SKKU:MEET

Revolutionary Online Education Platform for School using Real-Time Eye Tracking Software

Software Design Specification

2021.05.13.

Introduction to Software Engineering 41

TEAM 9

Team Leader 이보현
Team Member 강병남
Team Member 이재훈
Team Member 임승재
Team Member 조효정

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. Preface

이 챕터에서는 이 문서의 예상되는 독자를 정의하고 문서의 각 챕터를 소개한다.
그리고 문서의 Scope, Objective 및 이 문서의 구조에 대해 서술한다.

1.1. Readership

본 문서는 설계 및 개발에 관여하는 예상 독자들을 위해 작성되었다. 본 문서의 주 독자는 시스템을 개발하는 소프트웨어 엔지니어 (즉 Team 9의 구성원들)과 시스템 설계 및 개발에 관여하는 모든 관계자들, 소프트웨어 공학 개론 수업의 교수, 조교, 참여 학생이다.

1.2. Scope

이 문서는 eye-tracking 및 실시간 설문을 이용한 학생들의 집중도를 체크하는 서비스를 제공하는 스마트 캠퍼스 어플리케이션의 구현에 사용할 설계의 정립으로 사용된다.

1.3. Objective

이 소프트웨어 설계 문서의 주요 목적은 실시간 집중도 체크 및 수업 통합 관리를 위한 스마트 캠퍼스 어플리케이션의 기술적 설계(design)에 대한 설명이다. 이 문서는 스마트 캠퍼스 어플리케이션 구현의 기반이 되는 소프트웨어 프론트엔드, 백엔드 구조 및 프로토콜, 데이터베이스 측면에서의 설계를 정의한다. 또한 시스템의 테스트 및 개발 계획을 서술한다. 모든 설계는 앞서 제작된 Software Requirements Specification 문서의 요구 사항을 기반으로 작성되었다.

1.4. Document Structure

- **1. Preface**: 본 문서의 목적, 예상 독자 및 문서의 구조에 대해 설명한다.
- **2. Introduction**: 본 문서를 작성하는데 사용된 다양한 도구들과 다이어그램들, 참고 자료들에 대해 설명한다. 또한 본 프로젝트의 시스템의 범위를 서술한다.
- **3. Overall System Architecture**: 시스템의 전체적인 구조를 Context 다이어그램, Use-Case 다이어그램, Sequence 다이어그램을 이용하여 서술한다.
- **4. System Architecture - Frontend**: Frontend 시스템의 구조를 Class 다이어그램과 Sequence 다이어그램을 이용하여 서술한다.
- **5. System Architecture - Backend**: Backend 시스템의 구조를 Sequence

다이어그램과 Class 다이어그램을 다이어그램을 이용하여 서술한다.

- **6. Protocol Design**: 클라이언트와 서버의 커뮤니케이션을 위한 프로토콜 디자인을 서술한다.
- **7. Database Design**: 시스템의 데이터베이스 요구 사항을 기반으로 ER 다이어그램과 Relation Schema를 이용하여 시스템의 데이터베이스 디자인을 서술한다.
- **8. Testing Plan**: 시스템을 위한 테스트 계획을 서술한다.
- **9. Development Plan**: 시스템의 구현 계획 및 그것을 위한 개발 도구, 라이브러리 등의 개발 환경을 설명한다.
- **10. Supporting Information**: 본 문서의 작성 및 수정 기록을 기술한다.

2. Introduction

이 프로젝트는 코로나19 상황에서 원활한 비대면 수업의 활성화를 목적으로 한다. 이 시스템은 교수가 비대면 온라인 수업에서 실시간으로 학생들의 학습 집중도와 이해도를 파악할 수 있도록 하며, 외부의 시스템을 이용하지 않고 수업 관리 시스템과 실시간 수업 시스템을 한 번에 이용할 수 있도록 한다. 시스템은 학생의 집중도를 eye tracking을 이용하여 실시간으로 측정할 수 있어야 한다. 이 design document는 프로젝트의 구현에 있어서 기반이 될 수 있는 설계(design)을 제공한다. 또한, 설계는 앞서 제작된 Software Requirements Specification 문서에서 명시된 요구 사항을 따른다.

2.1. Objectives

이 챕터에서는 본 System Design Specification 문서의 작성에 사용된 도구들 및 다이어그램들을 설명한다. 또한 프로젝트에서 개발하는 시스템의 범위와 참조 자료들을 서술한다.

2.2. Applied Diagrams

A. Unified Modeling Language (UML)

Unified Modeling Language는 통합 다이어그램 집합으로 구성된 표준화된 통합 모델링 언어다. UML은 시스템의 모든 것을 문서화, 지정, 구축하는데 사용되는 표준 언어다. UML은 여러 개의 표준화된 다이어그램을 제공한다. 그 목록은 다음과 같다: use-case

diagram, class diagram, sequence diagram, communication diagram, activity diagram, state diagram, component diagram, deployment diagram, package diagram. UML 다이어그램을 이용함으로써 프로그래밍을 단순화시켜 표현하여 커뮤니케이션의 활성화가 가능하며 대규모 프로젝트 구조의 로드맵과 개발을 위한 시스템 구축의 기반을 설립 가능하다. UML은 시각화, 명세화, 구축, 문서화 언어다. 즉, 시각적인 형태로 표현하며 표준화된 다이어그램을 제공한다. 또한 소프트웨어 개발의 분석, 설계 단계의 각 과정에 필요한 모델을 명세화한다. 그리고 UML은 다양한 프로그래밍 언어로 표현할 수 있으며 UML 모델을 코드로 자동 변환할 수 있다. 마지막으로 UML의 설계한 내용을 자동으로 문서화 가능하다.

