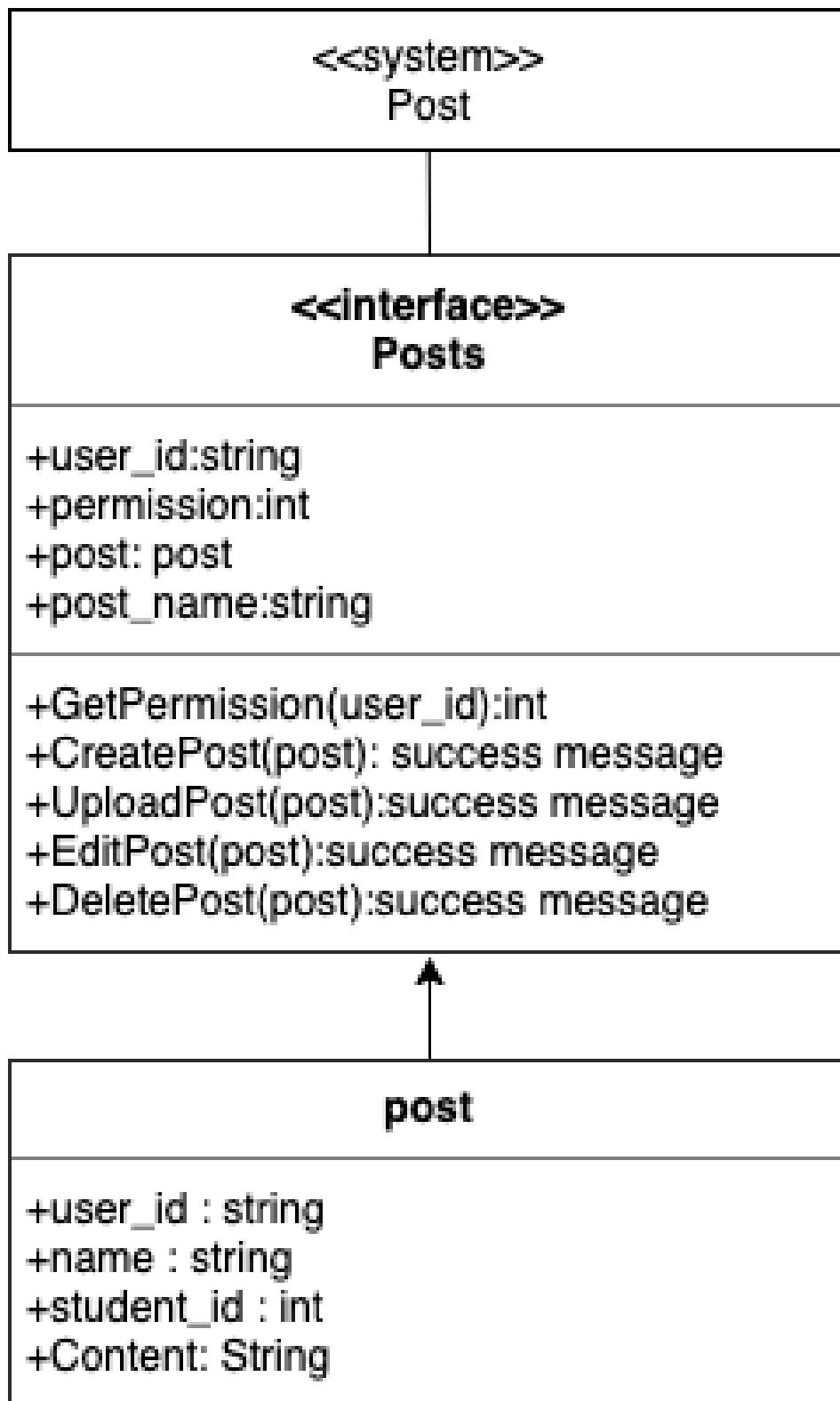


그림 14 Class diagram – 게시글 업로드 및 수정, 삭제

4.2.5.4. Sequence Diagram

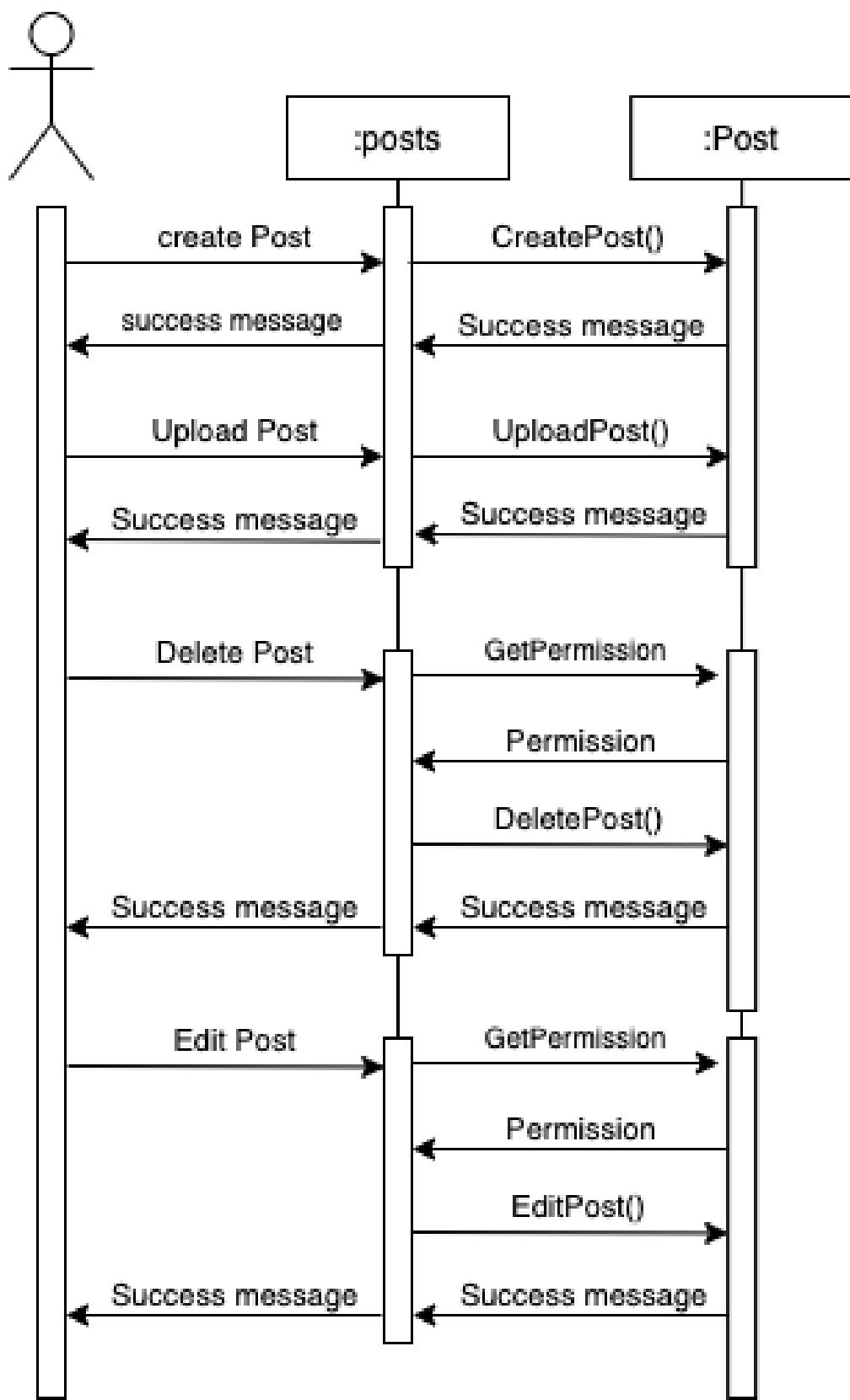


그림 15 Sequence diagram – 게시글 업로드 및 수정, 삭제

4.2.6. 게시물 열람 및 댓글 작성

4.2.6.1. Attributes

해당 object 가 가진 attribute 는 다음과 같다.

- Post: 열람하려는 대상 게시물에 대한 정보이다.
- Comment: 게시물에 달린 댓글 내용이다.
- View Permission: 게시물을 볼 수 있는 권한 여부를 나타낸다.
- Edit Comment Permission: 댓글을 수정/삭제할 수 있는 권한의 여부를 나타낸다.

4.2.6.2. Methods

해당 object 가 가진 methods 는 다음과 같다.

- GetPost()
- GetComments()
- AddComment()
- EditComment()
- DeleteComment()

4.2.6.3. Class Diagram

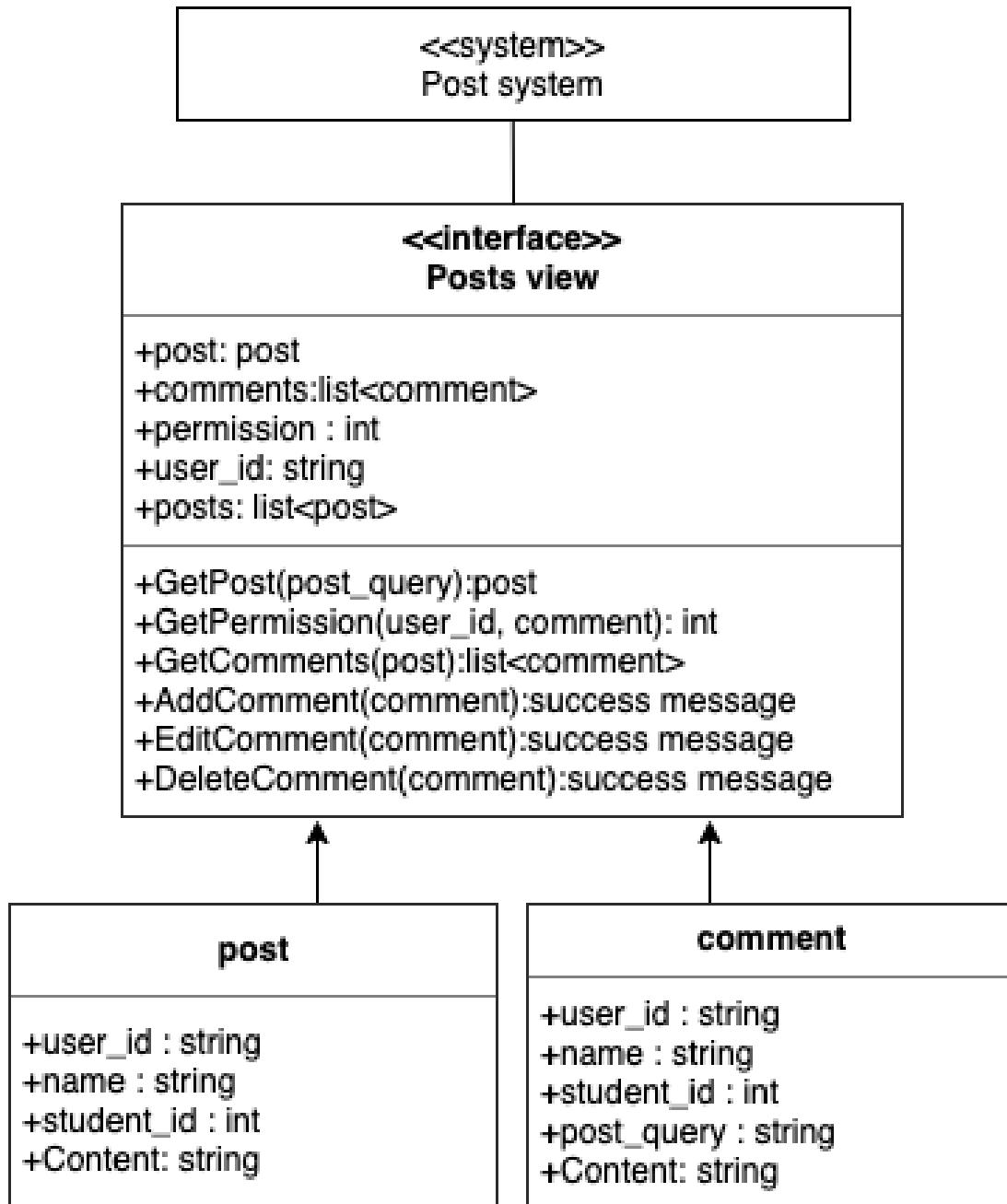


그림 16 Class diagram – 게시글 열람 및 댓글 작성

4.2.6.4. Sequence Diagram

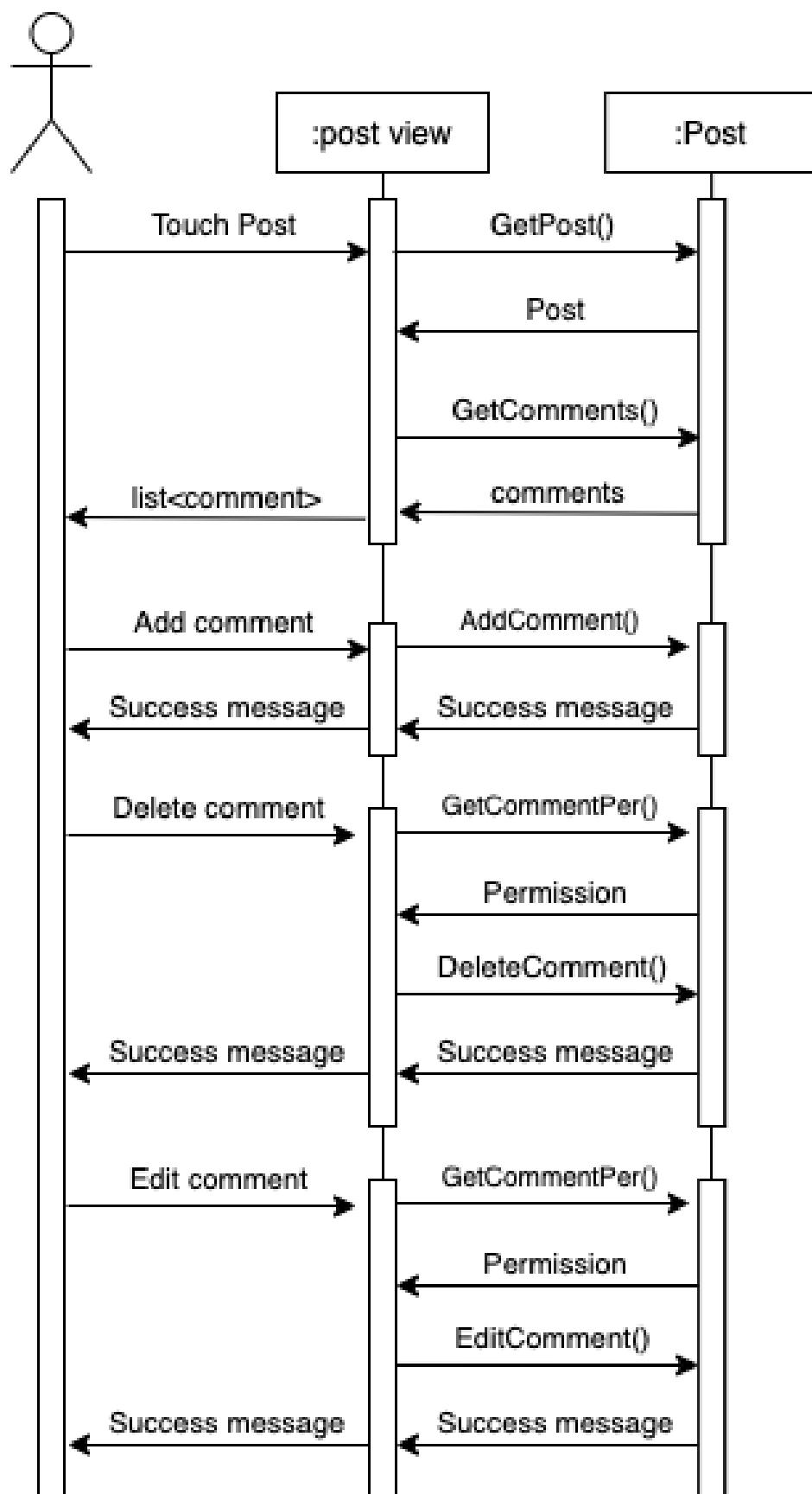


그림 17 Sequence diagram – 게시글 열람 및 댓글 작성

4.2.7. 실시간 공유 문서 생성 및 참여

4.2.7.1. Attributes

해당 object 가 가지는 attributes 는 다음과 같다.

- Comments: 문서에 달린 댓글이다.
- Edit Permission: 문서 편집 권한이다.
- Document: 실시간 작업이 이루어지는 대상 문서이다.
- Modification history: 실시간으로 문서가 수정되는 내역이다.
- Online users: 실시간으로 문서 수정에 참여하고 있는 사용자에 대한 정보이다.

4.2.7.2. Methods

해당 object 가 가지는 method 는 다음과 같다.

- GetEditPermission()
- SendModification()
- CreateDocument()

4.2.7.3. Class Diagram

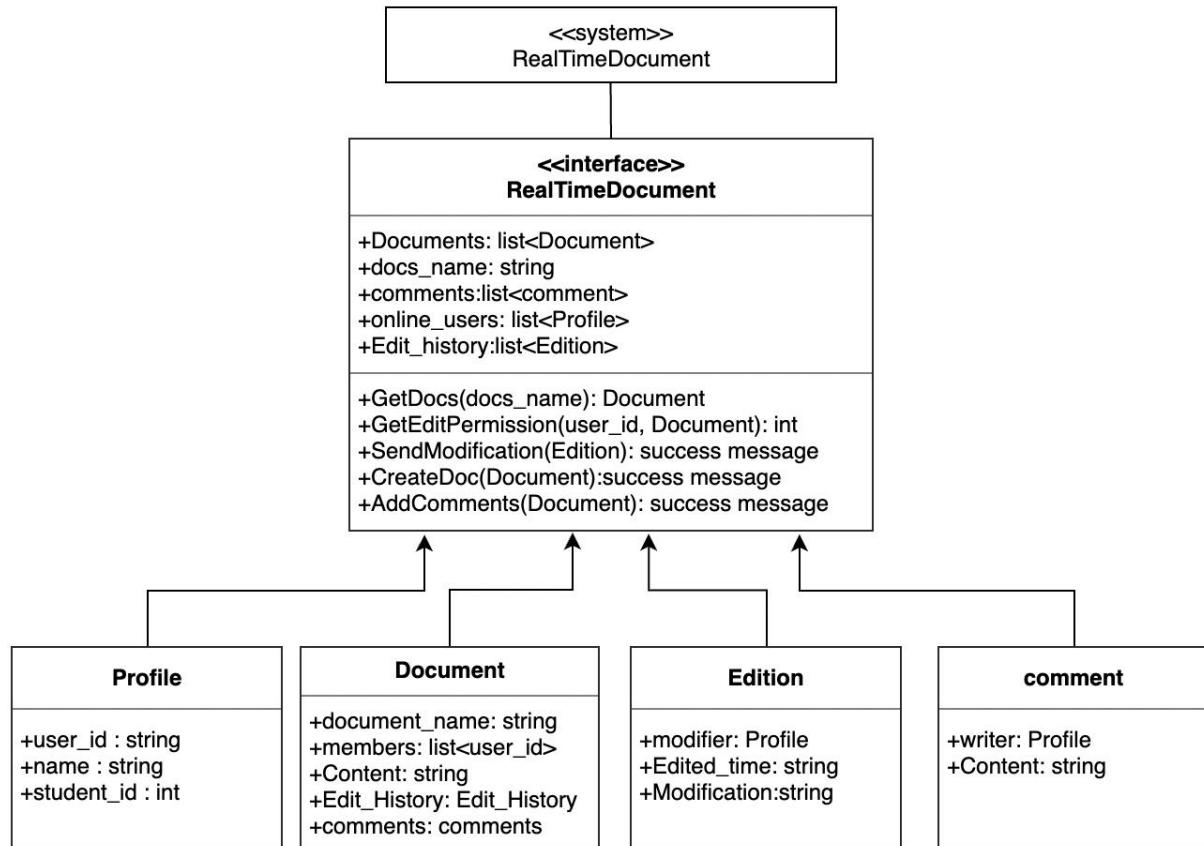


그림 18 Class diagram – 실시간 공유 문서 생성 및 참여

4.2.7.4. Sequence Diagram

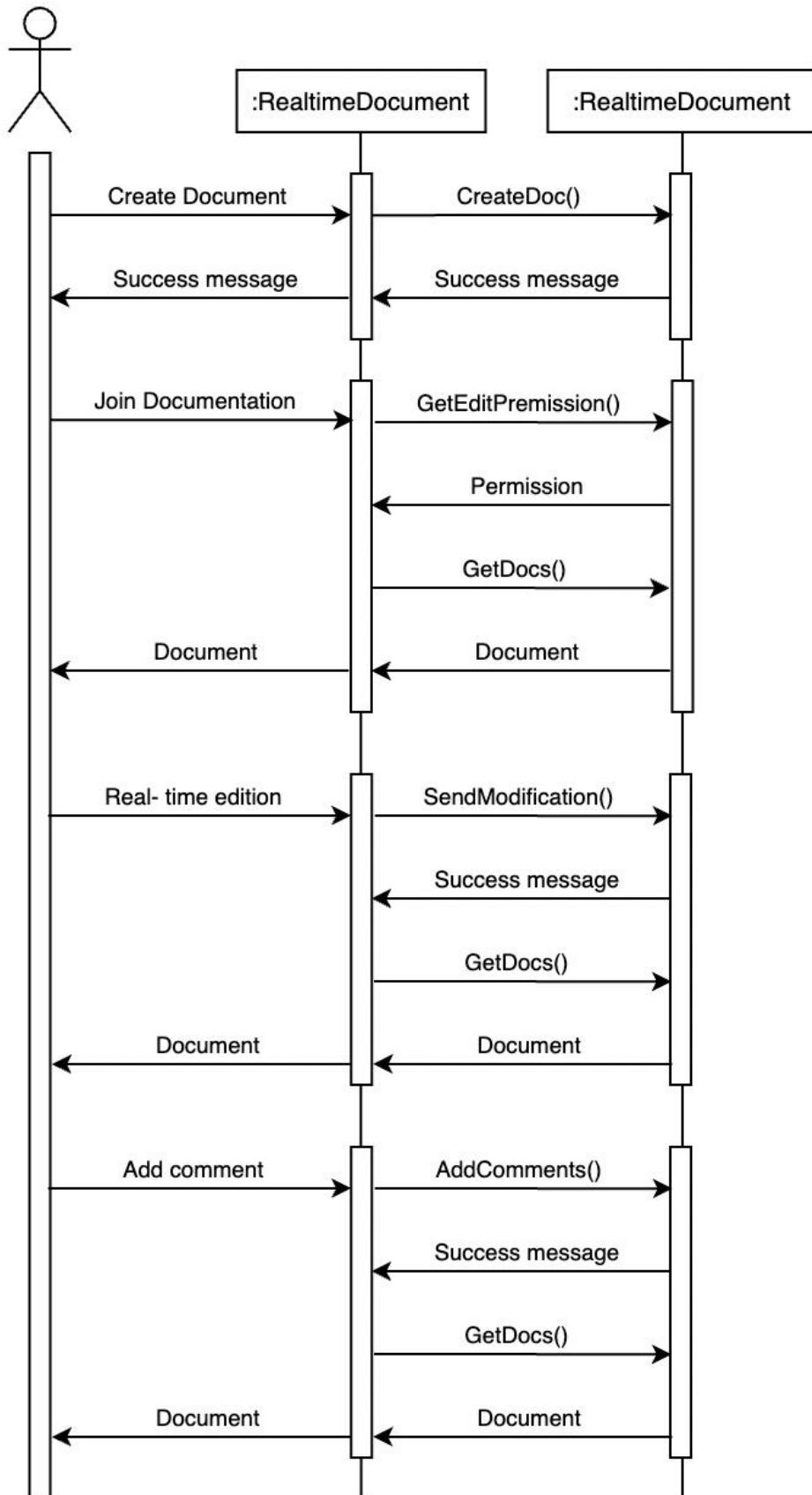


그림 19 Sequence diagram – 실시간 공유 문서 생성 및 참여

4.2.8. 강의 클립 생성 및 업로드

4.2.8.1. Attributes

강의 클립화 object에서의 attribute는 다음과 같다.

- Lecture: 생성하려는 강의 클립의 대상 강의에 대한 정보이다.
- Clip: 강의 클립에 대한 정보이다.
- uploader_student_id: 강의 클립을 업로드한 사람에 대한 정보이다.
- Bookmarked_users: 강의 클립을 즐겨찾기한 사용자들의 목록이다.
- Bookmark_num: 강의 클립의 즐겨찾기 개수이다.
- Uploaded_time: 강의 클립이 업로드된 시간이다.
- content: 강의 클립의 내용이다.

4.2.8.2. Methods

강의 클립화 object에서의 methods는 다음과 같다.

- CreateClip()
- UploadClip()

4.2.8.3. Class Diagram

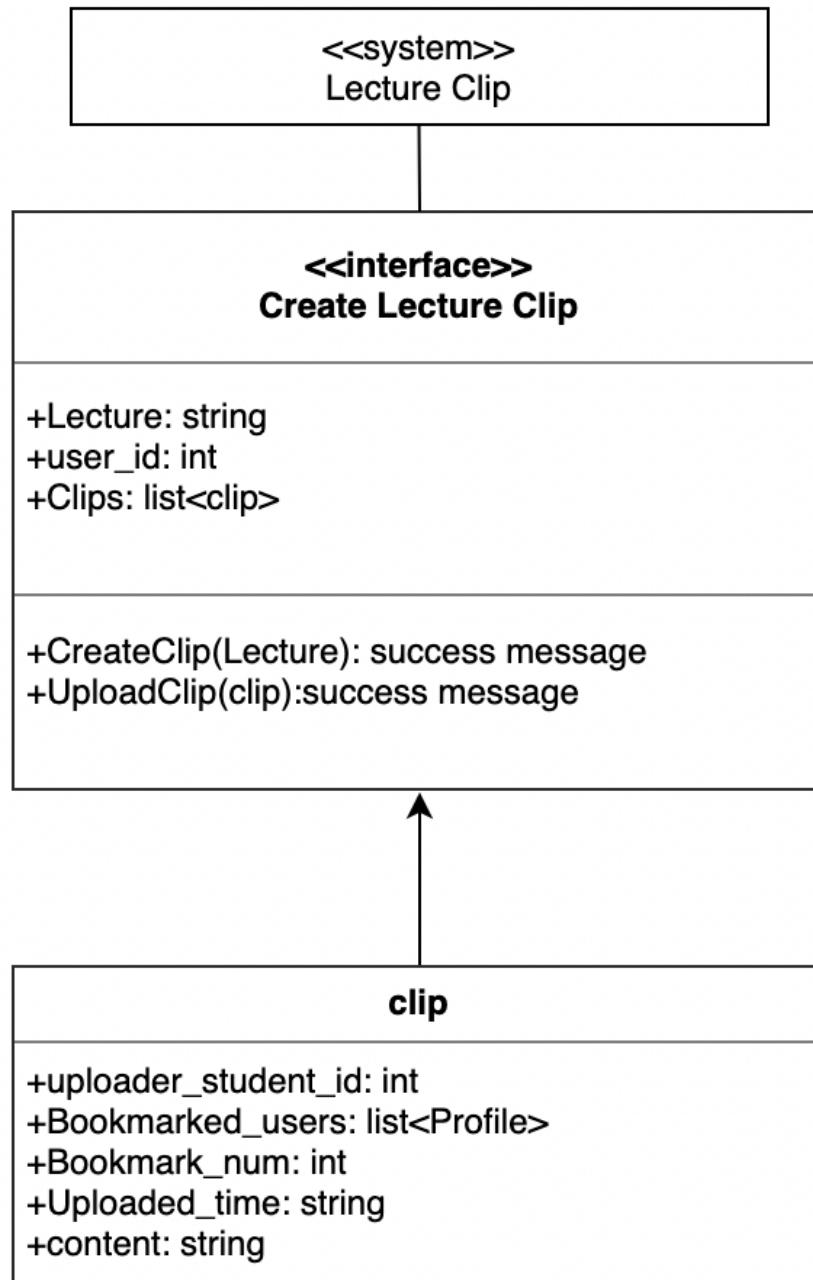


그림 20 Class diagram – 강의 클립 생성 및 업로드

4.2.8.4. Sequence Diagram

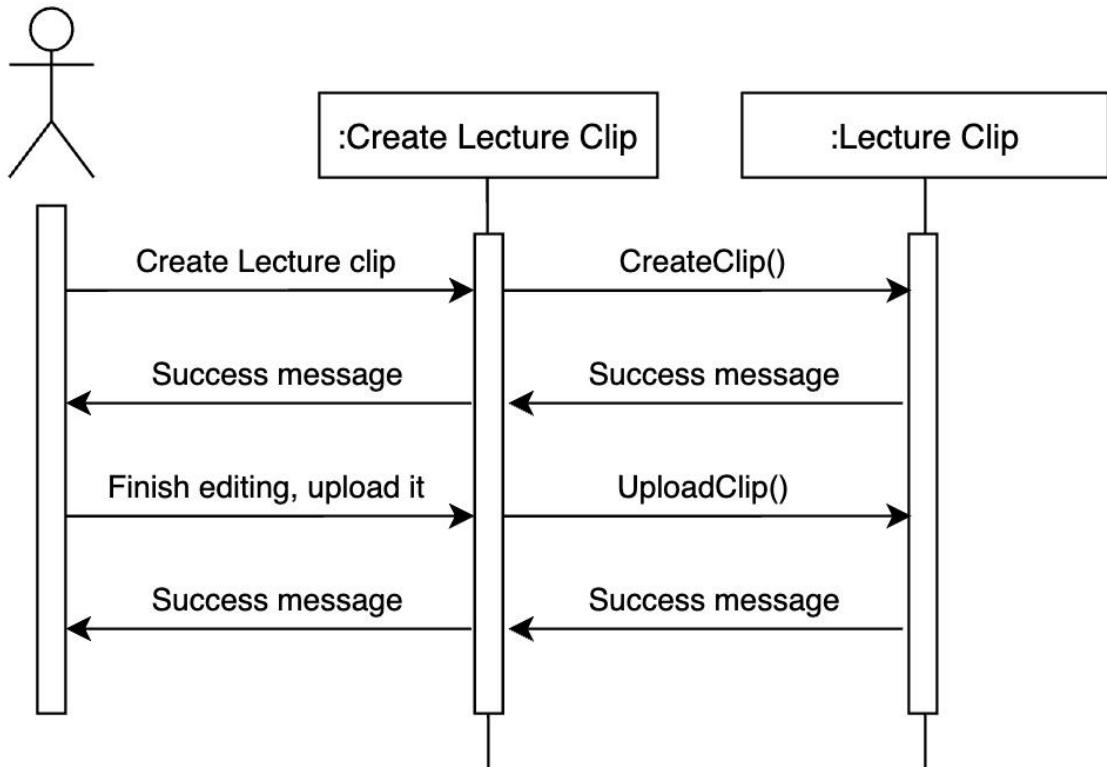


그림 21 Sequence diagram – 강의 클립 생성 및 업로드

4.2.9. 강의 클립 열람

4.2.9.1. Attributes

해당 object에서의 attribute는 다음과 같다.

- Sort_by: 강의 클립 목록을 정렬하는 기준이다.
- Search_query: 강의 클립 검색시 검색어이다.
- Clip: 강의 클립에 대한 정보이다.
- uploader_student_id: 강의 클립을 업로드한 사람에 대한 정보이다.
- Bookmarked_users: 강의 클립을 즐겨찾기한 사용자들의 목록이다.
- Bookmark_num: 강의 클립의 즐겨찾기 개수이다.
- Uploaded_time: 강의 클립이 업로드된 시간이다.
- content: 강의 클립의 내용이다.

4.2.9.2. Methods

해당 object에서의 methods는 다음과 같다.

- SortClip()
- Bookmark()
- RequestSearch()

4.2.9.3. Class Diagram

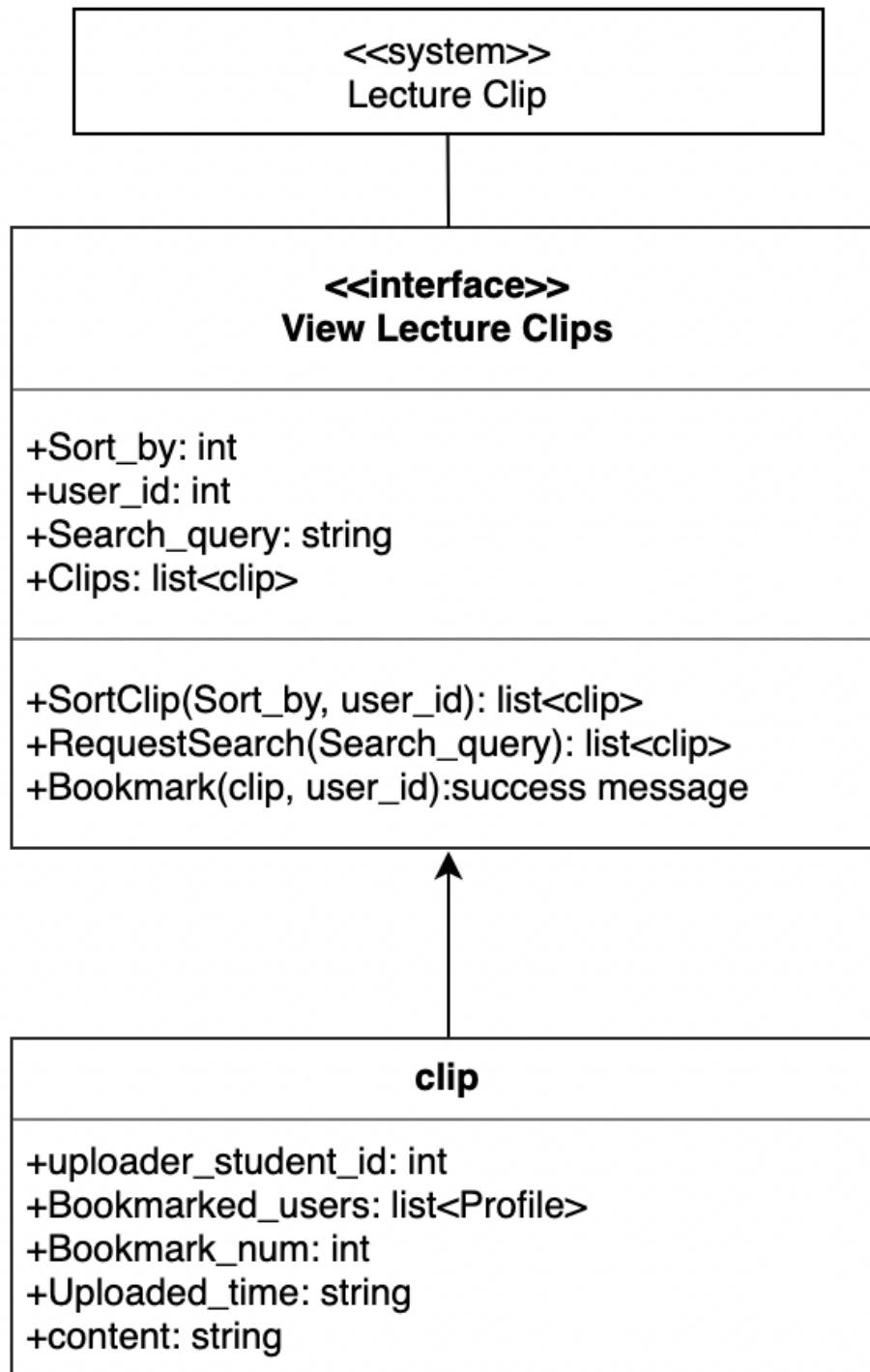


그림 22 Class diagram – 강의 클립 열람

4.2.9.4. Sequence Diagram

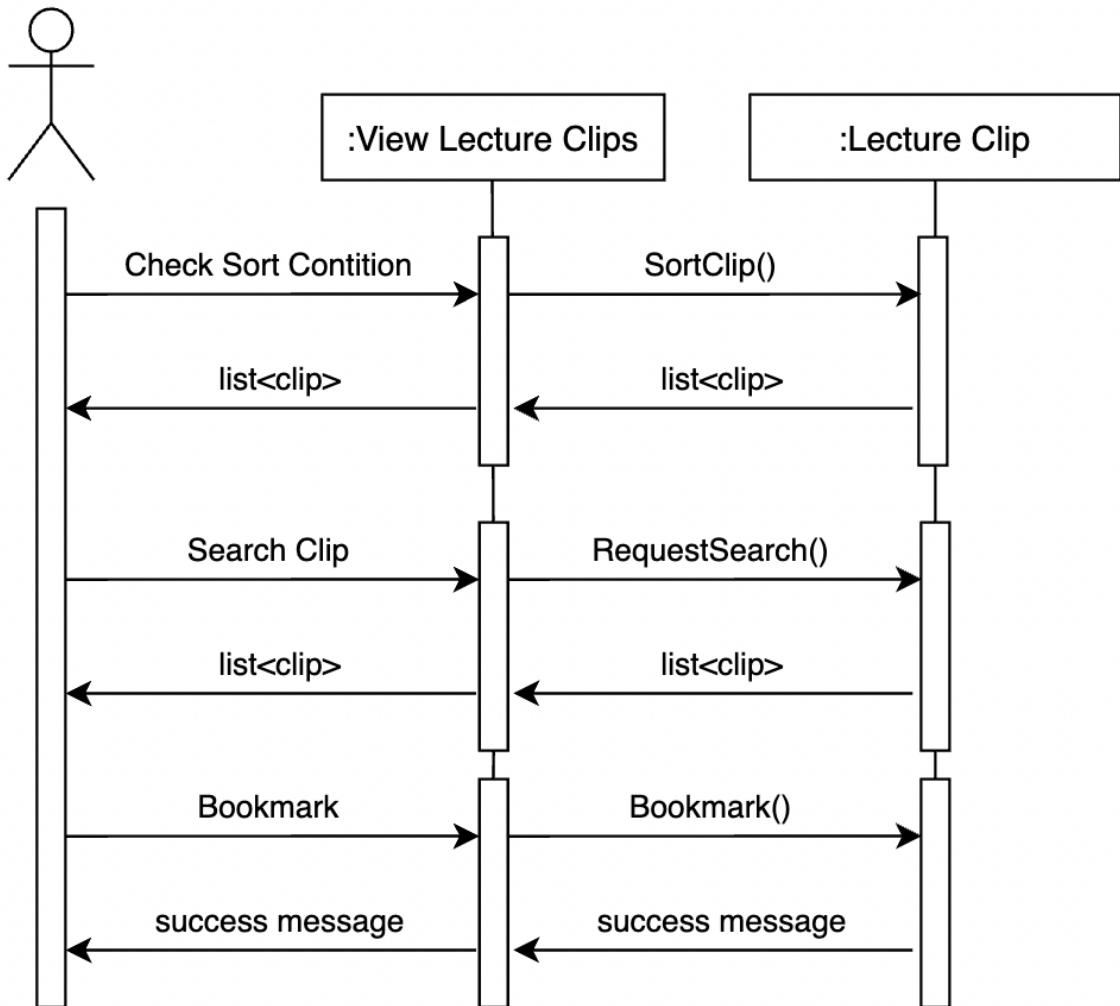


그림 23 Sequence diagram – 강의 클립 열람

4.2.10. 강의 노트 공유

4.2.10.1. Attributes

강의 노트 object 에서의 attribute 는 다음과 같다.

- Lecture: 생성하려는 강의 노트의 대상 강의에 대한 정보이다.
- Sort_by: 강의 노트 목록을 정렬하는 기준이다.
- Search_query: 강의 노트 검색시 검색어이다.
- Clip: 강의 클립에 대한 정보이다.
- uploader_student_id: 강의 노트를 업로드한 사람에 대한 정보이다.
- Bookmarked_users: 강의 노트를 즐겨찾기한 사용자들의 목록이다.
- Bookmark_num: 강의 노트의 즐겨찾기 개수이다.
- Uploaded_time: 강의 노트가 업로드된 시간이다.
- content: 강의 노트의 내용이다.

4.2.10.2. Methods

강의 클립화 object 에서의 methods 는 다음과 같다.