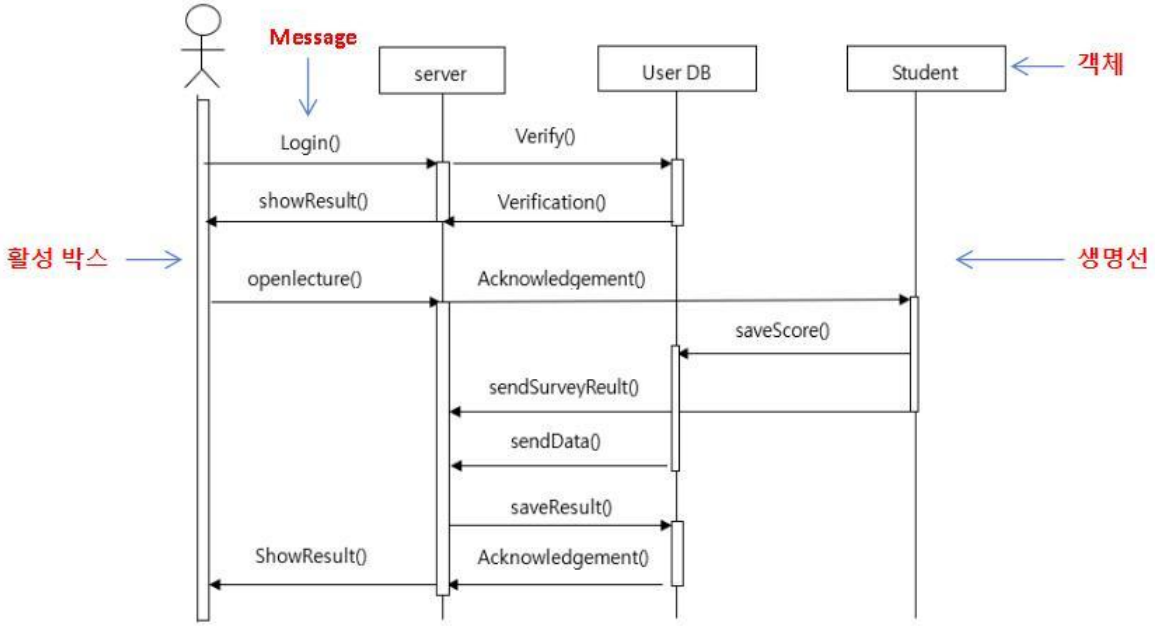
B. Context Diagram

Context Diagram은 Data Flow Diagram에서 가장 높은 수준의 다이어그램으로, 시스템 전체와 외부 요인의 입력 및 출력을 보여준다. Context diagram은 시스템과 그 시스템과 상호작용할 수 있는 모든 외부의 엔티티들을 나타내며, 그 사이의 정보의 흐름을 표현한다. 시스템 내부 구조에 대한 세부 정보는 명시적으로 제외한다. Context diagram은 비즈니스 계획 또는 프로젝트 요구 사항에서 누락 및 오류를 확인하는데 적합하며, 프로젝트의 위험을 크게 줄일 수 있다. 또한 Context diagram은 개발 담당자가 어떤 사용자 그룹을 고객으로 간주하는지 명확히 하는 데 도움이 된다. 이 다이어그램은 프로젝트에서 설계할 시스템의 세부 정보와 boundary를 이해하기 위해 널리 사용된다. 다른 프로젝트 다이어그램과 달리, Context diagram은 엔지니어 또는 개발자가 사용하기보다는 프로젝트의 이해 관계자가 사용한다. 따라서 이해관계자가 항목을 분석할 때 쉽게 이해할 수 있도록 간단하고 이해하기 쉬운 언어로 작성해야 한다.

C. Sequence Diagram

Sequence Diagram은 특정한 행동이 어떠한 순서로 어떤 객체와 어떻게 상호작용을 하는지 표현하는 행위 다이어그램이다. Sequence Diagram은 Event Diagram이라고도 부른다. 이것은 시나리오와 관련된 객체와 시나리오의 기능 수행에 필요한 객체 간에 교환되는 메시지를 순서대로 표현한다. 모든 커뮤니케이션은 시간 순으로 표현된다. 현재 존재하는 시스템이 어떤 시나리오로 작동하고 있는지를 나타낼 수 있다. Sequence Diagram을 위해 필요한 필수적인 구성 요소로는 생명선(life line), 메시지, 활성 박스(Activation box)가 있다. 생명선은 상호작용에 참여하는 오브젝트를 의미하며, 메시지는 서로 다른 객체 간의 상호작용 혹은 의사소통 통신을 정의하는 요소다.

메시지의 종류로는 동기 메시지, 비동기 메시지, 자체 메시지, 반환 메시지가 있다. 동기 메시지는 해당 메시지에 대한 응답을 기다린다. 비동기 메시지는 이와 반대로, 메시지 전송 객체가 계속하기 전까지 응답을 요구하지 않는 메시지다. 즉 전송 객체의 호출만을 표시한다. 자체 메시지는 자기 자신에게 보낸 메시지이며, 반환 메시지는 이전 호출의 반환을 기다리는 객체에 다시 반환되는 메시지다. 활성 박스란 객체의 호출이 이루어지는 박스다.

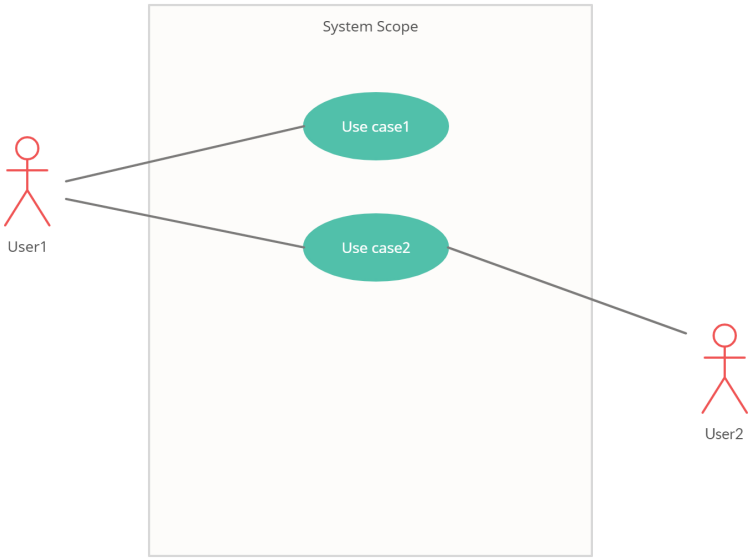


[Figure 1] Sequence Diagram Example

D. Use-Case Diagram

Use case Diagram은 시스템에서 제공해야 하는 기능이나 서비스를 명세한 다이어그램이다. Use case Diagram은 사용자와 그 사용자가 사용하는 use case들 간의 관계를 보여주며, 사용자 – 시스템 간 상호작용을 표현한다. 이 다이어그램을 통해 각기 다른 종류의 시스템 사용자와 각 사용자 별 use case를 알 수 있다. 외부에서 본 시스템의 기능을 표현하기 때문에, 사용자가 수행하는 기능을 파악하기 위해 사용된다. Use case Diagram의 기본적인 구성 요소로는 scope, use case, relationship 그리고 actor가 있다. Use case는 시스템이 제공하는 서비스와 기능이며, scope는 시스템이 제공하는 기능의 범위를 나타낸다. Actor는 시스템과 상호작용하는 시스템 외적 존재다. Actor는 사람일 수도 있으며, 사람이 아닌 외부적 시스템일 수도 있다.

Relationship은 actor와 use case, 또는 두 개의 use case 사이에 나타내며, association, include, generalization, extended라는 4가지 종류로 구성된다.