- UploadNote()
- SortNote()
- Bookmark()
- RequestSearch()

4.2.10.3. Class Diagram

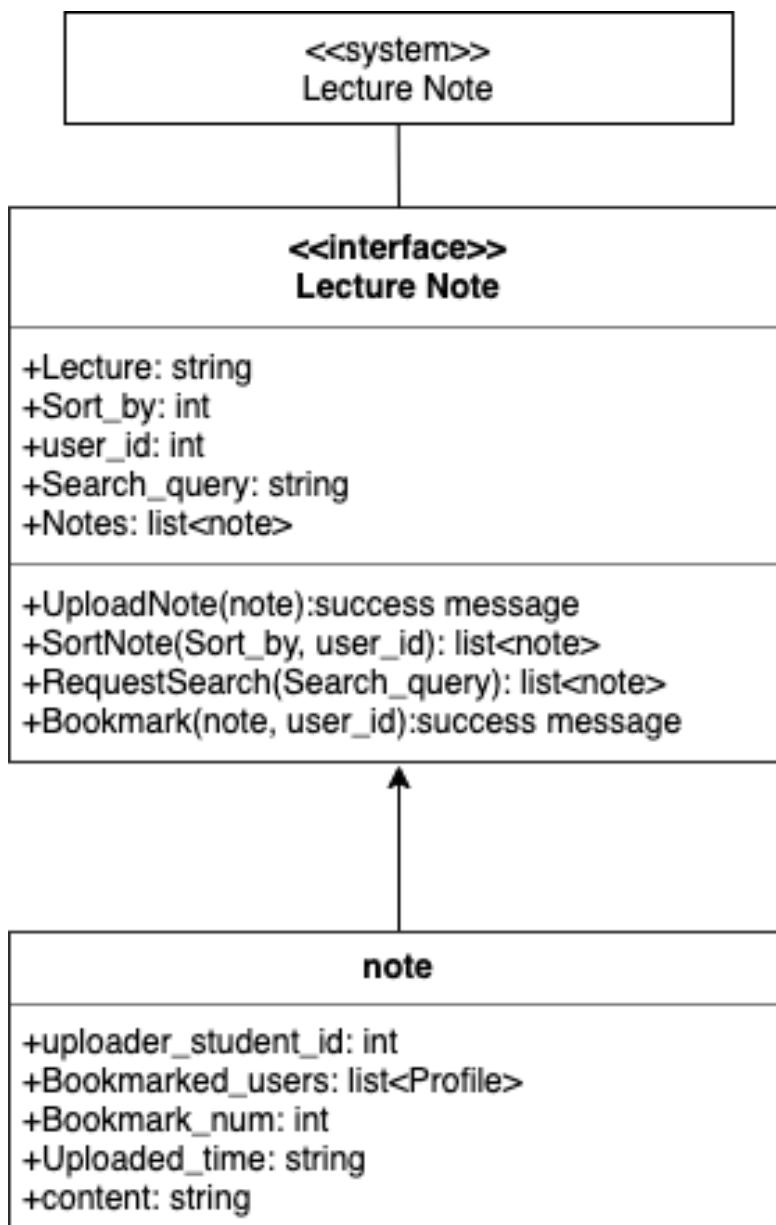


그림 24 Class diagram – 강의 노트 공유

4.2.10.4. Sequence Diagram

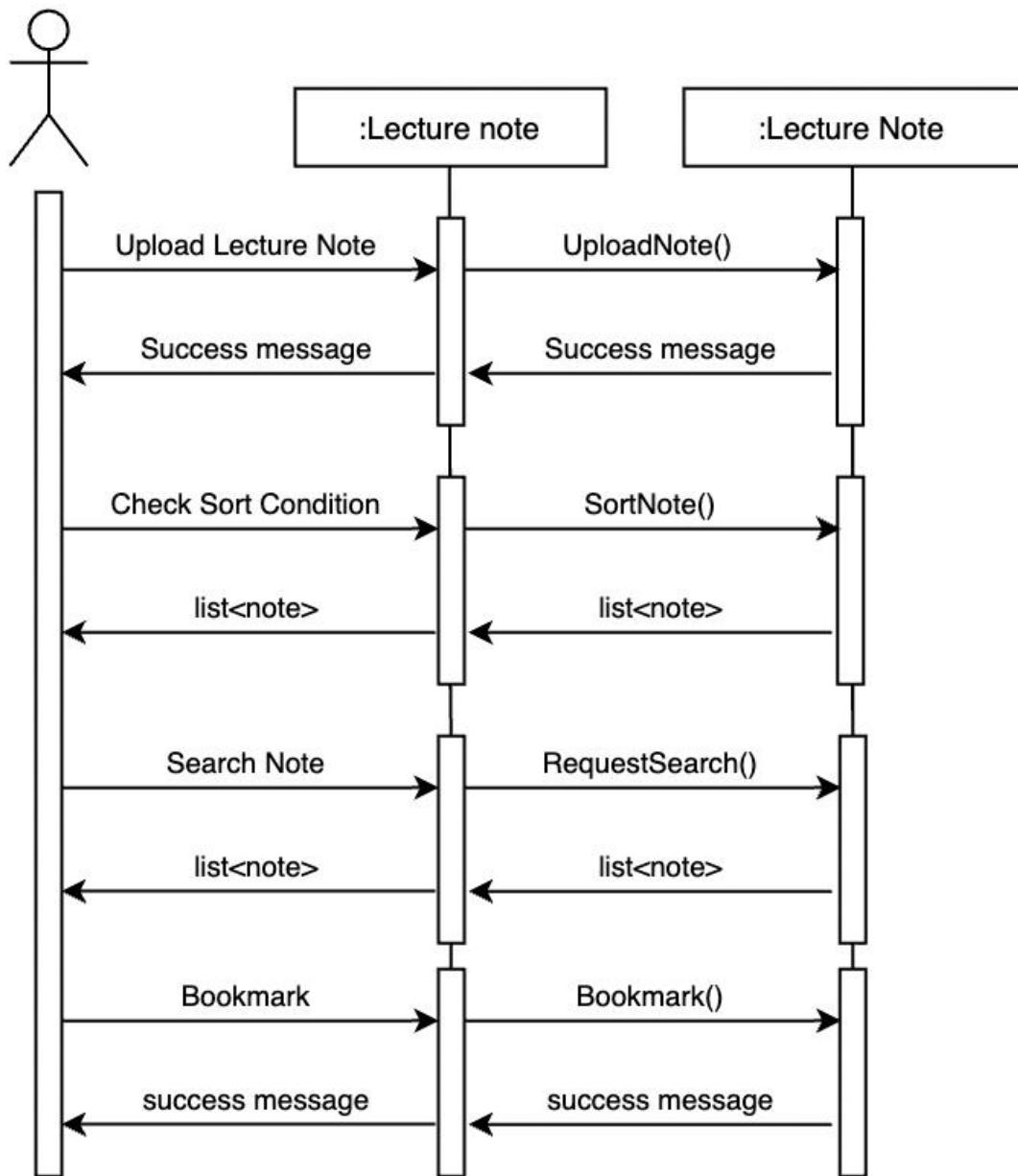


그림 25 Sequence diagram – 강의 노트 공유

4.2.11. eBook

4.2.11.1. Attributes

해당 object 가 가지는 attribute 는 다음과 같다.

- eBook: 사용자가 구독한 eBook 에 대한 정보이다.

4.2.11.2. Methods

해당 object 가 가지는 method 는 다음과 같다.

- Get_Subscription()

4.2.11.3. Class Diagram

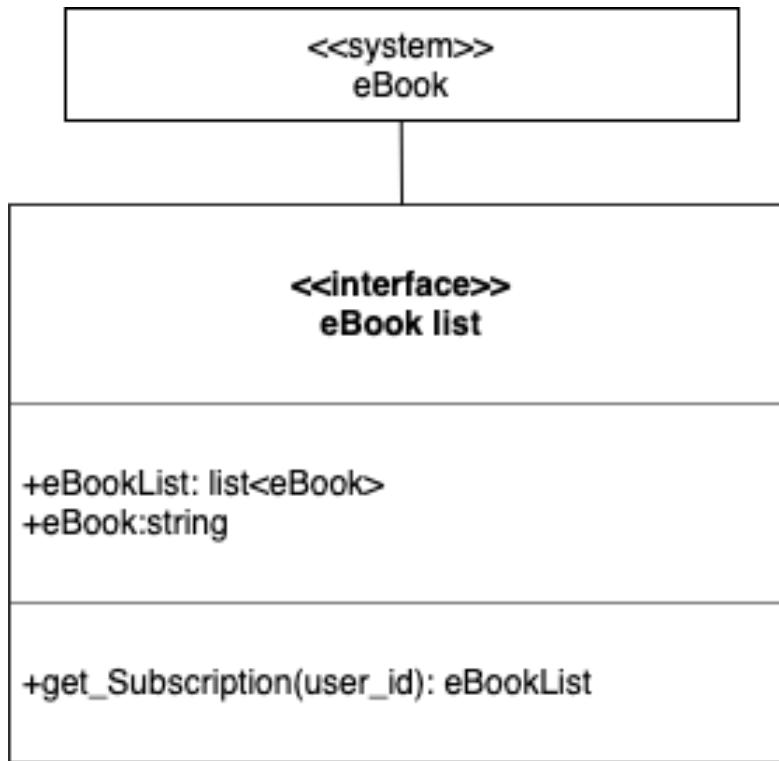


그림 26 Class diagram – eBook

4.2.11.4. Sequence Diagram

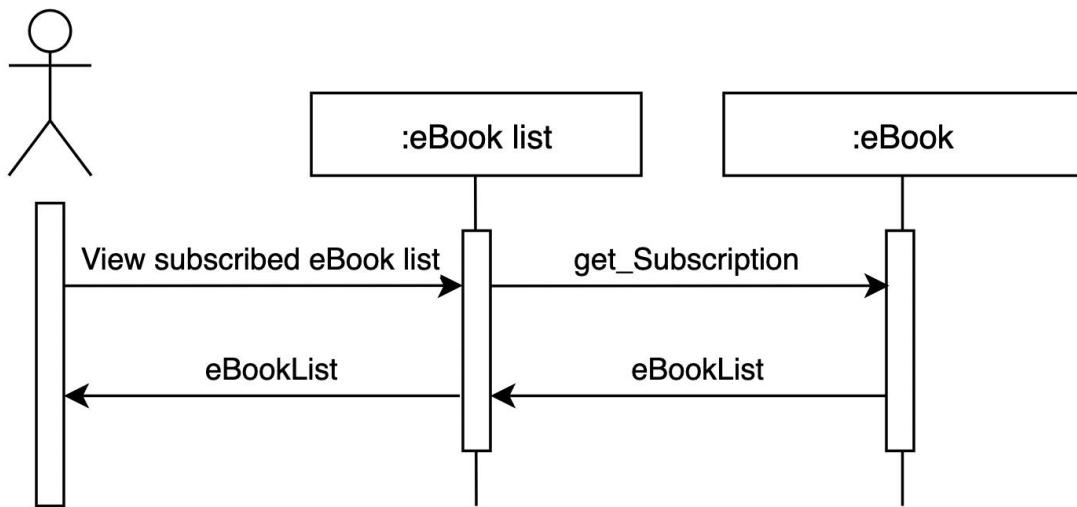


그림 27 Sequence diagram – eBook

4.2.12. 채팅

4.2.12.1. Attributes

해당 object 가 가지는 attribute 는 다음과 같다.

- chat room name: 채팅방의 이름
- chat room type: 채팅방의 종류(음성인지, 텍스트인지)
- participants: 채팅방에 참여중인 사용자
- mute: 음성 채팅방의 경우 음소거의 여부
- chat contents: 텍스트 채팅방의 경우 채팅 내용

4.2.12.2. Methods

해당 object 가 가지는 method 는 다음과 같다.

- Join_chatting()
- Create_room()
- Invite_chat()
- Leave_chat()

4.2.12.3. Class Diagram

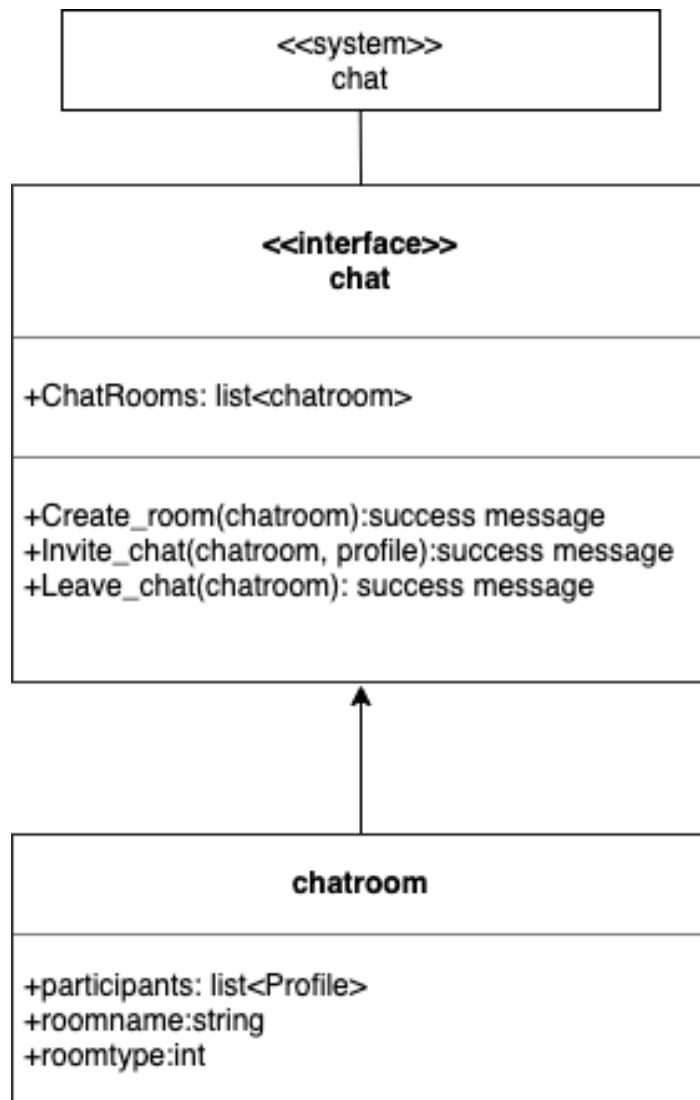


그림 28 Class diagram – 채팅

4.2.12.4. Sequence Diagram

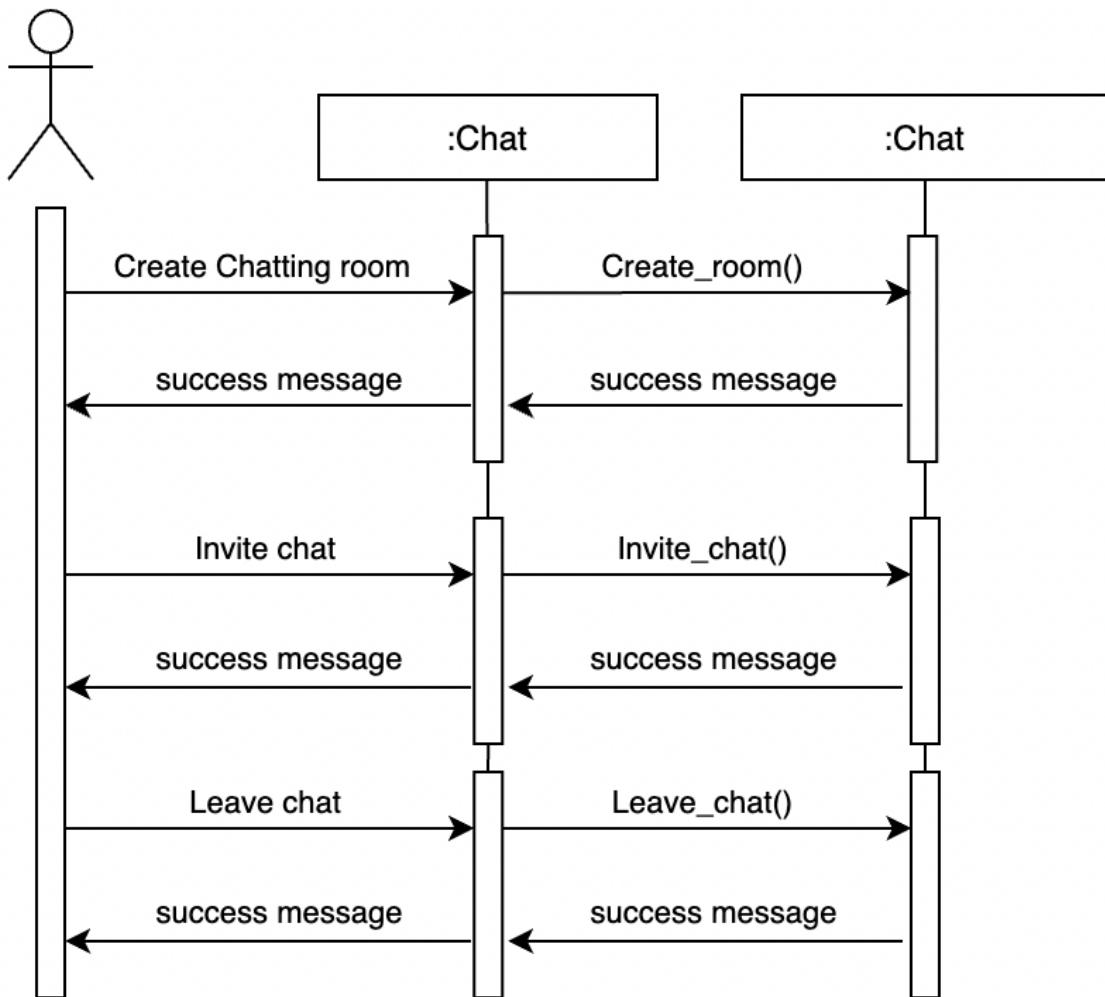


그림 29 Sequence diagram – 채팅

5. System Architecture – Backend

5.1. Objectives

이 챕터에서는 System Architecture 중에서 Firebase를 통해 구현할 백엔드 시스템의 구조와 기능에 대해서 설명한다.

5.2. Overall Architecture

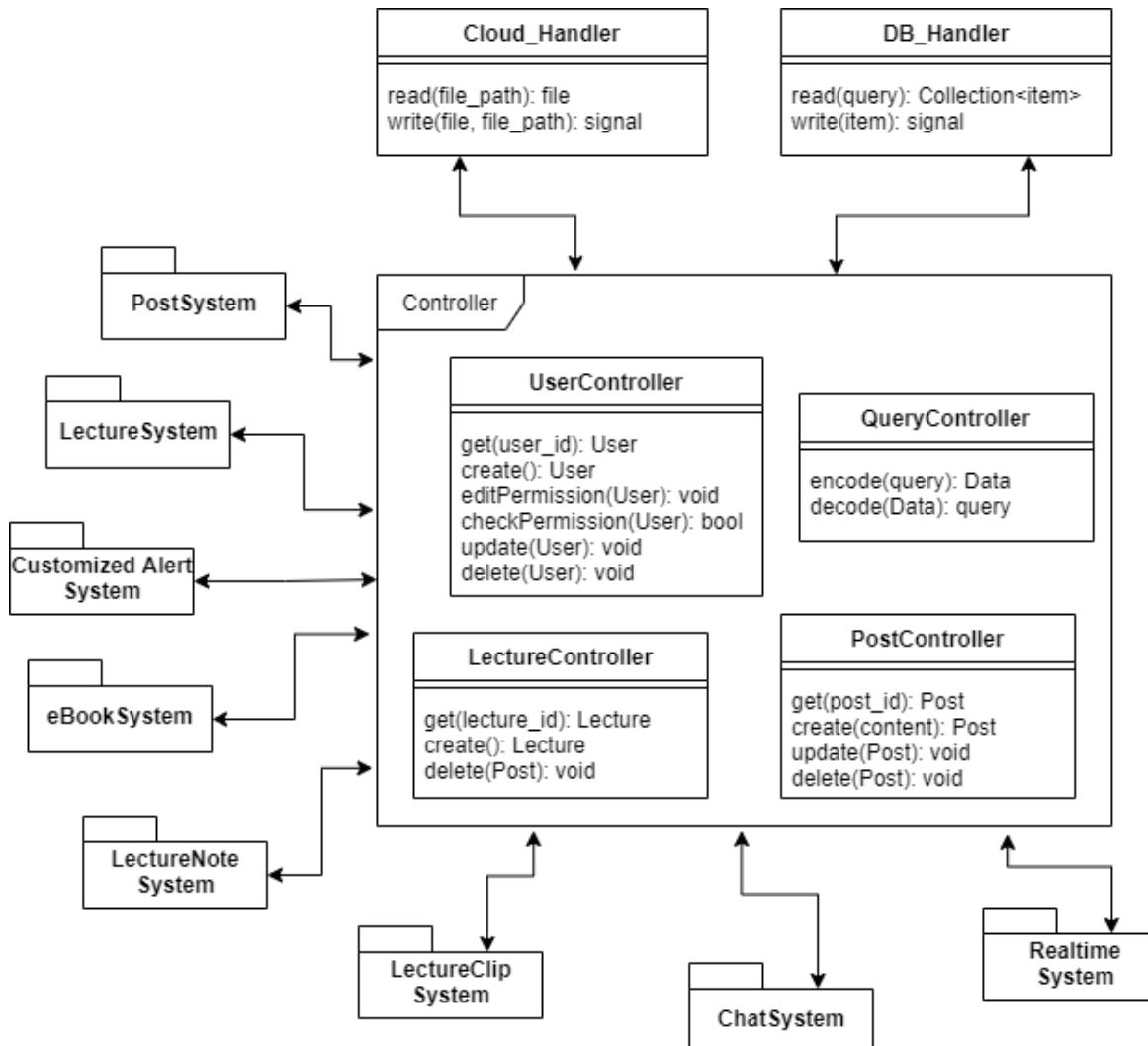


그림 30 Overall Architecture

위 그림은 백엔드 시스템의 전체적인 아키텍처를 나타낸다. 백엔드 시스템은 Firebase를 통해 구현되며, System, Handler, 그리고 Controller로 구성된다.

5.2.1. System

각 System은 세부 기능을 처리하는 역할을 하며 Post System, Lecture System, eBook System, Lecture Note System, Lecture Clip System, Chat System, 그리고 Real-time Document System 등 있다.

5.2.2. Handler

Handler는 DB Handler와 Cloud Handler가 있으며, DB Handler는 데이터베이스에서 데이터를 읽고 쓸 수 있도록 하는 인터페이스 역할을 하고, Cloud Handler는 클라우드 저장소에 있는 파일을 읽고 쓸 수 있도록 하는 인터페이스 역할을 한다.

5.2.3. Controller

Controller는 System과 Handler 사이에 있으며 User Controller, Lecture Controller, Post Controller, 그리고 Query Controller가 있다.

5.3. Subcomponents

5.3.1. Customized Alert System

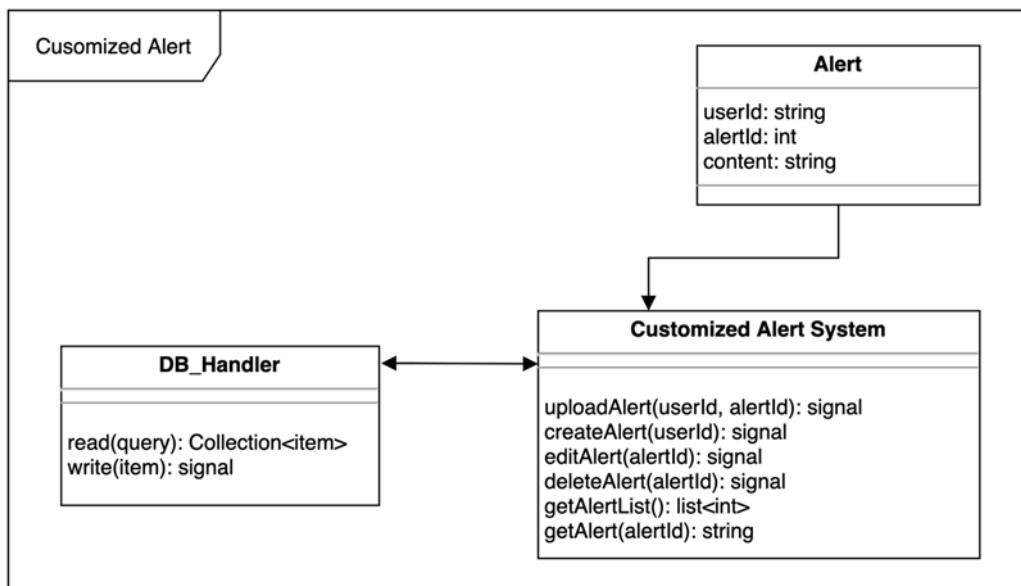


그림 31 Class Diagram – Customized Alert System

Customized Alert System Class는 맞춤형 알림을 생성, 수정, 삭제할 수 있도록 한다. 사용자가 새로운 알림을 생성하면 `uploadAlert()` 함수와 DB_Handler를 통해 생성된 알림 데이터를 업로드 할 수 있으며 유저의 알림 리스트에 불러올 수 있다. 유저가 알림을 수정하면 `editAlert()` 함수와 DB_Handler를 통해 기존 알림 데이터를 불러오고 수정 후 재 업로드할 수 있다. 유저가 알림을 삭제하면 `deleteAlert()` 함수와 DB_Handler를 통해 알림 데이터를 삭제한다.

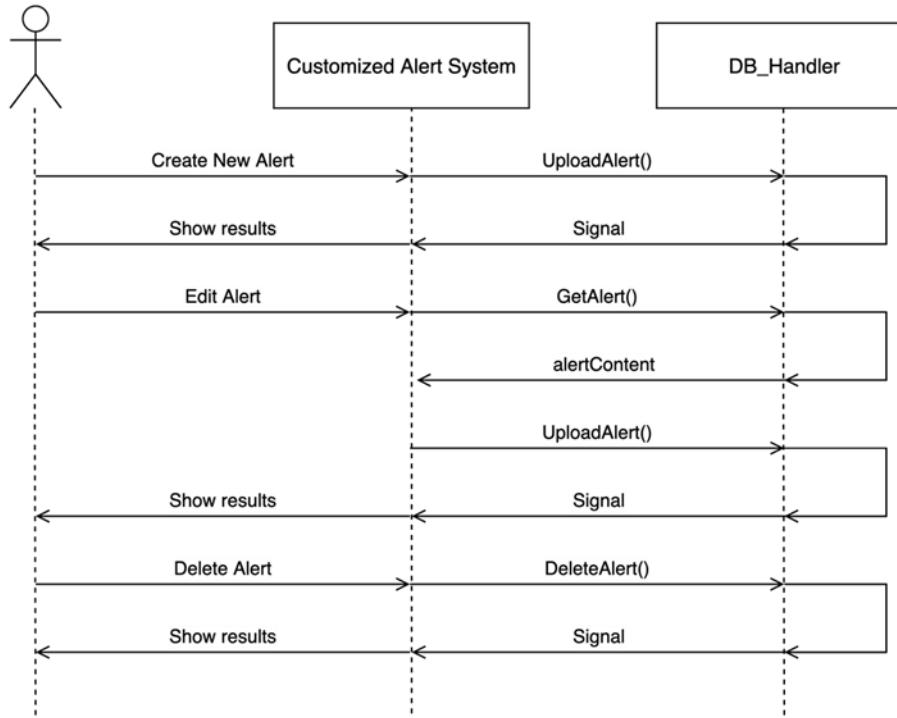


그림 32 Sequence Diagram – Customized Alert System

5.3.2. Post System

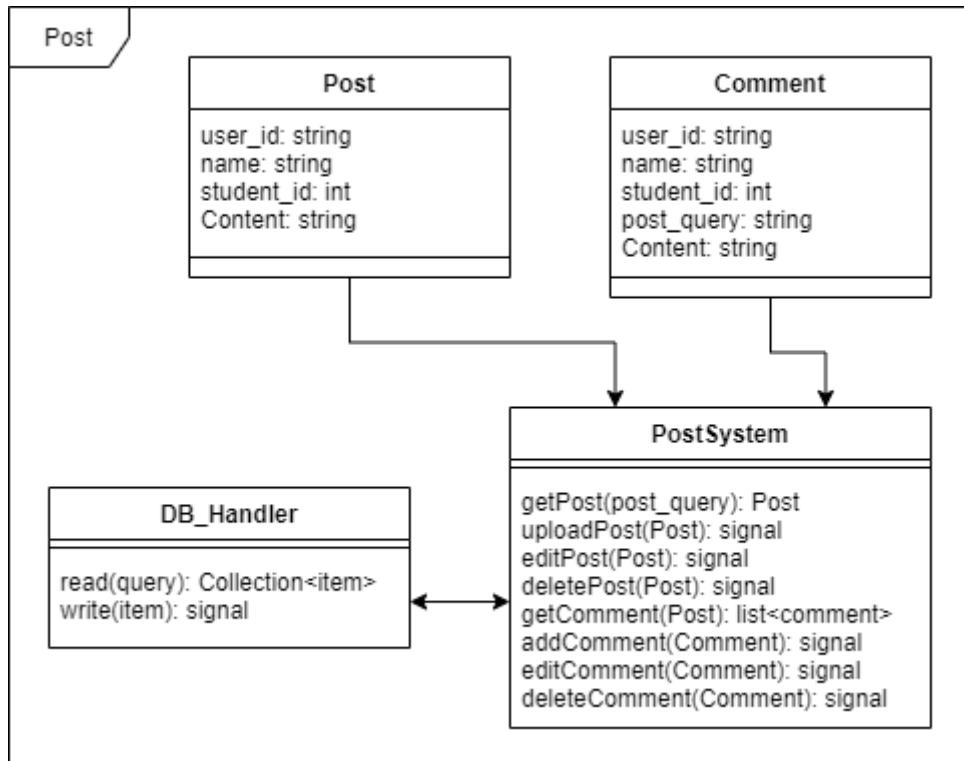


그림 33 Class Diagram – Post System

Post System Class 는 게시글을 업로드하고 열람할 수 있도록하는 역할을 한다. 사용자가 게시글을 열람하면 `getPost()` 함수와 DB_Handler 를 통해 게시글 데이터를 불러올 수 있고, `addComment()` 함수를 통해 해당 게시글에 댓글을 달 수 있다. 유저가 게시글을 작성하면 Post 를 생성해서 `uploadPost()` 함수와 DB_Handler 를 통해 데이터베이스에 저장한다. 또한 `editPost()`, `deletePost()`, `editComment()`, `deleteComment()` 함수를 통해 게시글과 댓글을 수정하고 지울 수 있다.

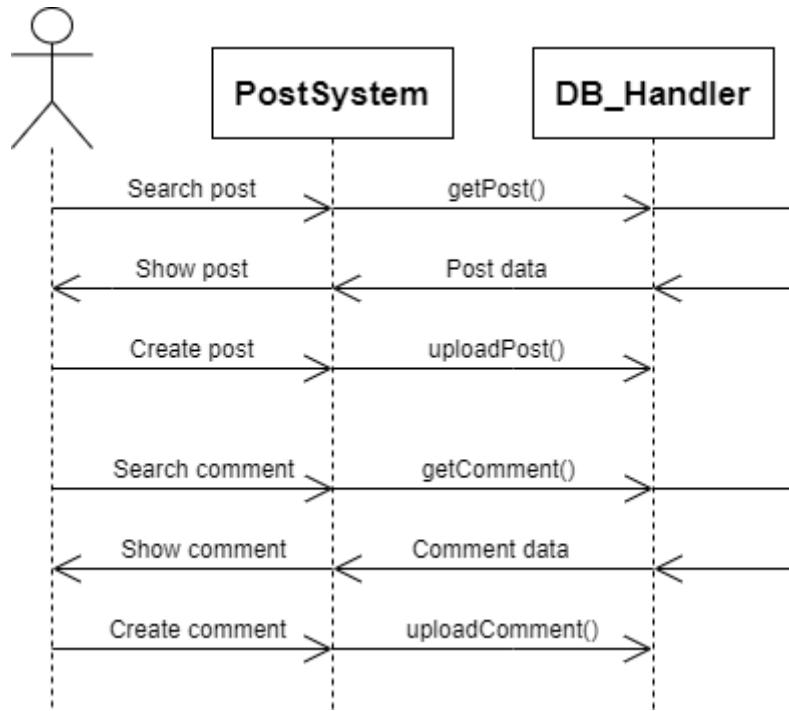


그림 34 Sequence Diagram – Post System

5.3.3. Lecture System

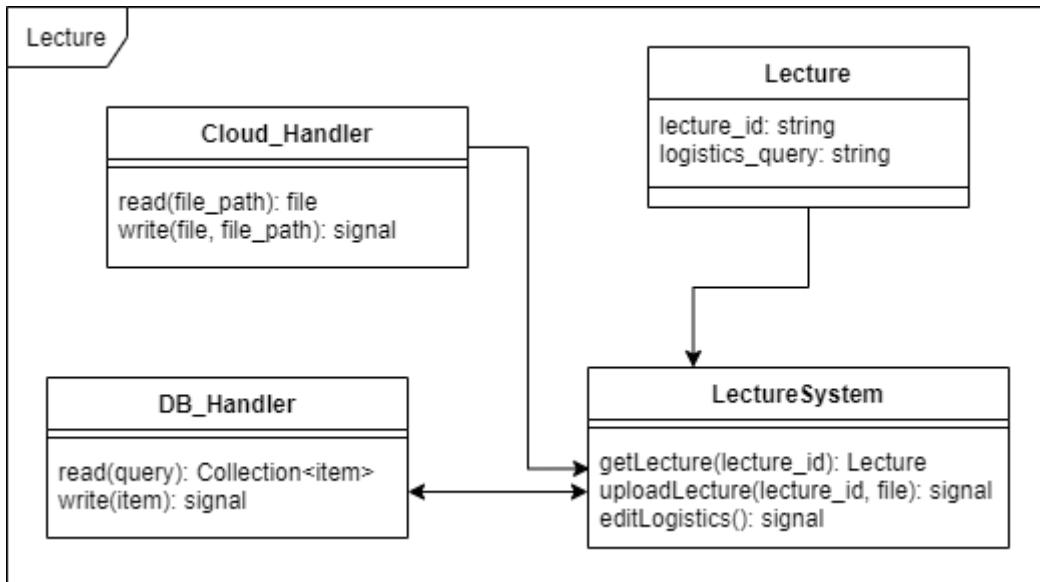


그림 35 Class Diagram – Lecture System

Lecture System Class 는 강의 컨텐츠를 업로드하고 열람할 수 있도록하는 역할을 한다. 강의자가 강의 컨텐츠를 업로드하면 uploadLecture() 함수와 DB_Handler 를 통해 정보를 데이터베이스에 저장하고, Cloud_Handler 를 통해 동영상을 Cloud 저장소에 저장한다. 또한 강의자는 editLogistics() 함수를 통해 성적 반영 비율이나 수강생들의 성적 같은 강의 정보를 편집할 수 있다. 수강생들은 강의 컨텐츠를 열람할 수 있다.

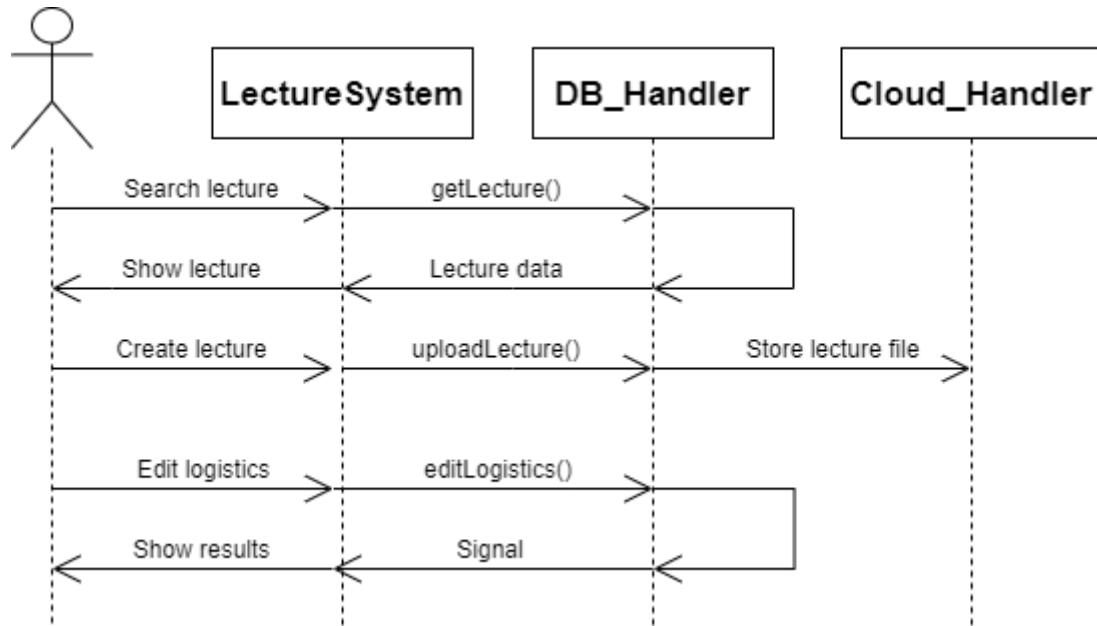


그림 36 Sequence Diagram – Lecture System

5.3.4. Lecture Note System

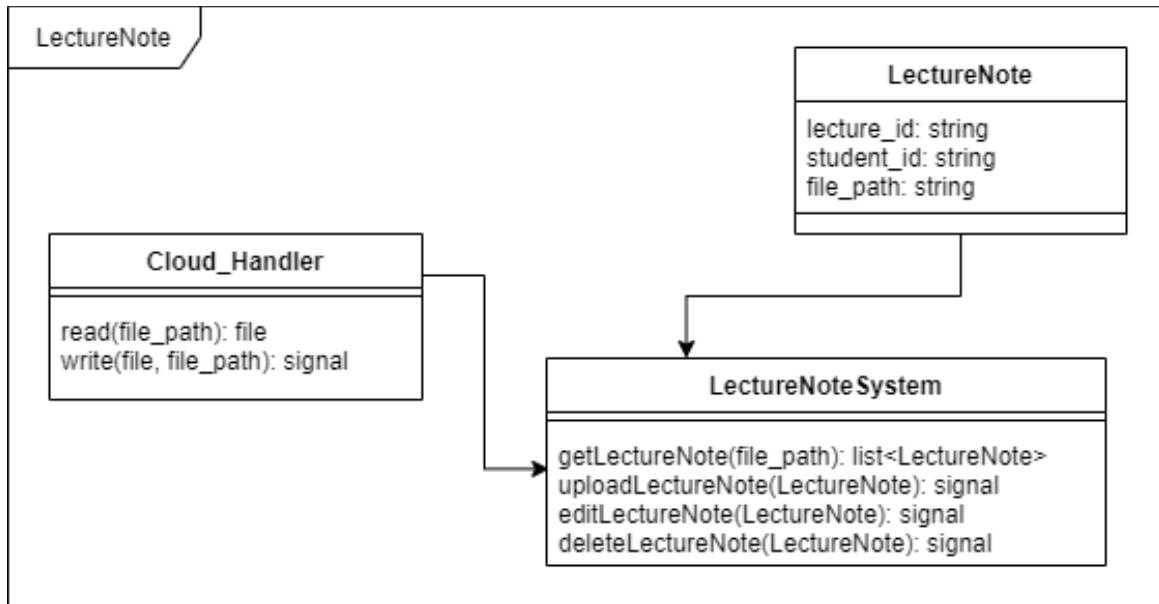


그림 37 Class Diagram – Lecture Note System

Lecture Note System Class 를 통해 강의 노트를 공유할 수 있다. 수강생이 강의 노트 파일을 업로드하면 uploadLectureNote() 함수와 Cloud_Handler 를 통해 Cloud 저장소에 강의 노트 파일을 업로드할 수 있고 다른 수강생들은 getLectureNote() 함수를 통해 강의 노트를 검색하고 열람할 수 있다.

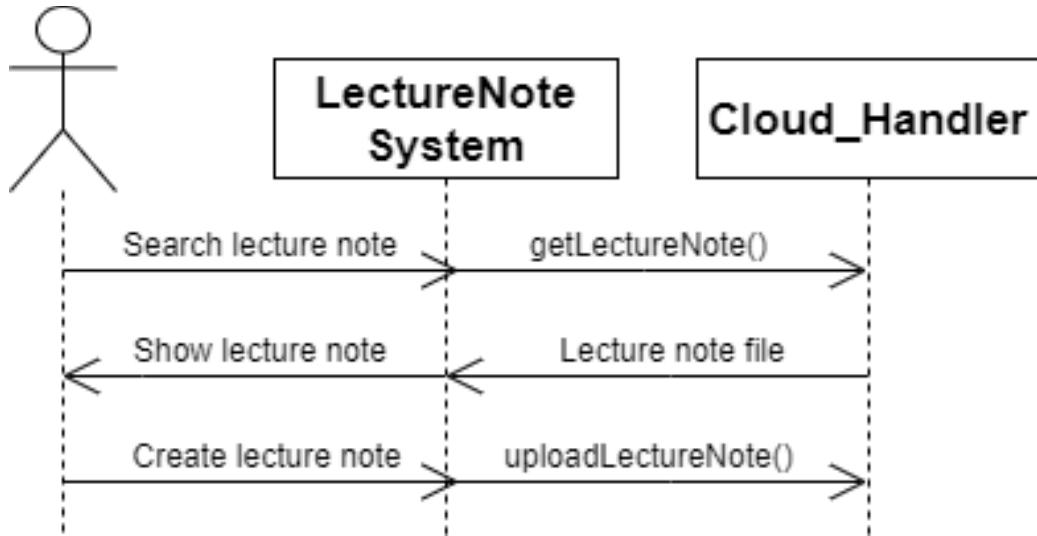


그림 38 Sequence Diagram – Lecture Note System

5.3.5. Lecture Clip System

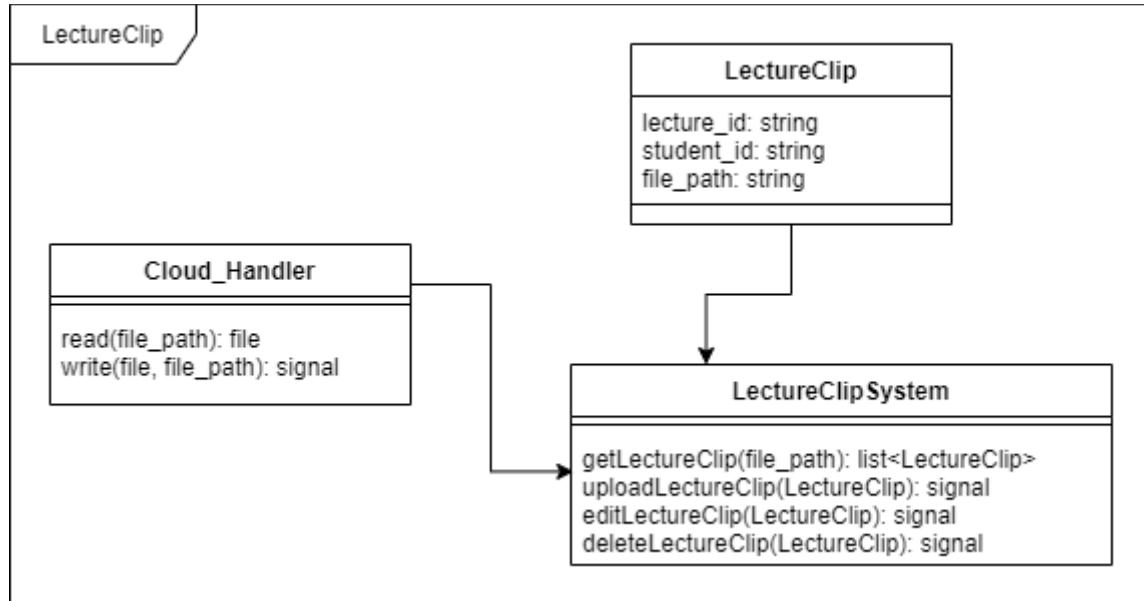


그림 39 Class Diagram – Lecture Clip System

Lecture Clip System Class 를 통해 강의 클립을 관리할 수 있다. 생성한 강의 클립을 수강생이 업로드하면 uploadLectureClip() 함수와 Cloud_Handler 를 통해 Cloud 저장소에 강의 클립을 저장할 수 있다.