[Figure 2] Use-Case Diagram Example

E. Class Diagram

Class Diagram은 시스템을 구성하는 클래스, 그 속성, 기능 및 객체들 간의 관계를 표현하여 시스템의 정적인 부분을 보여준다. 클래스란 동일한 속성과 행위를 수행하는 객체의 집합이자 실제 객체를 생성하는 설계도다. 이 다이어그램은 실제로 구현될 소스코드와는 다를 수 있으며 그 의미나 해석 또한 경우에 따라 달라질 수 있다. 전체 시스템의 구조 및 클래스의 의존성을 파악하고 다른 사람들과 커뮤니케이션, 설계 논의를 하기 위해 해당 다이어그램이 사용될 수 있으며, 유지보수를 위한 디자인의 back-end 문서에서도 사용된다. (본 문서에서도 back-end 설계에 사용되었다.) Class Diagram은 불필요한 내용을 제외하고 간결한 모델로 그려야 하며 직관성을 위해 의미 있고 명확한 이름을 사용해야 한다. 클래스는 3가지 부분 (클래스 이름, 멤버 변수, 메서드)으로 나누어 표현된다. 클래스 이름은 가장 상단에 작성되며 중간에 멤버 변수, 가장 하단에 메서드가 작성된다.

[Figure 3] Class Diagram Example

F. Entity Relationship Diagram

ER Diagram은 구조화된 데이터와 그들 간의 관계를 사람이 이해할 수 있는 형태로 표현하는 다이어그램이며, 현실에서의 요구사항들을 이용한 데이터베이스 설계 과정에서 활용된다. 해당 다이어그램은 엔티티, 속성 (attribute), 관계성(Relationship)으로 구성된다. 엔티티란 관리할 정보의 실체로, 둥근 사각형으로 작성한다. 모든 엔티티는 하나 이상의 식별자를 가진다. 속성은 엔티티를 구성하고 있는 구성 요소이며, 관계성은 엔티티들 간의 관계이다. 강한 관계성은 개체가 다른 개체를 통해 존재할 수 있는 의존적인 관계를 뜻하며, 약한 관계성은 개체가 다른 개체와 독립적으로 존재할 수 있는 독립적인 관계를 뜻한다. 이 3가지 구성 요소들을 표시하여 ER Diagram은 데이터베이스의 논리적 구조를 설명한다.

2.3. Applied Tools

A. Microsoft PowerPoint

Diagram 및 figure의 작성을 위해 PowerPoint가 사용되었다. PowerPoint는 다양한 도형, 그림 및 텍스트의 삽입과 편집을 지원하며, 표와 그래프의 생성 또한 가능하기 때문에 문서의 작성에 필요한 Diagram 및 figure를 생성할 때 유용하게 사용된다.

B. Creately

Creately는 Cinergix에서 설계 한 다이어그램 및 디자인 기능을 갖춘 SaaS 시각적 협업 도구이다. Use-Case Diagram, Sequence Diagram 등의 다양한 Diagram 템플릿을 제공한다.

2.4. Project Scope

본 문서에서 설계하는 스마트 캠퍼스 어플리케이션은 교수로 하여금 실시간으로 학생들의 집중도와 이해도를 확인할 수 있게 해주면서 원활한 수업의 진행 및 더욱 질 높은 수업을 제공할 수 있도록 도와준다. 또한 사용자는 번거롭게 웹엑스와 같은 외부 서비스를 이용할 필요 없이 바로 수업 리스트에서 바로 해당 수업의 미팅룸에 입장할 수 있게 된다. 이 시스템은 eye-tracking을 통해 얻을 수 있는 학생의 집중도에 관한 데이터베이스에 기반을 하고 있다. 수업에 참여하고 있는 모든 학생들의 집중도는 점수화 되어 데이터베이스 서버에 실시간으로 저장되고 스마트 캠퍼스에는 교수가 이를 확인하기 용이하도록 그래프와 같은 시각적인 형태로 전환되어 나타난다. 코로나19로 인해 비대면 수업의 비중이 높아진 상황에서, 더욱 편리한 강의 제공을 위해 우리는 교수가 학생들의 수업 참여 태도를 파악하기 힘든 상황에서 학생들의 집중도를 더욱 확인하기 쉽도록 하며 상호간의 원활한 의사소통이 이루어질 수 있게 도와주는 서비스를 경험할 수 있도록 하고자 한다.

2.5. References

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

- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications, In IEEEXplore Digital Library
- http://ieeexplore.ieee.org/Xplore/guesthome.jsp
- Team 9. “Software Requirement Specification”. SKKU, Last Modified: 4. 25, 2021.
- https://github.com/skkuse/2021spring_41class_team9

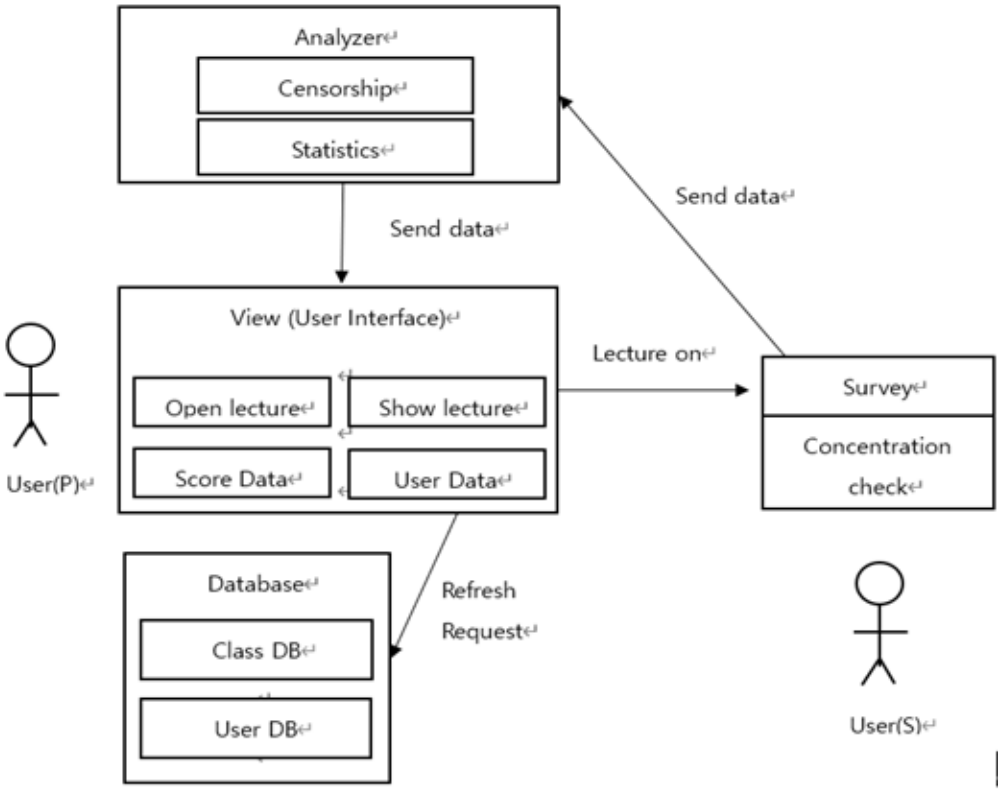
3. System Architecture – Overall

3.1. Objectives

이 챕터에서는 프론트 엔드 설계에서 백 엔드 설계에 이르는 프로젝트 애플리케이션의 시스템 구성에 대해 설명한다.

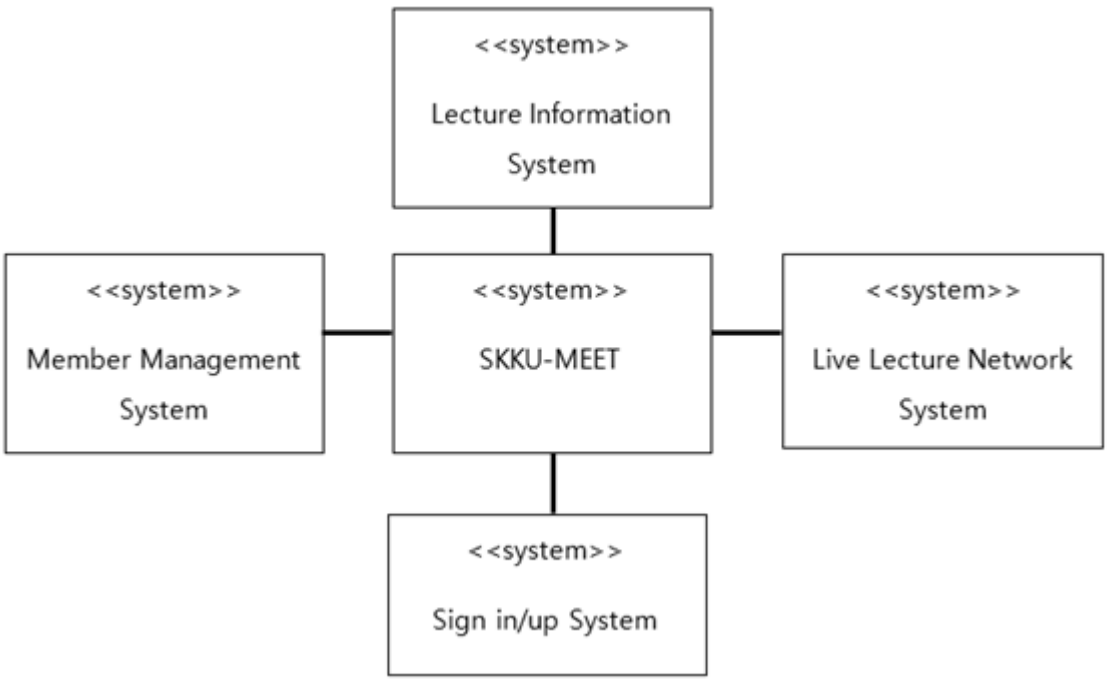
3.2. System Organization

이 서비스는 클라이언트 - 서버 모델을 적용하여 설계되었으며, 프론트 엔드 애플리케이션은 사용자와의 모든 상호작용을 담당하며, 프론트 엔드 애플리케이션 및 백 엔드 애플리케이션은 JSON 기반의 HTTP 통신을 통해 데이터를 주고받는다. 백 엔드 애플리케이션은 설계 사양 요청을 프론트 엔드로부터 컨트롤러로 배포하고, 오라클 데이터베이스에서 필요한 객체 정보를 얻어서 데이터베이스에서 처리한 후 JSON 형식으로 전달한다. 백 엔드 애플리케이션은 강의 관리 시스템에 의해 설문조사 결과를 수신 받고 검토정보가 수집되면 강의 관리 시스템은 최소한으로 처리된 설문조사 리스트를 전달하고, 강의 관리 시스템은 구글 자연어 API를 활용해 필요한 정보를 데이터베이스에 저장하게 된다. 그런 다음 사용자가 설문조사 결과에 관한 정보를 요청하면 업데이트된 정보가 전달됩니다. 비디오와 오디오 신호를 보내고 받기 위해서는 클라이언트 소프트웨어가 필요하고 또한 서버에 접속해 있는 모든 사람들에게 신호를 보내기 위해서는 리플렉터, 특별한 형태의 서버 하드웨어나 소프트웨어가 필요하다. 리플렉터는 호스트에 많은 실시간 화상회의 참가자를 접속시키기 위한 인터넷 컴퓨터이며 리플렉터에게 접속하려 할 때에는 컴퓨터에서 리플렉터에게 적절한 신호를 전송하여 리플렉터에게 접속을 하겠다는 것을 알리고 로그인하여 화상 강의를 이용할 수 있게 한다.