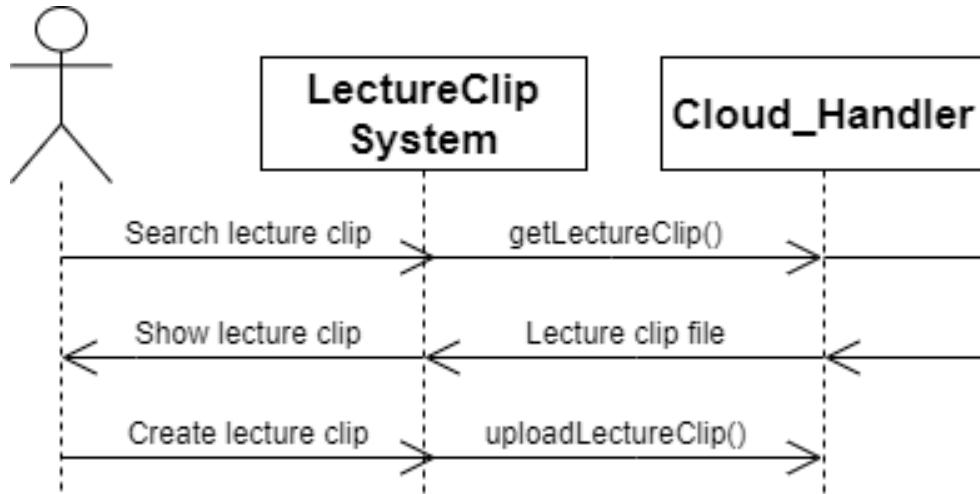


그림 40 Sequence Diagram – Lecture Clip System

5.3.6. eBook System

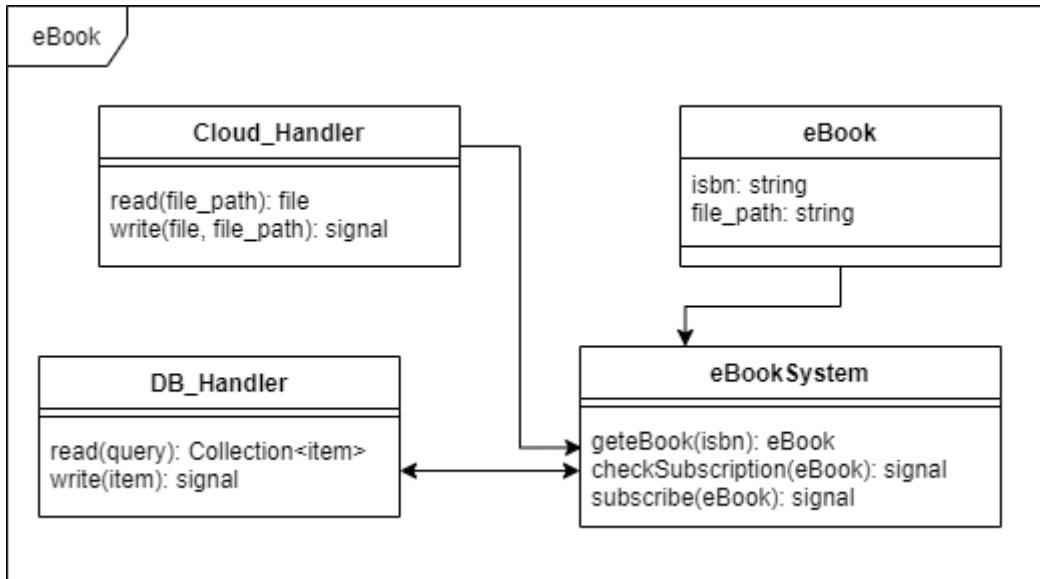


그림 41 Class Diagram – eBook System

eBook System Class 를 통해 사용자는 eBook 을 구독하고 열람할 수 있다. 사용자가 eBook 을 열람하는 경우 checkSubscription() 함수와 DB_Handler 를 통해 사용자의 구독정보를 확인하고 Cloud_Handler 를 통해 Cloud 저장소에 저장되어 있는 eBook 파일을 사용자에게 전달한다.

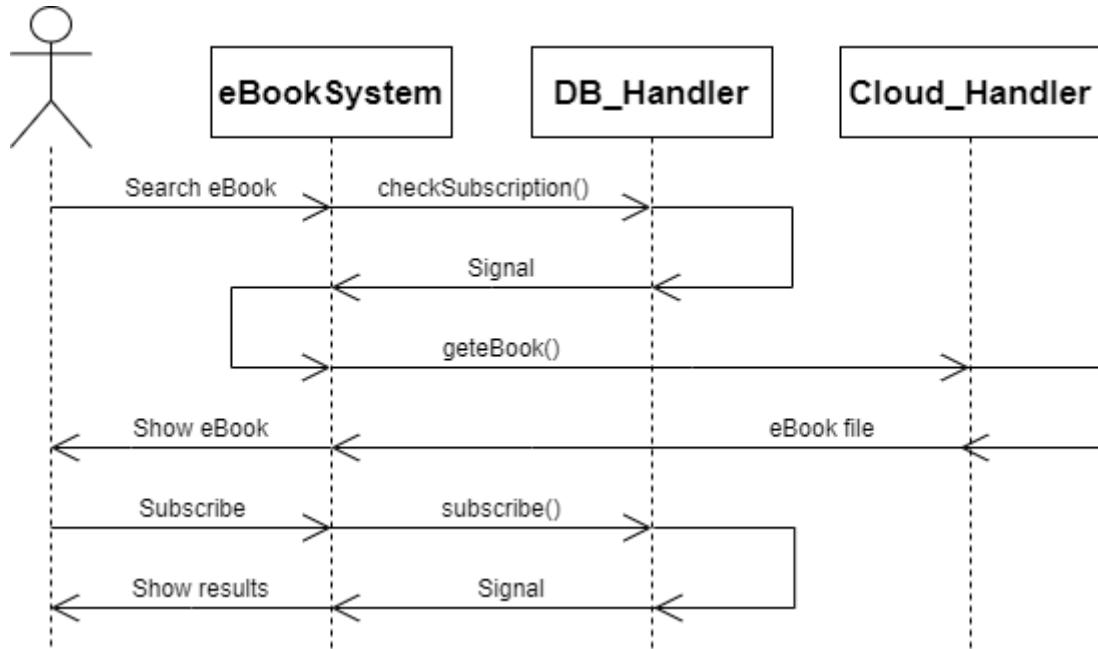


그림 42 Sequence Diagram – eBook System

5.3.7. Chat System

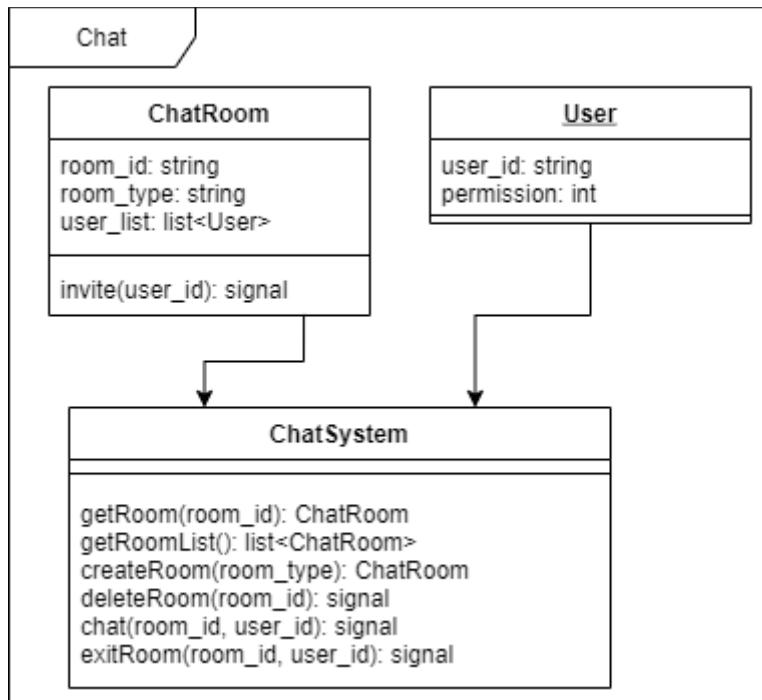


그림 43 Class Diagram – Chat System

Chat System Class 를 통해 채팅을 관리한다. 사용자는 createRoom() 함수를 통해 채팅방을 생성할 수 있고, invite() 함수를 통해 채팅방에 다른 사용자를 초대할 수 있다.

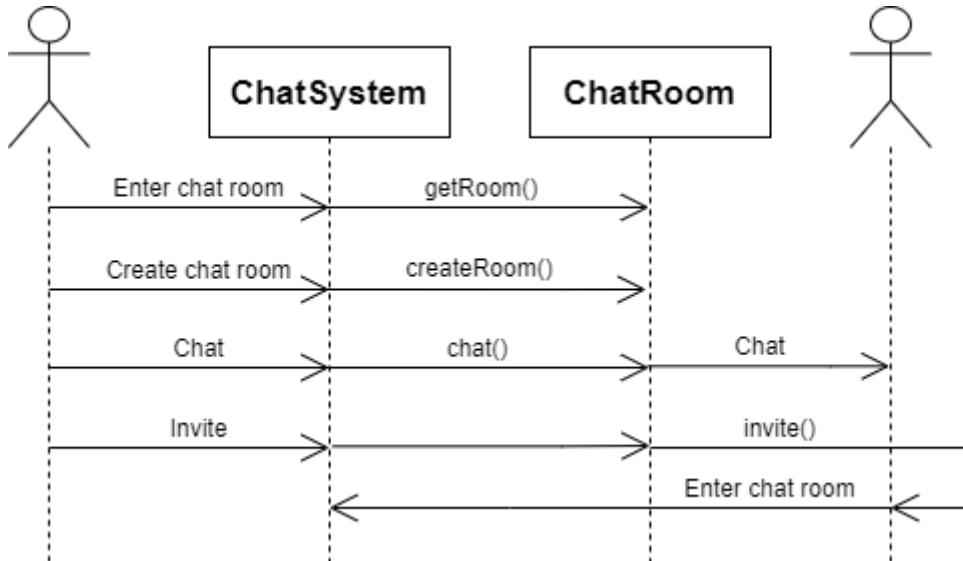


그림 44 Sequence Diagram – Chat System

5.3.8. Real-time Document System

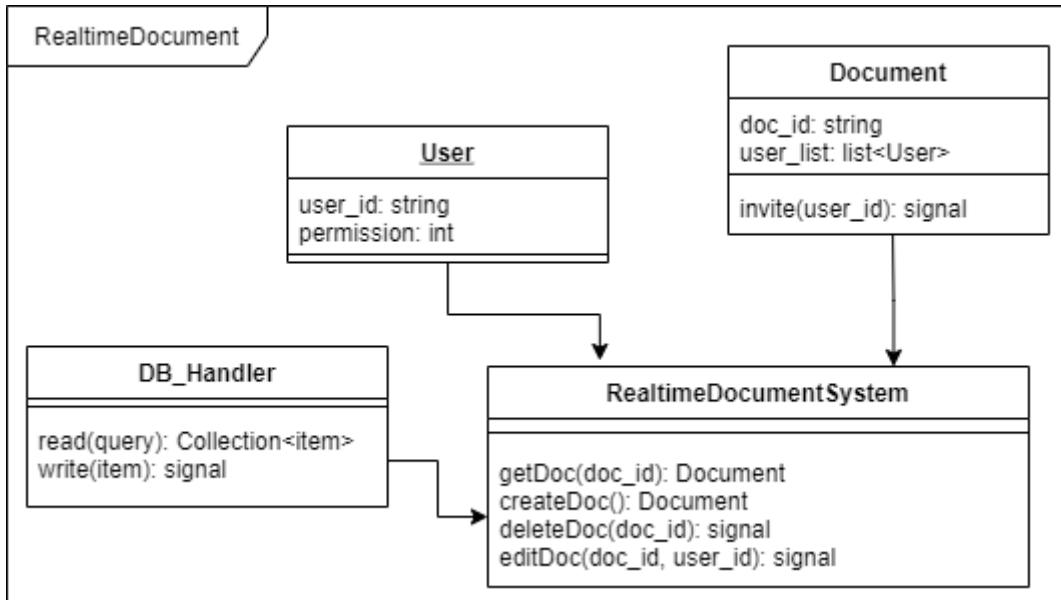


그림 45 Class Diagram – Real-time Document System

Real-time Document System 을 통해 실시간 공유 문서를 관리한다. 사용자가 실시간 공유 문서를 생성하면 createDoc() 함수와 DB_Handler 를 통해 데이터베이스에 문서 정보를 저장한다. 그리고 사용자가 문서를 수정하면 editDoc() 함수를 통해 수정사항이 처리된다. 각각 사용자는 퍼블리셔와 서브스크라이버로 간주되어 문서의 수정사항을 발신하고 수신한다. 문서의 수정사항은 GIT 을 통해 표현한다.

Project Highlight

Design Specification

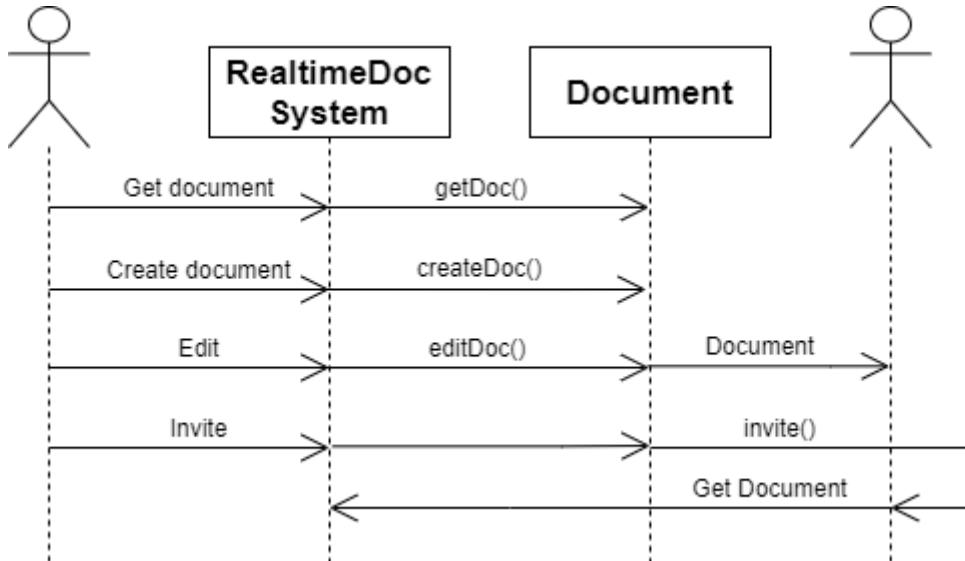


그림 46 Sequence Diagram – Real-time Document System

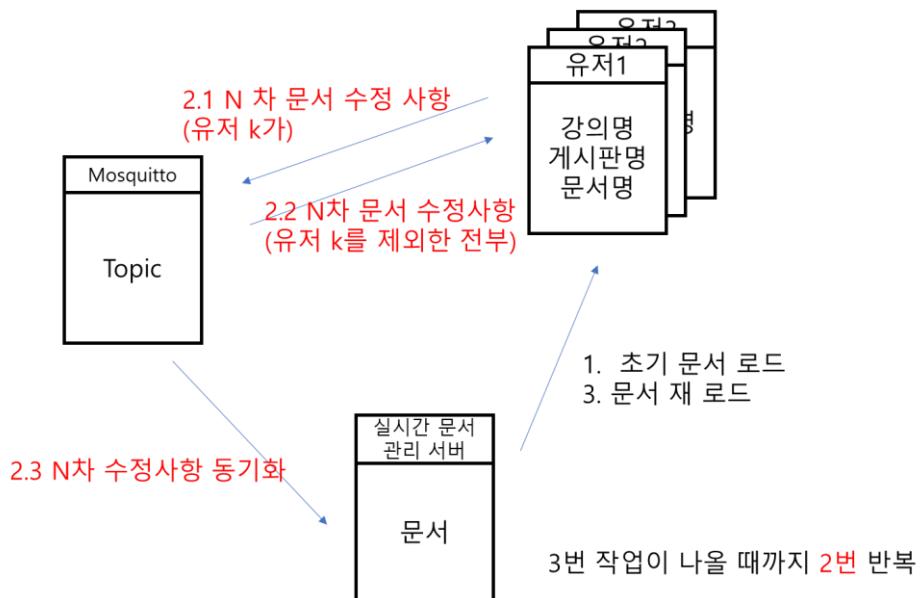


그림 47 Object oriented fuction

실시간 공유 문서는 위의 그림에 있는 방식으로 모든 유저가 문서들에게 계속 접근하지 않아도 같은 문서를 사용할 수 있다. 기존의 통신보다 훨씬 가벼운 mqtt 통신을 mosquitto 라는 브로커로 구현하였기 때문에 가능하다.

6. Protocol Design

6.1. Objective

프로토콜 디자인 파트는 Highlight application 과 서버 간의 통신이 어떻게 이루어지고, 통신이 이루어질 때 전달되는 방식과 형식에 대해서 설명한다.

6.2. 전달 형식

6.2.1. HTTP

HTTP 는 Hyper Text Transfer Protocol 의 두문자어로, 인터넷에서 데이터를 주고받을 수 있는 프로토콜이다. HTTP는 TCP/IP 단으로 구성되어 있으며, 요청(Request)와 응답(Response)로 통신을 한다. Client 가 형식에 맞춰서 요청을 하면, Server 는 알맞은 응답을 하는 형식이다. 요청을 할 때는 General, Headers, Body 로 이루어져 있다. General 에는 주소, Request 종류, Client 상태, 규칙 등이 들어간다. 헤더에는 요청에 대한 정보를 담고 있다. 그리고 BODY 에는 Header 에 맞게 정보를 보낼 수 있다. 이는 JSON 형태로 데이터를 주로 보낸다.

본 프로젝트에서는, HTTP 로 일반적인 통신을 다룬다

6.2.2. MQTT

MQTT 는 Message Queuing Telemetry Transport 의 약자로써, M2M(Machine to Machine) 특징을 가져, 디바이스 사이의 가벼운 통신 프로토콜이다. MQTT 는 HTTP, TCP 등의 통신과 같이 클라이언트–서버 구조로 이루어지는 것이 아닌, Broker, Publisher, Subscriber 구조로 이루어진다. Publisher 는 Topic 을 발행(publish)하고, Subscriber 는 Topic 에 구독(subscribe)합니다. Broker 는 이들을 중계하는 역할을 하며, 단일 Topic 에 여러 Subscriber 가 구독할 수 있기 때문에, 1:N 통신 구축에도 매우 유용하다.

본 프로젝트에서는 MQTT 로 채팅 어플리케이션과, 실시간 공유 문서를 제작할 때 사용한다.

6.3. Authentication

6.3.1. 로그인 및 회원가입

로그인 및 회원가입은 유저가 Highlight 앱을 이용하기 위해 가장 먼저 진행해야 되는 프로세스이다. 한 번 로그인을 하면, 로그아웃을 하기 전까지 계속 로그인이 유지되며, 로그인을 했을 때, Token 을 받아 유저 인증이 필요한 곳이면 지속적으로 Token 을 통해서 인증한다.