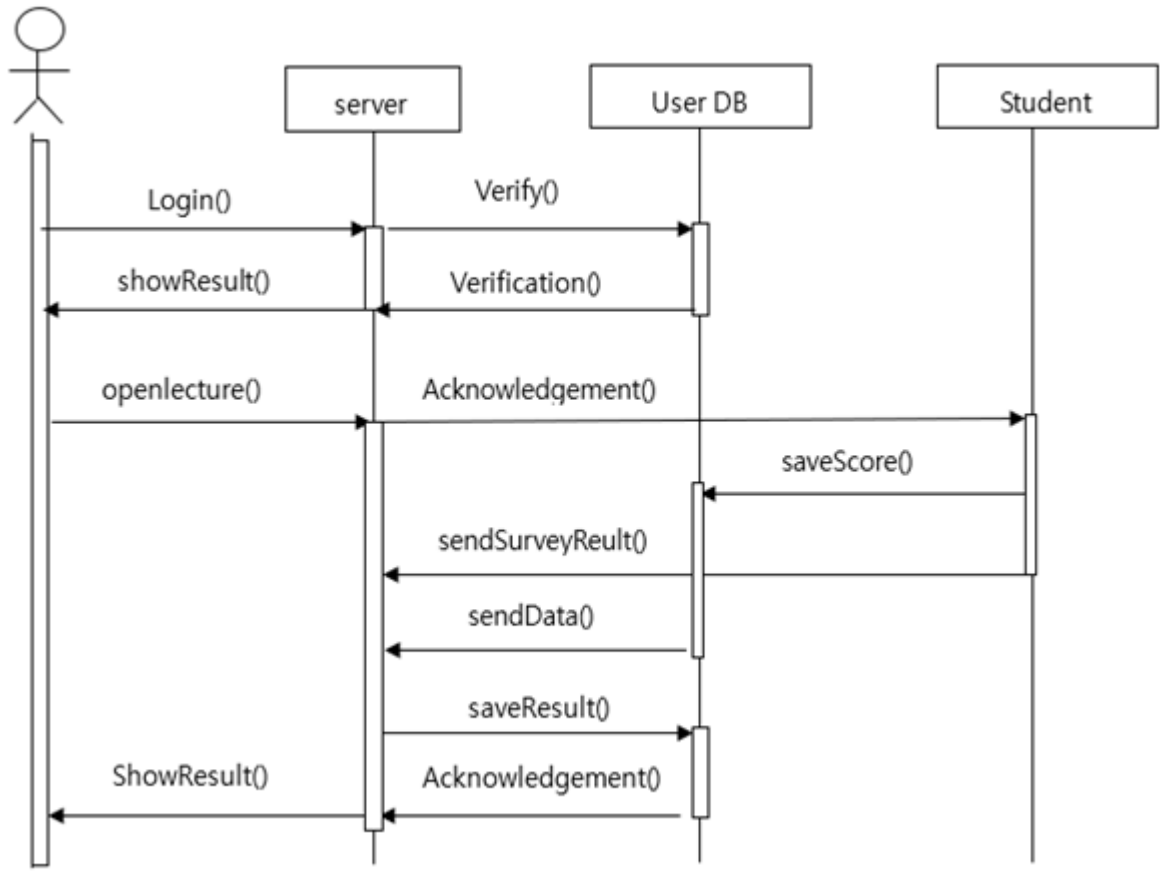
[Figure 4] Overall system architecture

3.2.1. Context Diagram



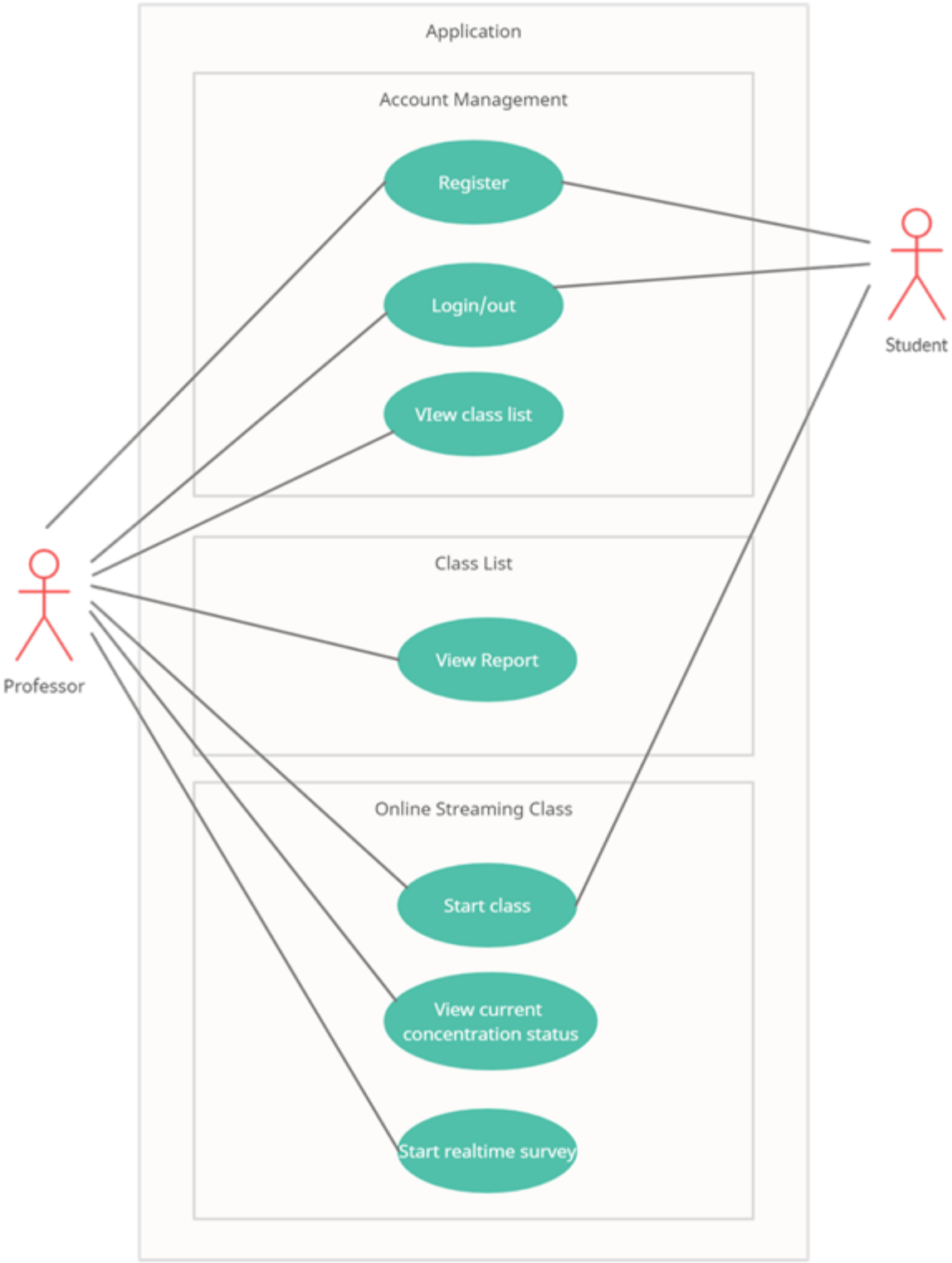
[Figure 5] Overall context diagram

3.2.2. Sequence Diagram



[Figure 6] Overall sequence diagram

3.2.3. Use Case Diagram



[Figure 7] Use case diagram

4. System Architecture – Frontend

4.1. Objectives

이 챕터에서는 프런트 엔드 시스템의 구조, 특성 및 기능을 설명하고 각 구성 요소의 관계를 설명한다.

4.2. Subcomponents

4.2.1. Profile

프로파일 클래스는 기본적인 사용자 정보를 처리한다. 사용자 등록 후 사용자는 프로파일을 입력해야 하며 사용자 로그인 후 사용자는 사용자 프로필을 수정할 수 있다.

4.2.1.1. **Attributes**

프로파일 개체에 있는 속성들은 다음과 같다.

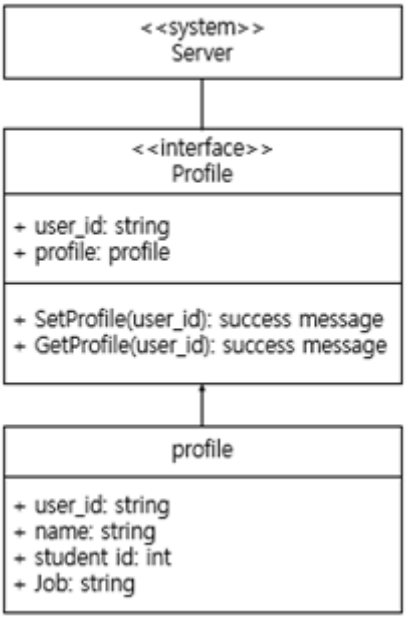
- **User id**: 사용자 id (email 주소)
- **Name**: 사용자 이름
- **Student id**: 사용자 학번(job이 student인 경우)
- **Job**: professor or student

4.2.1.2. **Methods**

프로파일 클래스가 가지는 메소드는 다음과 같다.

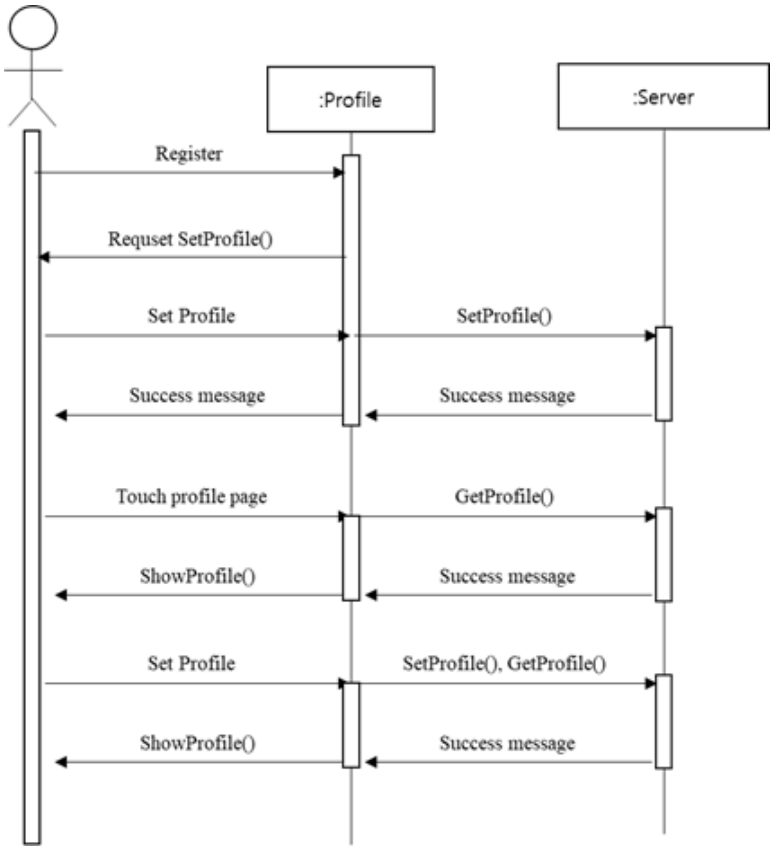
- SetProfile()
- GetProfile()

4.2.1.3. **Class Diagram**



[Figure 8] Class diagram – Profile

4.2.1.4. **Sequence Diagram**



[Figure 9] Sequence diagram – Profile

4.2.2. Score

스코어 클래스는 Eye tracking을 통한 집중도 테스트를 통해 산출된 스코어 정보를 처리하며 교수로 등록된 사용자가 조회가 가능하다.

4.2.2.1. **Attribute**

스코어 개체에 있는 속성들은 다음과 같다.

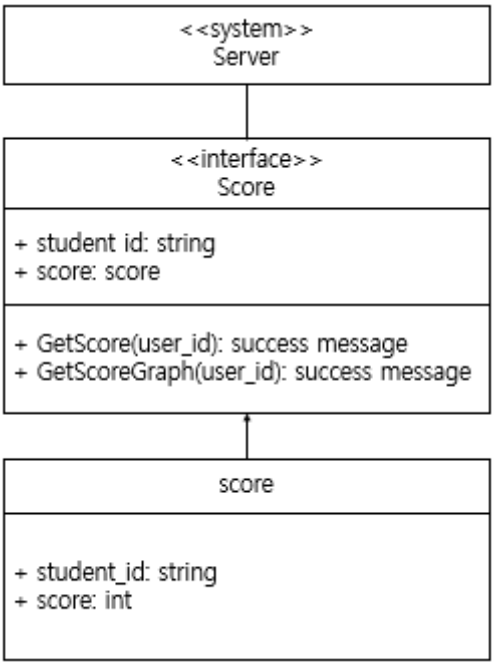
- **Student id:** score에 해당하는 학번
- **Score:** 집중도 점수

4.2.2.2. **Methods**

스코어 클래스가 가지는 메소드는 다음과 같다.

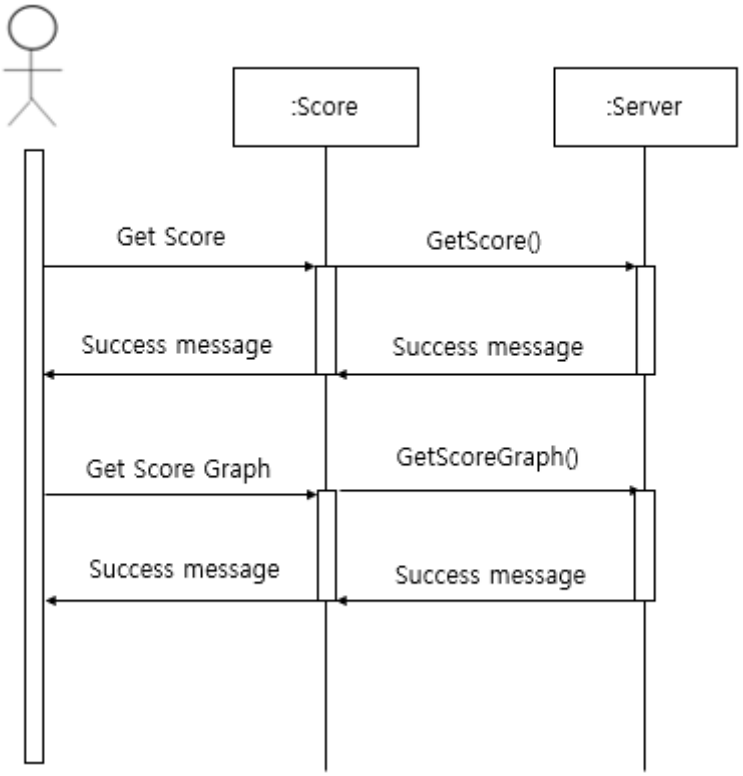
- GetScore()
- GetScoreGraph()

4.2.2.3. **Class Diagram**



[Figure 10] Class diagram – Score

4.2.2.4. **Sequence Diagram**



[Figure 11] Sequence diagram – Score

4.2.3. Meeting Room

미팅룸 클래스는 교수로 등록된 사용자가 개설한 미팅룸에 대한 정보를 처리한다. 미팅룸에서는 실시간 강의를 진행할 뿐만 아니라 Eye tracking을 활용한 집중도 체크와 그로 인해 산출된 점수를 저장하고 설문조사를 실시하여 산출된 결과를 저장한다. 설문조사 데이터는 작성자를 확인할 수 없게 익명으로 저장된다.