6.3.1.1. 회원가입

HTTP request

POST {{url}}/v1/user/registration

Request body

Parameters

Name	이름 (string)
ID	아이디
Password	비밀번호 (특수문자 포함 8자 이상) (string)
StudentID	학번 10자리 (string)
Type	사용자 유형 (0: 관리자, 1: 강의자, 2: 학생) (integer)
Phonenum	폰 번호 (string)
Address	주소 (string)

JSON representation

```
{
  "registration": {
    "Name": "홍길동",
    "ID": "hongildong",
    "Password": "ab2cd145!!",
    "StudentID": "2021123456",
    "Type": 2,
    "Phonenum": "01071585225",
    "Address" : "수원시 율전동"
  }
}
```

Response body

Parameters

Code	회원가입 성공 여부 (1000: 성공, 1001: 양식 불일치, 1002: 이미 존재하는 아이디)
Authorization	회원 고유 번호

JSON representation

```
{
  "code" : "1000",
  "Authorization" : "da123sdasvc"
}
```

6.3.1.2. 로그인

로그인을 완료하면, Token 을 받게 되고, 이 토큰을 통해서 Highlight 어플리케이션 내부에서 로그인을 유지한다.

HTTP request

POST {{url}}/v1/user/login

Request body

Parameters

ID	아이디
Password	비밀번호 (특수문자 포함 8자 이상) (string)

JSON representation

```
{
  "login": {
    "ID" : "hongildong",
    "Password" : "ab2cd145!!"
  }
}
```

Response body

Parameters

Code	로그인 성공 여부 (1000: 성공, 1001: 비밀번호 불일치 1002: 존재하지 않는 회원)
Token	로그인 성공 시 부여하는 고유한 key (session-token)

JSON representation

```
{
  code : "1000",
  token : "29da32b4c620b32a"
}
```

6.3.1.3. 로그아웃

허위로 로그아웃이 되는 것을 막기 위해서 Token 을 보내서 로그아웃 한다. 로그인과 로그아웃 로그들이 서버에 남기 때문에 해킹 같은 문제에 대비한다.

HTTP request

DEL {{url}}/v1/user/logout

Request body

Parameters

ID	아이디 (string)
Token	로그인 성공 시 부여받은 고유한 key (session-token)
Time	로그아웃 시간

JSON representation

```
{
  "logout": {
    "id" : "hongildong",
    "token" : "29da32b4c620b32a",
    "time" : 2021-01-11 17:04:02
  }
}
```

Response body

Parameters

Code	로그아웃 성공 여부 (integer) (1000: 성공)
------	------------------------------------

JSON representation

```
{
  "code" : "1000",
}
```

6.3.2. 개인정보 수정

6.3.2.1. 개인정보 불러오기

서버에 수정 로그를 남기고, 복구를 할 때 사용하기 위해서 유저의 개인정보 수정 시간을 보내준다.
0| 시작은 Time 형식으로 해도 되지만 아스키 타입으로 진행해도 무방하다.

HTTP request

GET {{url}}/v1/user/list

Request body

Parameters

ID	아이디 (string)
startTime	회원정보 조회 시작 시간 (time)

JSON representation

```
{
  "list": [
    {
      "ID" : "hongildong",
      "StartTime" : 2021-01-12 21:25:22,
    }
  ]
}
```

Response body

Parameters

Code	회원정보 조회 성공 여부 (integer) (1000: 성공, 1001: 실패, 1002: 조회 데이터 없음)
Result	조회 결과 리스트 (SQL)

JSON representation

```
{
  "Code" : "1000",
  "Result" : {
    ...
    ...
    ...
  }
}
```

학번, 생년월일, 등등

6.3.2.2. 개인정보 수정하기

수정한 내용들은 서버에서 무결성 검사가 완료된 다음에 수정이 반영된다. 수정이 반영된다면, 종료 시간을 서버에 기록해서 개인정보의 수정 로그를 남길 수 있게 한다.

HTTP request**POST** {{url}}/v1/user/update**Request body****Parameters**

ID	아이디
Password	비밀번호 (특수문자 포함 8자 이상) (string)
Newpassword	
Type	사용자 유형 (0: 관리자, 1: 강의자, 2: 학생) (integer)
Phonenum	폰 번호 (string)
Address	주소 (string)
Photo	프로필사진
endTime	회원정보 조회 종료 시간 (time)

JSON representation

```
{
  "update": [
    {
      "ID": "hongildong",
      "password": "ab2cd145!!",
      "newpassword": "qwer1234!!",
      "type": 0,
      "Phonenum": "01012345678",
      "address": "경기도 수원시 서부로~~"
      "photo": "~~",
      "endTime": "2021-01-12 21:25:22"
    }
  ]
}
```

Response body**Parameters**

Code	회원정보 수정 성공 여부 (1000: 성공, 1001: 비밀번호 불일치, 1002: 저장 실패 알림)
Message	에러시 에러 메세지

JSON representation

```
{
  "Code": "1000",
  "Message": "null"
}
```

6.3.3. 맞춤형 알림

맞춤형 알림 프로토콜은 프론트엔드에서 진행된 알림 제작 프로그램을 통해서 만들어진 알림들을 전송하는 프로토콜이다. 이 알림 파일은 html 상으로 인코딩 되어서 http 프로토콜을 통해서 들어가게 된다.

6.3.3.1. 맞춤형 알림 불러오기

HTTP request**get** {{url}}/v1/alarmload**Request body****Parameters**

userID	유저 아이디
--------	--------

JSON representation

```
{
  "alarmload": [
    {
      "UserID": "hongildong"
    }
  ]
}
```

Response body**Parameters**

code	승인 여부(100 승인, 101 권한 x 승인불가)
html	Alarm file

JSON representation

```
{
  "code": 101,
  "html": {
    "html infomation"
  }
}
```

6.3.3.2. 맞춤형 알림 만들기

HTTP request

post {{url}}/v1/alarmmake

Request body

Parameters

userID	유저 아이디
html	맞춤형 알림 html

JSON representation

```
{
  "alarmmake": [
    {
      "UserID": "hongildong",
      "html": {
        "html infomation"
      }
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
alarmNum	알람번호

JSON representation

```
{
  "code": 101,
  "alarmNum": 3
}
```

6.3.3.3. 맞춤형 알림 수정하기

HTTP request

post {{url}}/v1/alramMod

Request body

Parameters

userID	유저 아이디
alarmNum	강의번호
html	맞춤형 알림 html

JSON representation

```
{
  "alarmMod": [
    {
      "UserID": "hongildong",
      "alarm": 3,
      "html": {
        "html infomation"
      }
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  "code": 101
}
```

6.3.3.4. 맞춤형 알림 목록 보기

HTTP request

```
get {{url}}/v1/alramlist
```

Request body

Parameters

userID	유저 아이디
--------	--------

JSON representation

```
{
  "alarmlist": [
    {
      "UserID": "hongildong"
    }
  ]
}
```

Response body

Parameters

code	성공여부 (100 승인, 101 승인 거부(권한 x))
alarmNums	알람 개수
alarms	clip 리스트(아래 5개로 이루어진 리스트)
Alramtitle	클립 제목
alramnum	클립 번호

JSON representation

```
{
  "code": 100,
  "alarmNum": 5,
  "clips": [
    {
      "clipTitle": "소프트웨어 특강 과제 알림",
      "clipNum": 1
    },
    ...
  ]
}
```

6.3.4. 게시글 업로드 및 열람

기본적으로 강의 컨텐츠, 강의 클립, 강의 노트 등은 게시글의 형식을 따른다. 약간의 수정 사항이 있지만, 댓글 기능의 경우 위의 모든 기능 들에서 동일하게 작동되기 때문에 댓글 프로토콜은 1.3.4 절에서만 다룬다.

6.3.4.1. 게시글 목록 불러오기

HTTP request

```
get {{url}}/v1/lecture/list
```

Request body

Parameters

lectureID	강의 아이디
noticeID	게시판 아이디
pageNum	페이지번호

JSON representation

```
{
  "list": [
    {
      "lectureID": "32d123d",
      "noticeID": 3,
      "pageNum": 1
    }
  ]
}
```

Response body

Parameters

code	성공여부 (100 승인, 101 승인 거부(권한 x))
totalPageNum	총 페이지 숫자
elementNum	현재 페이지 게시글 개수
currrentPage	현재 페이지 숫자
posts	post 리스트(아래 4개로 이루어진 리스트)
posttitle	게시글 제목
postnum	게시글 번호
postviews	게시글 조회수
postcommentnum	게시글 댓글 개수
postNotice	공지사항 여부 (공지사항 1, 일반 게시글 0)

JSON representation

```
{
  "code": 100,
  "totalPageNum": 2,
  "elementNum": 20,
  "currrentPage": 1,
  "posts": [
    {
      "postTitle": "중요 공지사항입니다.",
      "postNum": 20132,
      "postViews": 41,
      "postCommentNum": 3,
      "postNotice": 1
    },
    ...
  ]
}
```

6.3.4.2. 게시글 불러오기

HTTP request

```
get {{url}}/v1/lecture/loadpost
```

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
postnum	게시글 번호

JSON representation

```
{
  "loadpost": {
    "UserID" : hongildong
    "lectureID": 32d123d
    "noticeID" : 3
    "postnum" : 1
  }
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
html	게시글 html

JSON representation

```
{
  code : 101
  html : {
    html information
  }
}
```

6.3.4.3. 게시글 업로드

HTTP request

```
post {{url}}/v1/lecture/uploadpost
```

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
html	게시글 html

JSON representation

```
{
  "uploadpost": {
    "UserID" : hongildong
    "lectureID": 32d123d
    "noticeID" : 3
    "html" : {
      html information
    }
  }
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  code : 101
}
```

6.3.4.4. 게시글 수정

기본적으로 게시글 수정은 게시글 불러오기 이후에 일어나는 프로토콜이다. 즉 수정 전 데이터를 유저 어플리케이션에서 가지고 있고, 권한 역시 가지고 있고 Token으로 가지고 있기 때문에 수정이 가능한 사람을 프로토콜을 사용하지 않고 유저 어플리케이션에서 확인할 수 있다. 즉 수정 내용 업로드 프로토콜만 존재한다.

HTTP request

```
post {{url}}/v1/lecture/modost
```

Request body**Parameters**

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
postnum	게시글 번호
html	게시글 html

JSON representation

```
{
  "modpost": [
    {
      "UserID": "hongildong",
      "lectureID": "32d123d",
      "noticeID": 3,
      "html": {
        "html information"
      }
    }
  ]
}
```

Response body**Parameters**

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  "code": 101
}
```

6.3.4.5. 게시글 삭제

HTTP request

```
del {{url}}/v1/lecture/delpost
```

Request body**Parameters**

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
postnum	게시글 번호

JSON representation

```
{
  "delpost": [
    {
      "UserID": "hongildong",
      "lectureID": "32d123d",
      "noticeID": 3,
      "postnum": 1
    }
  ]
}
```

Response body**Parameters**

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  "code": 101
}
```

6.3.4.6. 댓글 작성

HTTP request

post {{url}}/v1/lecture/commentwrite

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
postnum	게시글 번호
text	댓글 내용

JSON representation

```
{
  "commentwrite ":
  {
    "UserID" : hongildong
    "lectureID": 32d123d
    "noticeID" : 3
    "postnum" : 1
    "Text" : "교수님 질문이 있습니다."
  }
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  code : 101
}
```

6.3.4.7. 댓글 수정 신청

댓글은 예외적으로 댓글을 달은 사람만이 수정해야 되기 때문에 댓글 수정 신청 프로토콜이 존재한다.

HTTP request

get {{url}}/v1/lecture/commentwriteinit

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
postnum	게시글 번호
commentnum	댓글 번호

JSON representation

```
{
  "commentwriteinit":
  {
    "UserID" : hongildong
    "lectureID": 32d123d
    "noticeID" : 3
    "postnum" : 1
    "commentnum" : 1
  }
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  code : 100
}
```

6.3.4.8. 댓글 수정

HTTP request

post {{url}}/v1/lecture/commentmod

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
postnum	게시글 번호
commentnum	댓글 번호
text	댓글 내용

JSON representation

```
{
  "commentmod": {
    "UserID": "hongildong",
    "lectureID": "32d123d",
    "noticeID": 3,
    "postnum": 1,
    "commentnum": 1,
    "Text": "교수님 여쭤볼게 있습니다."
  }
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  "code": 100
}
```

6.3.4.9. 댓글 삭제

HTTP request

del {{url}}/v1/lecture/delcomment

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
postnum	게시글 번호
commentnum	댓글 번호

JSON representation

```
{
  "delcomment": {
    "UserID": "hongildong",
    "lectureID": "32d123d",
    "noticeID": 3,
    "postnum": 1,
    "commentnum": 1
  }
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  "code": 101
}
```

6.3.5. 강의 컨텐츠 업로드 및 열람

강의 컨텐츠는 기본적으로 게시글에 동영상이 직접 첨부된 형식이다. File 형식으로 게시글에 첨부되는 것이 아니고, 직접 화면상에서 실행되며, Highlight 의 주요 컨텐츠이기 때문에 다른 incoding 된 동영상을 프로토콜에서 따로 다루며, 이는 DB에서도 예외적으로 강의 컨텐츠 동영상 DB에 저장된다.

6.3.5.1. 강의 컨텐츠 불러오기

HTTP request

```
get {{url}}/v1/lecture/loadvideo
```

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
videonum	강의 컨텐츠 번호

JSON representation

```
{
  "loadpost": [
    {
      "UserID" : hongildong,
      "lectureID": 32d123d,
      "noticeID" : 3,
      "videonum" : 1
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
html	Html 코드

JSON representation

```
{
  "code" : 101,
  "html" : {
    "html infomation"
  }
}
```

6.3.5.2. 강의 컨텐츠 업로드

HTTP request

```
post {{url}}/v1/lecture/uploadvideo
```

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 번호
videoinfo	강의 컨텐츠 동영상 정보
html	강의 컨텐츠 html

JSON representation

```
{
  "uploadvideo": [
    {
      "UserID" : hongildong,
      "lectureID": 32d123d,
      "noticeID" : 1,
      "videoinfo" : {
        "incoded video"
      },
      "html" : {
        "html infomation"
      }
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
videoNum	강의 컨텐츠 번호

JSON representation

```
{
  "code" : 101,
  "videoNum" : 3
}
```

6.3.5.3. 강의 컨텐츠 수정

HTTP request

post {{url}}/v1/lecture/modost

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
videonum	강의 컨텐츠 번호
videoinfo	강의 컨텐츠 동영상 정보
html	강의 컨텐츠 html

JSON representation

```
{
  "modpost": [
    {
      "UserID": "hongildong",
      "lectureID": "32d123d",
      "noticeID": 1,
      "videoinfo": {
        "incoded video"
      },
      "html": {
        "html infomation"
      }
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  "code": 101
}
```

6.3.6. 실시간 공유 문서

실시간 공유 문서는 수정 시 지속적인 mqtt 통신을 통해서 동시에 사용하는 모든 유저의 어플리케이션에서 수정사항을 Real Time으로 반영할 수 있도록 디자인하였다. 통신의 부하를 줄이기 위해서 git에서 사용하는 수정 사항 형식을 반영했다.

6.3.6.1. 실시간 공유 문서 생성

HTTP request

post {{url}}/v1/lecture/RTPostCreate

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
title	문서 제목
userInfo	초기 유저 범위 list

JSON representation

```
{
  "RTPostCreate": [
    {
      "UserID": "hongildong",
      "lectureID": "32d123d",
      "noticeID": 1,
      "title": "공유문서 1",
      "userInfo": [
        { "user": "leesorong", "team": 32 }
      ]
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
RTPostNum	실시간 공유문서 번호

JSON representation

```
{
  "code": 101,
  "RTPostNum": 3
}
```

6.3.6.2. 실시간 공유 문서 추가

HTTP request

post {{url}}/v1/lecture/RTPostUser

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
RTPostNum	실시간 공유문서 번호
userInfo	추가 유저 범위 list

JSON representation

```
{
  "RTPostUser": [
    {
      "UserID": "hongildong",
      "lectureID": "32d123d",
      "noticeID": 1,
      "RTPostNum": 3,
      "userInfo": [
        { "user": "choonhang"}, { "team": 45 }
      ]
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  "code": 101
}
```

6.3.6.3. 실시간 공유 문서 삭제

HTTP request

del {{url}}/v1/lecture/RTPostDel

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
RTPostNum	실시간 공유문서 번호

JSON representation

```
{
  "RTPostDel": [
    {
      "UserID": "hongildong",
      "lectureID": "32d123d",
      "noticeID": 1,
      "RTPostNum": 3
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  "code": 101
}
```

6.3.6.4. 실시간 공유 문서 불러오기

HTTP request

```
get {{url}}/v1/lecture/RTPostLoad
```

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
RTPostNum	실시간 공유문서 번호

JSON representation

```
{
  "RTPostDel": [
    {
      "UserID": "hongildong",
      "lectureID": "32d123d",
      "noticeID": 1,
      "RTPostNum": 3
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
html	Html 코드

JSON representation

```
{
  "code": 101,
  "html": {
    "html infomation"
  }
}
```

6.3.6.5. 실시간 공유 문서 사용

git에서 부분적인 수정사항만 commit 하듯이 mqtt에서도 수정사항만 message로 보낸다.

MQTT Topic

```
{{url}}/$lecturenum/$noticenum/$RTPostNum
```

Message

```
git information about modify
```

6.3.7. 강의 클립화

강의 클립화 기능은, 강의 컨텐츠의 동영상을 편집한 “클립 동영상”과 그 클립을 포함하고 있는 게시글로 이루어진다. 또한 강의 클립은 프론트 엔드의 클립 제작기를 통해서 이루어진다. 프로토콜에서는 강의를 불러와서 강의 동영상을 획득한 다음에, 클립 제작기로 2차 창작한 클립을 다시 통신으로 다루는 프로토콜에 대해서 다루고 있다.