4.2.3.1. **Attributes**

미팅룸 개체에 있는 속성들은 다음과 같다.

- **Meeting room id:** 미팅룸의 id
- **Meeting room name:** 미팅룸의 이름(강의 이름)
- **Meeting room manager name:** 강의를 관리하는(개설한) 사용자의 이름
- **Connected user name:** 강의에 참가한 사용자의 이름

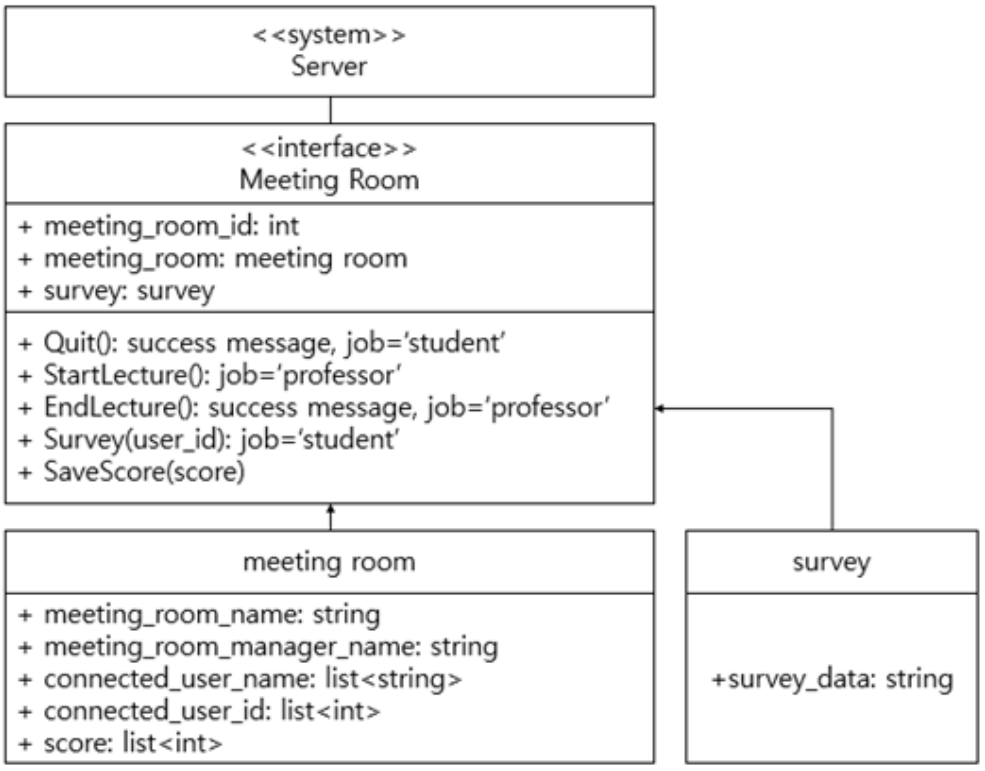
- **Connected user id:** 강의에 참가한 사용자의 id
- **Score:** 집중도 점수
- **Survey data:** 설문조사 데이터

4.2.3.2. **Methods**

미팅룸 클래스가 가지는 메소드는 다음과 같다.

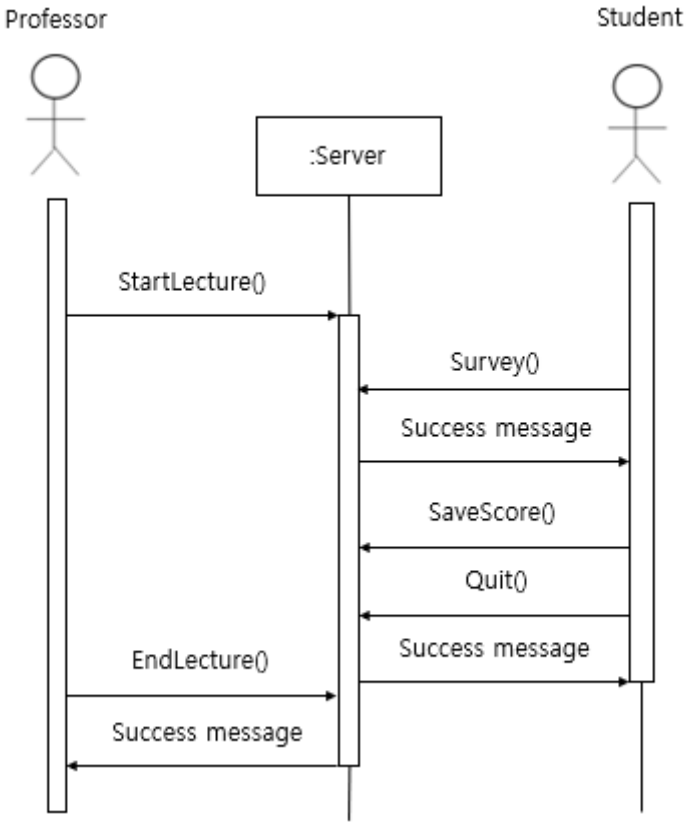
- Quit()
- StartLecture()
- EndLecture()
- Survey()
- SaveScore()

4.2.3.3. **Class Diagram**



[Figure 12] Class diagram – Meeting Room

4.2.3.4. **Sequence Diagram**



[Figure 13] Sequence diagram – Meeting Room

4.2.4. Lecture

강의 클래스는 사용자가 저장한 강의에 대한 정보를 처리한다. 강의에 대한 정보는 강의id, 강의 이름, 강의 진행시간, 강의 일정등이 저장된다. 사용자는 강의를 추가, 삭제할 수 있으며 생성된 강의에 입장할 수 있다. 교수는 자신이 관리하는 강의에서 미팅룸을 생성할 수 있으며 강의의 데이터를 수정할 수 있다.

4.2.4.1. **Attributes**

강의 개체에 있는 속성들은 다음과 같다.