6.3.7.1. 강의 불러오기

HTTP request

get {{url}}/v1/lecture/cliploadvideo

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
videonum	강의 컨텐츠 번호

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
videoInfo	비디오 내용
html	총 페이지 숫자

JSON representation

```
{
  "cliploadvideo": {
    "UserID" : hongildong,
    "lectureID": 32d123d,
    "noticeID" : 3,
    "videonum" : 1
  }
}
```

JSON representation

```
{
  "code" : 101,
  "videoinfo" : {
    incoded video
  },
  "html" : {
    html infomation
  }
}
```

6.3.7.2. 클립 올리기

HTTP request

post {{url}}/v1/lecture/clipupload

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
clipInfo	클립 내용
html	클립 html

JSON representation

```
{
  "uploadpost": {
    "UserID" : hongildong,
    "lectureID": 32d123d,
    "noticeID" : 3,
    "clipinfo" : {
      incoded video
    },
    "html" : {
      html infomation
    }
  }
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
clipNum	강의번호

JSON representation

```
{
  code : 101,
  clipNum : 3
}
```

6.3.7.3. 클립 수정

HTTP request

post {{url}}/v1/lecture/clipupload

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
clipNum	클립 번호
clipInfo	클립 내용
html	클립 html

JSON representation

```
{
  "uploadpost": {
    "UserID": "hongildong",
    "lectureID": "32d123d",
    "noticeID": 3,
    "clipNum": 3,
    "clipinfo": {
      "incoded video"
    },
    "html": {
      "html infomation"
    }
  }
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  "code": 101
}
```

6.3.7.4. 클립 목록보기

HTTP request

get {{url}}/v1/lecture/listClip

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
pageNum	페이지번호
clipRange	클립 범위(lecture ID or All)

JSON representation

```
{
  "list": {
    "UserID": "hongildong",
    "lectureID": "32d123d",
    "noticeID": 3,
    "pageNum": 1,
    "clipRange": "All"
  }
}
```

Response body

Parameters

code	성공여부 (100 승인, 101 승인 거부(권한 x))
totalPageNum	총 페이지 숫자
elementNum	현재 페이지 게시글 개수
currrentPage	현재 페이지 숫자
clips	clip 리스트(아래 5개로 이루어진 리스트)
cliptitle	클립 제목
Clipnum	클립 번호
Clipviews	클립 조회수
clipLikeNum	클립 좋아요 개수
clipFavorites	좋아요 여부

JSON representation

```
{
  "code": 100,
  "totalPageNum": 2,
  "elementNum": 20,
  "currrentPage": 1,
  "clips": [
    "clip": {
      "cliptitle": "1주차 중요 내용",
      "clipnum": 20132,
      "clipviews": 41,
      "clipLikeNum": 3,
      "clipFavorites": 1
    },
    ...
  ]
}
```

6.3.8. 강의 노트 공유

6.3.8.1. 강의 노트 올리기

HTTP request

post {{url}}/v1/lecture/LNUpload

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
html	강의노트 html

JSON representation

```
{
  "LNUpload": [
    {
      "UserID": "hongildong",
      "lectureID": "32d123d",
      "noticeID": 3,
      "html": {
        "html infomation"
      }
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
LNum	강의노트 번호

JSON representation

```
{
  "code": 101,
  "LNum": 3
}
```

6.3.8.2. 강의 노트 수정

HTTP request

post {{url}}/v1/lecture/LNmod

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
LNum	강의노트 번호
html	강의노트 html

JSON representation

```
{
  "LNmod": [
    {
      "UserID": "hongildong",
      "lectureID": "32d123d",
      "noticeID": 3,
      "LNum": 3,
      "html": {
        "html infomation"
      }
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)

JSON representation

```
{
  "code": 101
}
```

6.3.8.3. 강의 노트 목록보기

HTTP request

```
get {{url}}/v1/lecture/listLN
```

Request body

Parameters

userID	유저 아이디
lectureID	강의 아이디
noticeID	게시판 아이디
pageNum	페이지번호
LNRange	클립 범위(lecture ID or All)

JSON representation

```
{
  "listLN": [
    {
      "UserID": "hongildong",
      "lectureID": "32d123d",
      "noticeID": 3,
      "pageNum": 1,
      "LNRange": "All"
    }
  ]
}
```

Response body

Parameters

code	성공여부 (100 승인, 101 승인 거부(권한 x))
totalPageNum	총 페이지 수
elementNum	현재 페이지 개수
surrentPage	현재 페이지 수
NLs	clip 리스트(아래 5개로 이루어진 리스트)
LNTtitle	클립 제목
LNNum	클립 번호
LNViews	클립 조회수
LNLikeNum	클립 좋아요 개수
LNFavorites	좋아요 여부

JSON representation

```
{
  "code": 100,
  "totalPageNum": 2,
  "elementNum": 20,
  "surrentPage": 1,
  "clips": [
    {
      "LNTtitle": "1주차 중요 내용",
      "LNNum": 20132,
      "LNViews": 41,
      "LNLikeNum": 3,
      "LNFavorites": 1
    },
    ...
  ]
}
```

6.3.9. eBook

eBook은 한번의 인증을 통하여서 인증이 완료된 사람들만 아웃소싱 된 eBook 사이트를 제공해준다. eBook 사이트에서 유저는 필요한 책들을 읽을 수 있다.

HTTP request

```
GET {{url}}/v1/user/eBookRegi
```

Request body

Parameters

ID	아이디 (string)
----	--------------

JSON representation

```
{
  "eBookRegi": [
    {
      "id": "hongildong"
    }
  ]
}
```

Response body

Parameters

Code	eBook 구독여부 (1000: 등록, 1001: 등록X, 1002: 조회 데이터 없음)
html	등록되어 있을시 eBook html

JSON representation

```
{
  "code": "1000",
  "html": {
    "html infomation"
  }
}
```

6.3.10. 채팅

6.3.10.1. 채팅 방 생성

HTTP request

post {{url}}/v1/ChatCreate

Request body

Parameters

userID	유저 아이디
title	채팅 제목
userInfo	초기 유저 범위 list

JSON representation

```
{
  "RTPostCreate": [
    {
      "UserID": "hongildong",
      "title": "나만의 채팅1",
      "userInfo": [
        { "user": "leesorong"}, { "team": 32 }
      ]
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
ChatNum	채팅 방 번호

JSON representation

```
{
  "code": 101
  "ChatNum": 3
}
```

6.3.10.2. 채팅 방 유저 추가

HTTP request

post {{url}}/v1/lecture/RTPostUser

Request body

Parameters

userID	유저 아이디
ChatNum	채팅 방 번호
userInfo	추가 유저 범위 list

JSON representation

```
{
  "RTPostUser": [
    {
      "UserID": "hongildong",
      "ChatNum": 3,
      "userInfo": [
        { "user": "choonhang"}, { "team": 45 }
      ]
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  "code": 101
}
```

6.3.10.3. 채팅 나가기

모든 유저가 채팅방에서 나가면 채팅방은 자동으로 삭제되기에 채팅 방 나가기 프로토콜은 없다.

Project Highlight

Design Specification

HTTP request

```
post {{url}}/v1/lecture/RTPostUser
```

Request body

Parameters

userID	유저 아이디
ChatNum	채팅 방 번호

JSON representation

```
{
  "RTPostUser": [
    {
      "UserID": "hongildong",
      "ChatNum": 3
    }
  ]
}
```

Response body

Parameters

code	승인 여부(100 승인, 101 권한 x 승인불가)
------	-------------------------------

JSON representation

```
{
  "code": 101
}
```

6.3.10.4. 채팅 목록보기

HTTP request

```
get {{url}}/v1/lecture/listLN
```

Request body

Parameters

userID	유저 아이디
--------	--------

JSON representation

```
{
  "listLN": [
    {
      "UserID": "hongildong"
    }
  ]
}
```

Response body

Parameters

code	성공여부 (100 승인, 101 승인 거부(권한 x))
ChatNums	채팅방 개수
Chats	채팅 방 리스트(아래 2개로 이루어진 리스트)
ChatTitle	채팅 방 제목
ChatNum	채팅 방 번호

JSON representation

```
{
  "code": 100,
  "ChatNums": 34,
  "Chats": [
    {
      "ChatTitle": "소공개 팀플 채팅방",
      "ChatNum": 32
    },
    ...
  ]
}
```

6.3.10.5. 채팅

채팅은 일반적인 채팅(Text)와, 특수 채팅(file 첨부한 text 와 voice chat)으로 이루어져 있다. 기본적으로 채팅은 mqtt 형식으로 이루어진다. 이 mqtt 의 message에는 chat에 대한 정보가 들어가 있고, chat의 json 형식을 분석함으로써 다른 유저들은 알맞은 행동을 한다.

6.3.10.5.1. 일반 채팅(Text)

일반적은 채팅은 mqtt에서 받은 text 메시지를 다른 유저들에게 직접적으로 보여준다.

MQTT Topic

{{url}}/\$ChatNum

Message

```
{
  "user" : $userID
  "chat" : {
    "text" : text that chat
  }
}
```

6.3.10.5.2. 특수 채팅(file) – MQTT

File 이 첨부된 채팅은, mqtt 에서 file 에 대한 정보를 주고, 이를 http 를 통해서 서버로 보낸다. 또한 message 를 받은 유저도 file 다운로드시 http 정보를 통해서 file 을 서버에서 받아온다.

MQTT Topic

{{url}}/\$ChatNum

Message

```
{
  "user" : $userID
  "chat" : {
    "text" : text that chat
    "file" : $fileNo
  }
}
```

6.3.10.5.3. 특수채팅(file) – HTTP Send Request**File Http(send)****post** {{url}}/v1/chat/file**Request body****Parameters**

fileNo	메시지에서 보낸 file No
file	파일

JSON representation

```
{
  "file": {
    "fileNo" : "425dag"
    "file" : {
      "file"
    }
  }
}
```

Response body**Parameters**

Code	파일전송 성공 여부
------	------------

JSON representation

```
{
  "code" : "1000",
}
```

6.3.10.5.4. 특수채팅(file) – HTTP Receive Request

File Http(receive)

get {{url}}/v1/chat/file

Request body

Parameters

fileNo	메시지에서 보낸 file No
--------	------------------

JSON representation

```
{
  "logout": [
    {
      "fileNo": "425dag",
      "file": {
        "file"
      }
    }
  ]
}
```

Response body

Parameters

Code	파일전송 성공 여부
file	파일

JSON representation

```
{
  "code": "1000",
  "file": {
    "file"
  }
}
```

6.3.10.5.5. 특수채팅 Voice

Voice 채팅의 경우 FFMPEG 의 live audio stream 기능을 사용하고, voice chat 을 시작하면 이에 대한 주소를 주어 유저들이 streaming 에 참여할 수 있도록 정보를 보낸다.

MQTT Topic

{{url}}/\$ChatNum

Message

```
{
  "user": $userID
  "chat": {
    "voice": voice chat address
  }
}
```

6.3.11. 권한부여

HTTP request

POST {{url}}/v1/user/endue

Request body

Parameters

ID	아이디(부여하는 사람의)
Password	비밀번호(부여하는 사람의)
TargetID	아이디 (부여 당하는 사람의)
Type	사용자 유형 (0: 관리자, 1: 강의자, 2: 학생) (integer)

JSON representation

```
{
  "user": {
    "ID": "hongildong",
    "password": "ab2cd145!!",
    "TargetID": "leesunsin",
    "type": 0
  }
}
```

Response body

Parameters

Code	권한 부여 성공 여부 (1000: 성공, 1001: 부여자 권한 없음, 1002: 이미 부여된 권한임)
Message	에러 시 에러 메세지

JSON representation

```
{
  "code": "1000",
  "Message": "null"
}
```

7. Database design

7.1. Objective

7 번 섹션에서는 데이터베이스의 구성과 그 상세를 보여준다. 이 섹션은 데이터베이스의 E-R 다이어그램, 객체들, 관계 스키마, 그리고 데이터베이스의 실제 구현된 SQL DLL을 보여준다. SQL로 표현될 수 없는 데이터는 표현하지 않았다.

7.2. E-R diagram

데이터는 크게 4 개의 엔티티로 구성된다. 유저, 강의, 강의노트, eBook 으로 총 데이터베이스가 구성된다. 각 엔티티와 관계는 <그림 1>의 E-R 다이어그램으로 표현된다

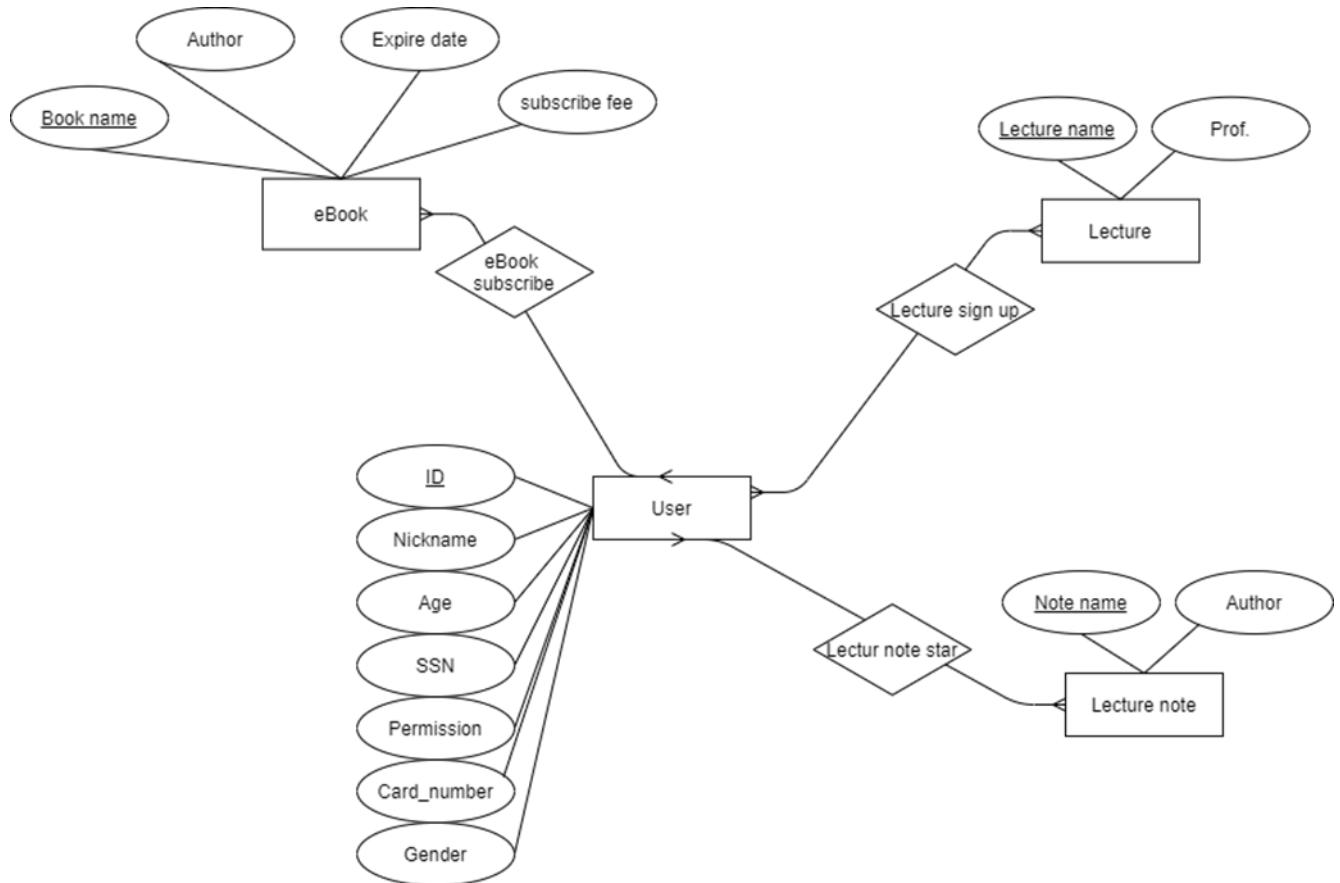


그림 48 E-R 다이어그램

7.2.1. Entities

7.2.1.1. User

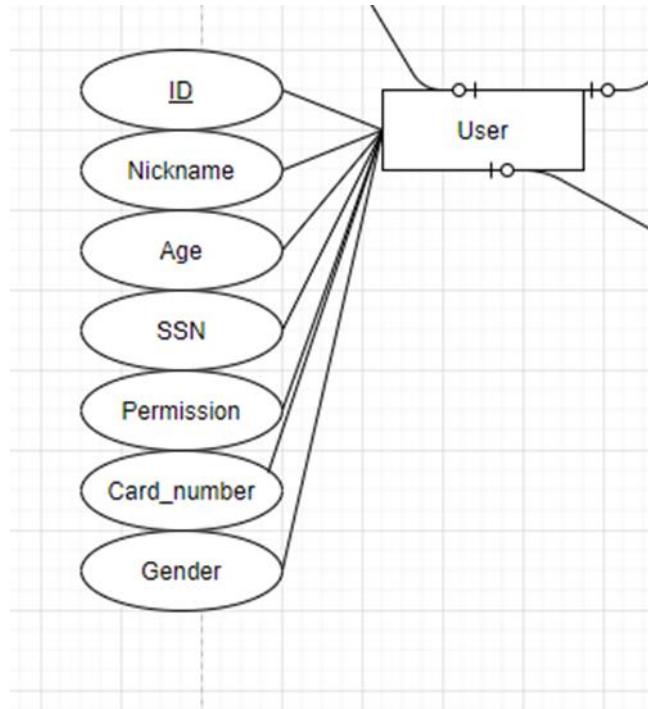


그림 49 User Entity

이 엔티티는 유저에 관련된 정보를 포함한다. 아이디, 별명, 나이, 주민등록번호, 권한, 결제를 위한 신용카드 정보 및 성별을 수집한다. 이 중, 아이디는 기본키로 지정되며 중복될 수 없다. Lecture, eBook, Lecture note 와 다대다 관계를 맺는다.

7.2.1.2. Lecture

이 엔티티는 강의와 관련된 정보를 포함한다. 강의명과 강의를 진행하는 강의자의 아이디를 포함하며, 강의자의 아이디는 유저 엔티티에서 외래키로 받아온다. 강의명을 기본키로 지정하며, 중복될 수 없다. User 와 다대다 관계를 맺는다.

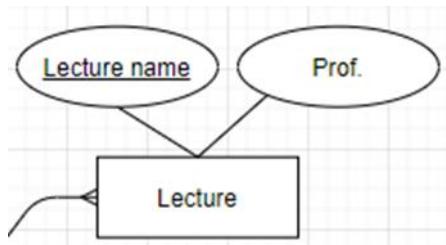


그림 50 Lecture Entity

7.2.1.3. eBook

이 엔티티는 eBook 과 관련된 정보를 포함한다. 책 이름을 기본키로 가지며, 저자, 만료 날짜, 구독료 등을 포함한다. User 와 다대다 관계를 맺는다.

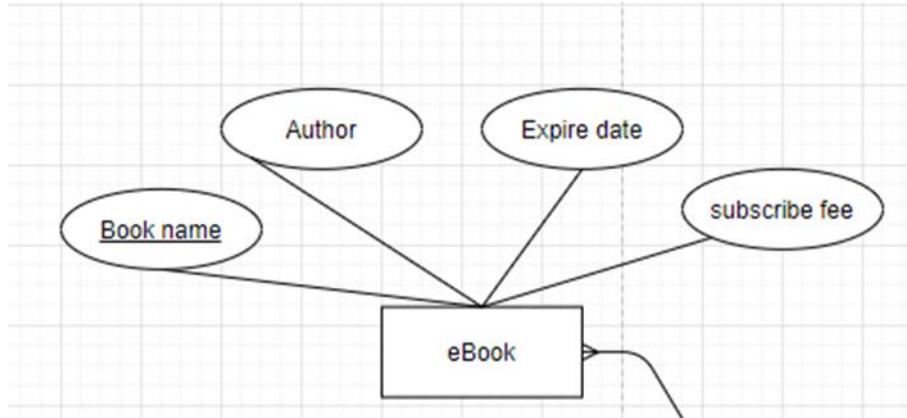


그림 51 eBook Entity

7.2.1.4. Lecture note

이 엔티티는 유저가 공유하는 강의 노트와 관련된 정보를 포함한다. 노트 이름과 노트를 등록한 유저의 아이디를 포함한다. 노트를 등록한 유저의 아이디는 유저 엔티티에서 외래키로 활용한다. 노트 이름을 기본키로 지정하며, 중복될 수 없다. User 와 다대다 관계를 맺는다.

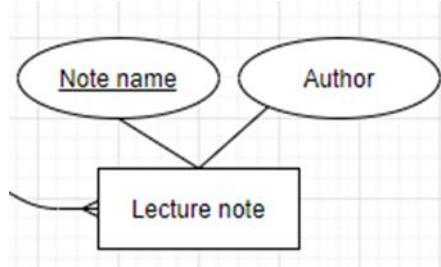


그림 52 Lecture Note Entity

7.3. Relational Schema

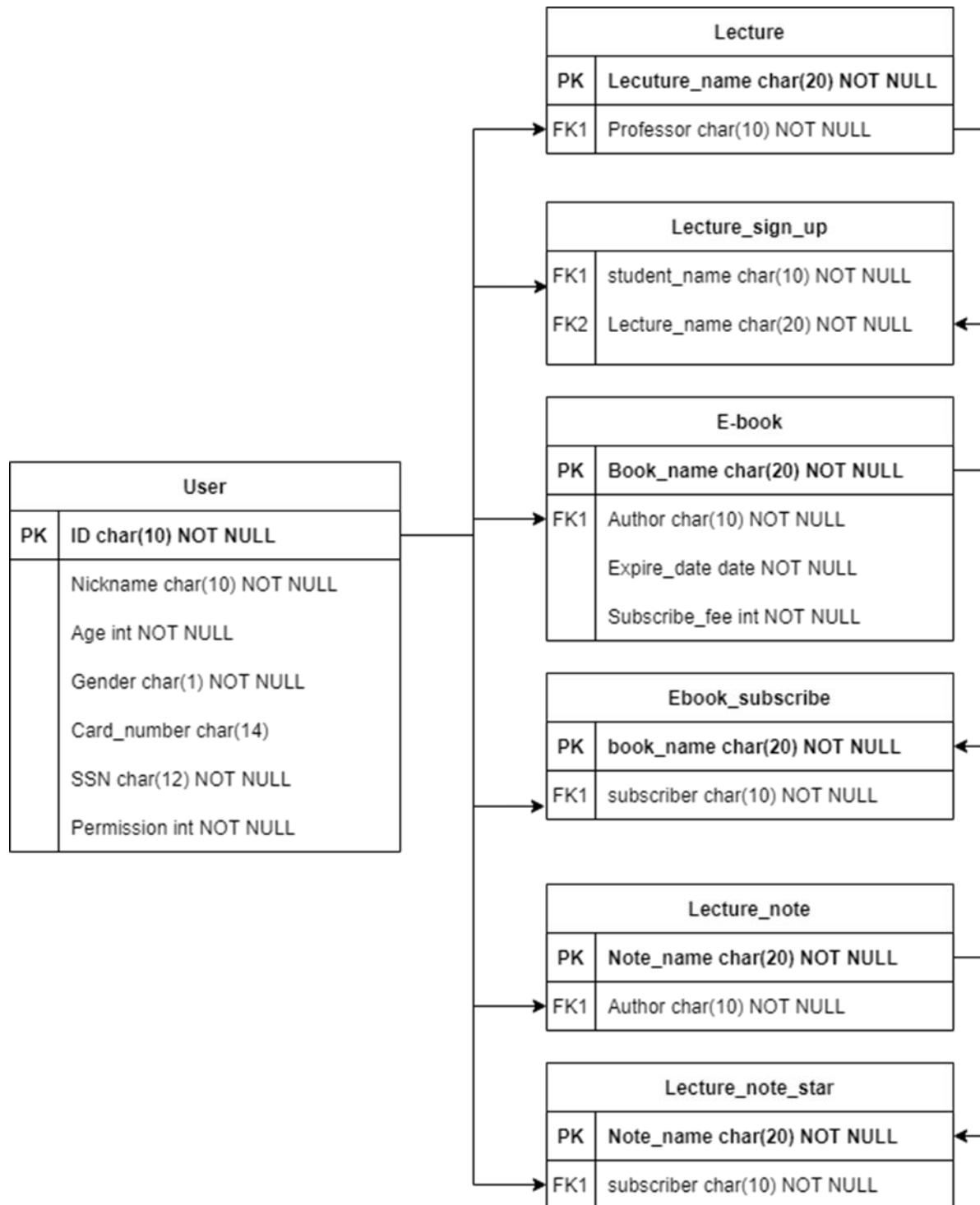


그림 53 Relational Schema

7.4. SQL DDL

7.4.1. User

```
CREATE TABLE User
(
    id CHAR(10) NOT NULL,
    nickname CHAR(10) NOT NULL,
    age INT NOT NULL,
    gender char(1) NOT NULL,
    card_number char(14),
    ssn char(12) NOT NULL,
    permission INT NOT NULL,
    PRIMARY KEY (id)
);
```

7.4.2. Lecture

```
CREATE TABLE Lecture
(
    lecture_name CHAR(20) NOT NULL,
    professor CHAR(10) NOT NULL,
    PRIMARY KEY (lecture_name),
    FOREIGN KEY (professor) REFERENCES
User(id)
);
```

7.4.3. Lecture sign up

```
CREATE TABLE Lecture_sign_up
(
    student_name CHAR(20) NOT NULL,
    lecture_name CHAR(10) NOT NULL,
    FOREIGN KEY (student_name)
REFERENCES User(id)
    FOREIGN KEY (lecture_name)
REFERENCES Lecture(lecture_name)
);
```

7.4.4. eBook

```
CREATE TABLE Ebook
(
    book_name CHAR(20) NOT NULL,
    author CHAR(10) NOT NULL,
    expire_date DATE NOT NULL,
    subscribe_fee INT NOT NULL,

    PRIMARY KEY (book_name),
    FOREIGN KEY (author) REFERENCES
User(id)
);
```

7.4.5. eBook subscribe

```
CREATE TABLE Ebook_subscribe
(
    book_name CHAR(20) NOT NULL,
    subscriber CHAR(10) NOT NULL,

    PRIMARY KEY (book_name)
    FOREIGN KEY (book_name)
    REFERENCES Ebook(book_name)
    FOREIGN KEY (subscriber) REFERENCES
User(id)
);
```

7.4.6. Lecture note

```
CREATE TABLE Lecture_note
(
    note_name CHAR(20) NOT NULL,
    author CHAR(10) NOT NULL,

    PRIMARY KEY (note_name),
    FOREIGN KEY (author) REFERENCES
User(id)
);
```

7.4.7. Lecture note star

```
CREATE TABLE Lecture_note_star
(
    note_name CHAR(20) NOT NULL,
    subscriber CHAR(10) NOT NULL,
    PRIMARY KEY (note_name),
    FOREIGN KEY (note_name) REFERENCES
    Lecture_note(note_name)
    FOREIGN KEY (author) REFERENCES User(id)
);
```

8. Testing Plan

8.1. 이 챕터에서는...

이 챕터에서는 개발, 배포, 사용자 부문의 3 가지 분류에 따라 시스템의 테스트에 대한 계획을 기술한다. 테스트는 시스템이 갖는 잠재적인 에러 및 제품의 약점을 찾아내고, Highlight 를 소비자에게 배포하는 데에 무결점성과 안정성을 보장하기에 아주 중요하다.

8.2. Testing Policy

8.2.1. Development Test

Development test 는 장래에 생길 여러 잠재적인 위협과 시간/금전적 비용을 경감하고, 넓은 범주에서 문제사항들을 예방하고 발견하는 전략으로써 진행된다. 이 단계에서 Highlight 는 아직 충분한 테스트를 거치지 못하였기 때문에 불안정할 수 있고, 구성요소들 간에 충돌이 발생할 수 있다. 따라서, 정적 코드 분석, 데이터 플로우 분석, 피어 코드 리뷰, 단위 테스팅 등의 여러 테스트가 이루어져야 한다. 이런 과정들을 통해, 우리는 성능, 신뢰도, 보안성의 수준을 충분히 끌어올릴 것이다.

8.2.1.1. Performance

Highlight 가 기존의 강의 플랫폼과 가장 큰 차이를 나타내는 부분이 실시간 강의를 위한 영상, 음성 및 여러 데이터의 동시다발적 공유이기 때문에, 성능적인 결합으로 인한 데이터 표시에 딜레이가 과해서는 안될 것이다.

통신상태에 문제가 없다는 가정하에, 기본적으로 앱의 실행이후 3초 이내 메인화면에 진입할 수 있도록 앱 최초 실행과정에서 최적화에 신경 써야 한다. 또한 로그인 프로세스는 2초 이내에 진행되어야 하고, 각 기능의 선택에 따른 화면의 전환과 데이터의 표시는 1초 이내에 이루어질 수 있도록 해야한다. 또한 Highlight 에는 이전 플랫폼의 강의 업로드 기능이 여전히 남아있고, 관련해서 업로드할 강의 영상의 인코딩은 1시간 강의 기준 1분 이내에 완료되어야 한다. 이런 요구사항들은 어플리케이션 자체적인 최적화 및 개별 기기의 하드웨어적 스펙에 따라 좌우되는 부분이므로, 어플리케이션의 최소 기기 요구사양인 2GB RAM, 1.6GHz CPU, 32GB 저장공간, Android 6.0 또는 iOS 13.0 에 최대한 근사한 기기를 통한 테스트가 진행되어야 할 것이며, 이 기기들을 기준으로 위의 제한 시간 내에 실행이 완료되도록 최적화되어야 할 것이다.

또한 모든 강의관련 자료의 업로드는 최소 10Mbps 에서 최대 100Mbps 의 속도를 지원해야 하고, 다운로드는 10Mbps 에서 200Mbps 의 속도를 지원해야 한다. 이 테스트는 pdf, jpg, docx, pptx, xlsx, png, tar, zip 등의 다양한 확장자를 갖는 파일에 대해 진행될 예정이며, 인터넷 연결 상태에 따른 전송 속도의 차이를 분석하여 업/다운로드 가능한 자료의 크기 또한 제한되어야 할 것이다.

또한 실시간 공유 문서의 편집은 기기에서 편집 후 약 1초 이내의 딜레이로 서버에 적용되어야 한다. 동시에 한 공유 문서에는 최대 6명의 유저가 편집기능을 실행한 채로 접근할 수 있어야 한다. 이와 함께 채팅기능은 0.1초 이내의 딜레이 만이 허용된다. 이 부분에서는 문서의 편집과, 편집 결과가 실시간으로 서버에 반영, 다시 다른 유저들에게 적용되는 과정까지 시간을 측정하는 테스트가 있을 예정이고, 이 과정에서 문서에 글 및 서식 뿐만이 아닌, 사진 등의 문자 외적인 요소의 편집이 있는 경우도 테스트 해야 할 것이다.

8.2.1.2. Reliability

Highlight 는 기존의 플랫폼에 더해 수많은 서브 컴포넌트로 구성되어 있다. 따라서 Highlight 가 오류없이 작동하기 위해서는 각각의 서브 컴포넌트가 제대로 작동해야 하고, 이들이 아키텍처에 따라 정확히 결합되어야 할 것이다. 따라서, development test 는 unit developing stage 에서부터 진행되어야 하고, 매번 다른 컴포넌트가 추가될 때마다 반복적으로 진행되어야 한다.

8.2.1.3. Security

사용자의 개인 정보를 보호하는 것은 개발자 차원에서 당연히 안전장치를 마련해야 할 부분이다. 본인 또는 개인 정보 수집에 대해 약속되지 않은 사용자가 개인 정보에 무단으로 접근하는 것은 절대 있어서는 안된다.

시스템의 보안성을 위해서 우리는 매 버전 어플리케이션의 개발 및 진화가 완료된 뒤, 수동으로 코드에 대한 리뷰를 진행하고, 이로부터 보안 이슈를 잡아내어 수정해야 한다. 또한 개발자 차원에서의 직접적 분석 이외에도 Ostorlab 등의 어플리케이션 자동 분석 서비스를 이용할 수 있다.

8.2.2. Release Test

성공적인 소프트웨어 개발 프로젝트란, 그 설계, 구현 뿐만 아니라 배포까지 포함하는 개념이다. 아무리 무결점에 최적화가 잘 된 어플리케이션일지라도 배포하는 방법이 잘못되어있다면 그 어플리케이션은 구동과정에서 문제를 일으킬 가능성이 있다. 따라서, release test는 소프트웨어의 배포가 문제없이 진행되어, 설계된 어플리케이션이 사용자에게 계획대로 사용될 수 있는지를 테스트하는 과정이다. 모순처럼 느껴질 수 있지만, 이러한 release test는 반드시 소프트웨어의 정식 배포 전에 실행되어야 한다.

소프트웨어 배포 생명 주기(Software Release Life Cycle)에 따르면, release test는 기본적인 구현이 끝난 시점부터 진행하며, 이를 '알파' 테스트라고 한다. Development test는 이 알파 테스트 기간부터 계속해서 진행되며, 알파 테스트의 결과에 따라 '베타' 버전을 새로이 배포, '베타' 테스트에서는 실제 유저들과 함께 테스트를 진행하게 될 것이다.

베타 테스트가 끝난 뒤, 개발자 뿐만 아닌, 유저들로부터의 의견도 수렴하여 어플리케이션의 수정을 진행할 수 있을 것이다.

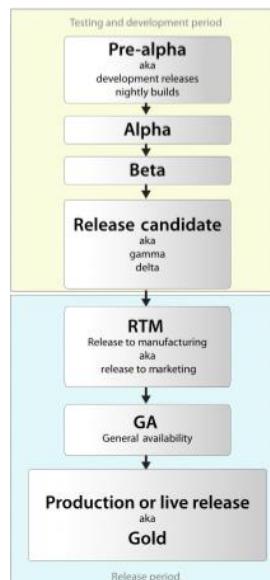


그림 54 소프트웨어 배포 생명 주기

8.2.3. User Test

위의 테스트에 덧붙여, 우리는 실제 사용자가 처할 법한 상황을 상정할 필요가 있다. 이를테면, 1 명의 교수와 2 명의 조교, 60 명의 학생 집단에 대해서 Highlight 의 베타 버전을 배포하고, 이를 통한 수업의 진행으로 실제 상황에 대한 테스트를 진행할 수 있다. Highlight 를 통한 간단한 수업 및 팀 과제를 진행한 후, 각각의 사용자들로부터 피드백을 수렴하여 Highlight 의 개선에 이용할 것이다.

8.2.4. Test Case

우리의 목적은 Highlight 가 “본래 기획한 의도대로 사용될 수 있는가”, “실제 구동과정에 예기치 못한 오류가 생기는가”, “실제 사용자의 환경에서 충분히 제 성능을 낼 수 있는가”의 3 가지 과제에 대해 문제없이 기능을 수행할 수 있도록 하는 것이다. 각각의 과제는 “기획하지 않은 의도의 사용에 대한 테스트”, “시스템의 허점을 찌르는 경우에 대한 테스트”, “네트워크 환경 및 사용자의 실제 기기 상황에서 사용시에 생기는 문제점에 대한 테스트”를 진행하여 문제를 파악할 수 있을 것이며, 이를 기반으로 시스템 평가서를 작성할 것이다.

9. Development Plan

9.1. 이 챕터에서는...

이 챕터에는 어플리케이션 개발에 사용된 여러 외부 기술 및 프로그램이 기술되어있다.

9.2. Frontend Environment

9.2.1. Adobe Illustrator



그림 55 Adobe Illustrator 로고

Adobe Illustrator 는 벡터 이미지를 제작하는 툴로, 간단히 말해 “그림을 그리는 프로그램”이다. Highlight 가 사용자 맞춤형 프로그램이라는 성향을 띠는 만큼, 사용자에게 친근한 백그라운드 이미지 및 아이콘의 사용은 필수적이고, Adobe Illustrator 는 이러한 목적에 부합하는 이미지를 제작하는데 사용할 수 있다.

9.2.2. Adobe Xd



그림 56 Adobe Xd 로고

Adobe Xd 는 웹 디자인 또는 모바일 어플리케이션의 UI 및 UX 디자인을 위한 프로토타이핑 소프트웨어이다. Adobe 사에서 무료로 제공하는 프로그램으로, 이 프로그램의 주된 장점 중 하나는 디자인한 어플리케이션에 대해 즉각적으로 프로토타입을 만들어 시각화 할 수 있고, 만들어진 프로토타입을 개발팀원들과 즉각적으로 공유가 가능하다는 점이다. 따라서 디자인 과정에서의 빠른 피드백이 가능하므로 Adobe Xd 를 사용한다.

9.2.3. Flutter



그림 57 Flutter 로고

Flutter 는 구글에서 제공하는 크로스 플랫폼 개발 환경이다. Flutter 를 통해 Highlight 의 구현된 소스코드는 Android OS, iOS 의 두 환경에서 구동 가능한 제품으로 컴파일 될 수 있다. 또한 Flutter 의 사용은 웹 브라우저, Windows, Mac, Linux 등의 수많은 환경에서 가능하기 때문에 다수의 개발자가 통일된 환경에서 컴파일 하는 것이 가능하다는 장점이 있다.

9.3. Backend Environment

9.3.1. Github



그림 58 Github 로고

Github는 소프트웨어 개발 버전을 관리하여 다수의 개발자들 간의 협업을 지원하는 코드 호스팅 플랫폼이다. 이를 통해 개발팀은 Highlight 프로젝트를 동시다발적으로 개발할 수 있고, 개발된 요소들을 쉽게 취합할 수 있도록 한다.

9.3.2. Jenkins



그림 59 Jenkins 로고

Jenkins는 위의 Github과 마찬가지로 소프트웨어 개발에 있어 지속적인 통합과 지속적인 전달 환경을 구축하기 위한 툴이다. 그럼에도 Jenkins를 사용하는 이유는 Jenkins가 제공하는, 구현된 요소들에 대한 빠르고 강력한 통합, 빌드, 테스트에 있다. 또한 Adobe XD로 작성한 UI까지도 자동화된 과정으로 통합하여 배포에 더 가까운 위치까지 테스트할 수 있다는 이점이 있다.

9.3.3. Firebase



그림 60 Firebase 로고

Firebase는 실시간 데이터베이스, 클라우드 저장소, 앱 테스트 등의 기능의 지원을 통해 어플리케이션의 개발을 돋는 통합 어플리케이션 개발 플랫폼이다. 우리는 실시간 데이터 베이스 관리 기능을 통해 사용자들의 개인정보, 과목 내 강의 컨텐츠의 관리 등을 수월하게 진행할 수 있다. Firebase가 지원하는 실시간 데이터베이스 지원 기능 덕분에 Highlight의 사용자들이 업로드한 데이터는 일괄적으로 반영되어 공유 가능하다.

9.3.4. Mosquitto



그림 61 Mosquitto 로고

Mosquitto는 MQTT 브로커 중 하나이다. 기존의 backend environment들은 http 통신과 같이 복잡하고 무거운 통신들을 담당하고 있는데 mosquitto의 경우 가벼운 mqtt 통신을 진행한다.

Highlight 는 이러한 기능을 사용하여서 전달 내용이 많지 않지만, 초단의로 통신이 일어나는 프로세스들을 이용 할 때 사용한다. 특히 mosquitto 의 경우 유저가 연결되어 있는 것을 실시간으로 파악 할 수 있기 때문에 실시간 공유 문서를 개발하는데 도움이 된다.

9.3.5. FFMPEG



그림 62 FFMPEG 로고

FFMPEG 는 Highlight 의 음성 채팅 기능을 사용 하기 위해 선택 한 툴이다. 기존의 음성 스트리밍의 경우 RTP 나 RTTP 를 사용해야 하지만 이 툴에는 live audio streaming 이 있기 때문에 이를 통해서 음성 채팅 기능을 사용 할 수 있다.

9.4. Constraints

Highlight 는 이전에 작성된 요구사항 명세서 및 해당 문서로부터 상세화된 본 문서의 아키텍처 디자인에 따라 구현될 예정이다. 이 외의 세부적인 구현 방향성은 아래의 제약사항에 따라 결정되고, 이 범위를 넘어선 것은 Team 1 의 재량 및 소비자의 추가적인 요구사항에 따라 진행될 것이다.

- Highlight 는 기기 내 앨범 및 마이크/카메라의 사용에 있어 반드시 사용자 기기에 접근권한을 요구하며, 권한의 변경은 사용자 기기 내에서 조정/변경이 가능해야 한다.
- 최종적으로 Highlight 가 차지하는 용량이 너무 과도해서는 안된다. 즉, 최대한 차지 용량이 적은 방향으로 개발해야 한다.
- 데이터의 다운로드 및 업로드는 어플리케이션의 정상 작동 중에만 진행되어야 하며, 사용자가 다운로드/업로드에 대해 인지할 수 있도록 해야 한다.
- eBook, 강의 컨텐츠 접근에 있어 저작권 윤리를 해치는 행위가 불가능해야 한다.
- 개발과정에서 open source SW 가 아닌 다른 API 의 사용을 피해야 한다.
- 가능한 한 성능을 개선하는 방향으로 개발해야 한다.
- 기본 UI 를 설계함에 있어, 색각 이상자의 정상적인 사용 가능여부 등 다양한 사용자 환경 변수를 최대한 고려해야 한다.
- 소스 코드를 작성함에 있어, 차후의 시스템 진화나 유지보수를 위해 주석의 작성은 충분해야 한다.
- 불필요한 리소스의 소모를 막기 위해 가능한 한 최적화에 신경써야 한다.

9.5. Assumption and Dependencies

Highlight 의 구동을 위한 기기 최소 사양은 2GB 의 RAM 용량과 1.6GHz 프로세서의 내장으로, 프로그램의 출시를 위한 테스트는 해당 성능과 제일 유사한 성능의 기기를 통해 진행될 것이다. 또한 OS 환경은 안드로이드 OS 6.0(API 레벨 23), iOS13.0 이상의 버전을 전제로 하며, 테스트 역시 같은 환경에서 진행될 예정이다. 따라서 위에서 언급한 사양에 미달한 기기로는 Highlight 의 구동이 매끄럽지 못하거나 구동이 불가능할 수 있다.

Highlight 는 제공하는 각 기능들에 사용된 오픈 소스 툴(Agora, Blockly, OpenShot 등)에 대한 의존성을 갖는다. 또한 카메라/マイ크/내부 데이터 접근/푸시 알림 등 사용자 기기에 대한 접근 권한들을 요구할 수 있으며, 허용되지 않은 권한에 대해서는 어플리케이션의 정상적인 사용이 제한될 수 있다.

10. 추가 정보

10.1. 서식

이 요구사항 명세서는 IEEE Recommendation (IEEE Recommended Practice for Software Requirements Specifications, IEEE-Std-830). 서식을 따라 제작되었다.

10.2. 문서 시간표

날짜	버전	설명(편집 파트)	참가자
2021/04/27	0.0.0	문서 작업 시작	전체
2021/04/31	0.0.1	세부 문서 사항 명세	안이은/이성원/김지훈
2021/05/07	0.0.2	문서 양식 확정	김지훈
2021/05/08	1.0.0	7	박정인
2021/05/09	1.1.0	1, 2, 3, 8, 9	이성원
2021/05/10	1.2.0	4	정세린
2021/05/10	1.1.1	1, 2, 3, 8, 9	이성원
2021/05/10	1.3.0	5	양유진
2021/05/10	1.4.0	6	김지훈
2021/05/11	1.3.1	5	양유진

Project Highlight**Design Specification**

2021/05/12	0.0.3	맞춤형 알림 개조	안이은
2021/05/12	1.2.1	4	정세린, 안이은
2021/05/12	1.3.2	5	양유진
2021/05/12	1.4.1	6	김지훈
2021/05/12	1.0.1	7	박정인
2021/05/12	1.1.3	1, 2, 3, 8, 9	이성원, 안이은
2021/05/14	1.3.3	5	양유진, 안이은
2021/05/14	1.4.3	6	김지훈, 안이은
2021/05/15	1.1.4	1, 2, 3, 8, 9	이성원
2021/05/16	2.0.0	수합 및 통일	김지훈
2021/05/16	2.0.1	그림 업데이트	양유진
2021/05/16	2.0.2	폰트 통일	양유진
2021/05/16	2.0.3	캡션 수정	양유진 정세린
2021/05/16	2.0.4	브로커 정보 추가, OMT 추가	김지훈