- **Lecture id:** 강의 id
- **Lecture name:** 강의 이름
- **Lecture time:** 강의 진행 시간(시간)

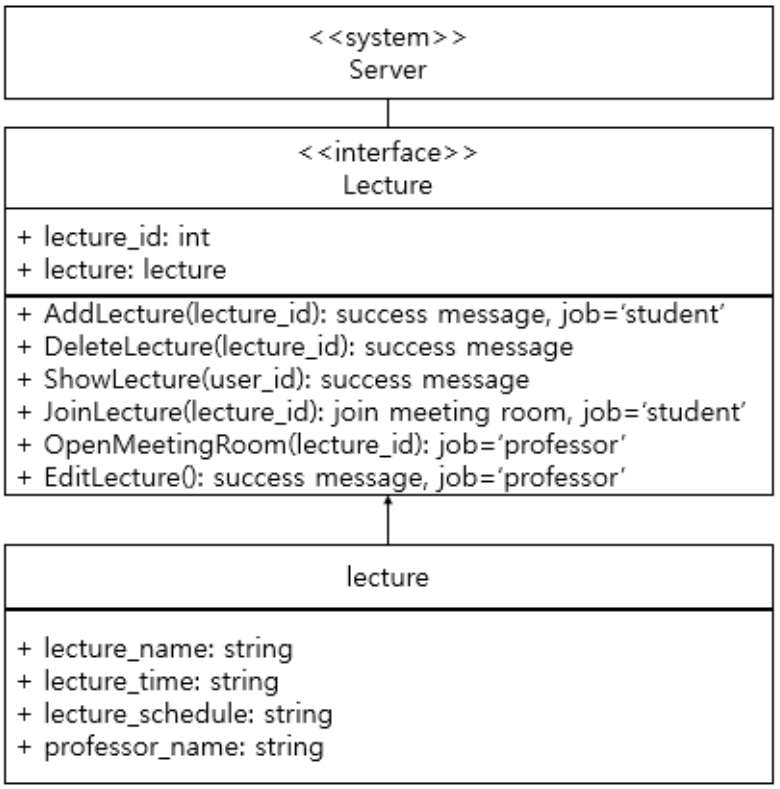
- **Lecture schedule:** 강의 스케줄(날짜(연/월/일), 시간)
- **Professor name:** 강의 담당 교수의 이름

4.2.4.2. **Methods**

강의 클래스가 가지는 메소드는 다음과 같다.

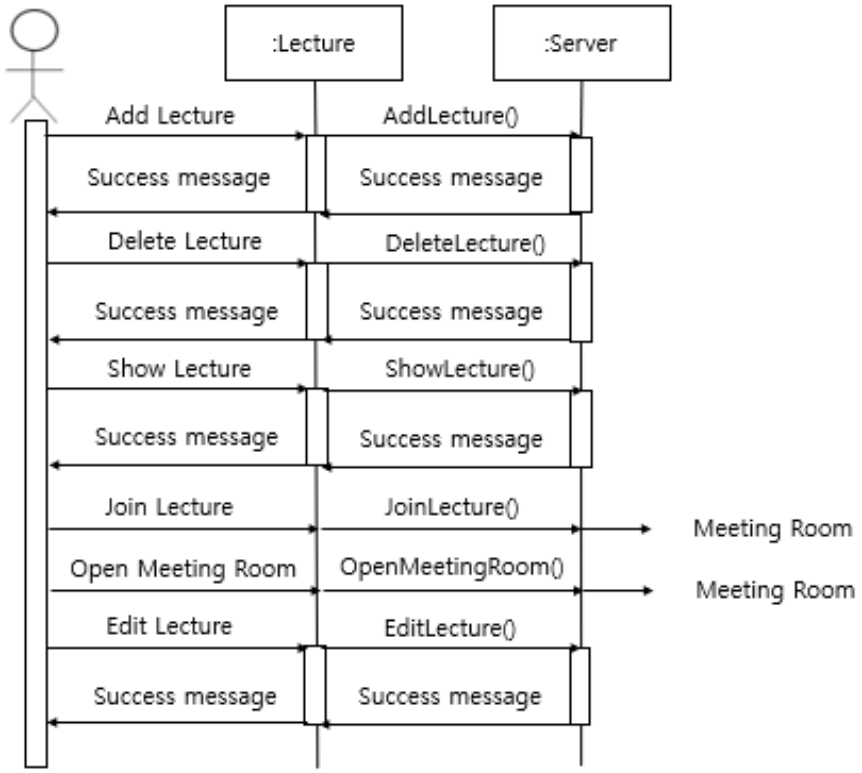
- AddLecture()
- DeleteLecture()
- ShowLecture()
- JoinLecture()
- OpenMeetingRoom()
- EditLecture()

4.2.4.3. **Class Diagram**



[Figure 14] Class diagram – Lecture

4.2.4.4. **Sequence Diagram**



[Figure 15] Sequence diagram – Lecture

4.2.5. Survey Result

설문조사 결과 클래스는 미팅룸에서 저장된 설문조사 데이터를 처리한다. 교수는 자신이 관리하는 강의에 해당하는 설문조사 데이터를 조회할 수 있다.

4.2.5.1. **Attributes**

설문조사 결과 개체에 있는 속성들은 다음과 같다.

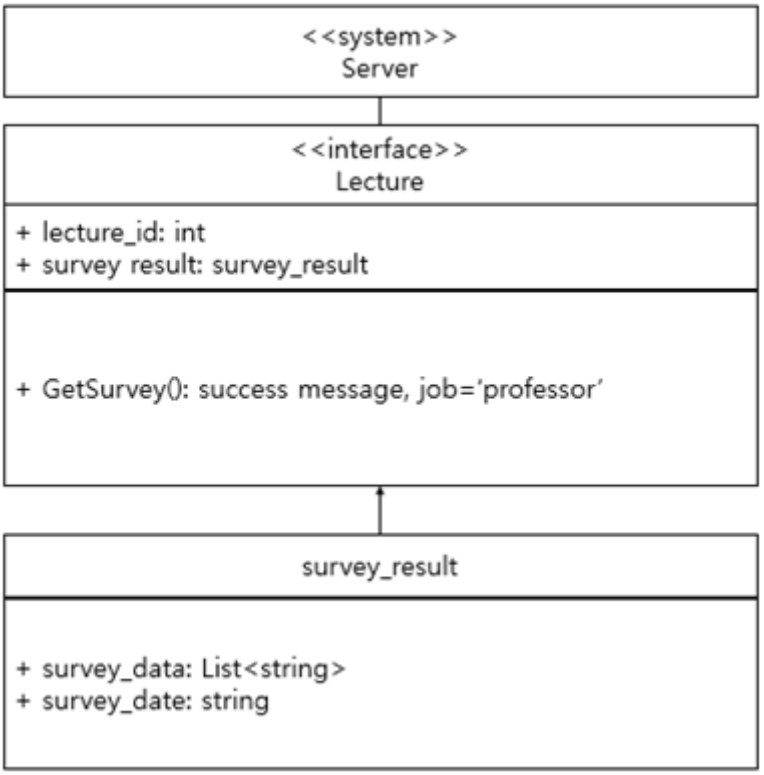
- **Lecture id:** 설문조사 결과를 확인하고 싶은 강의
- **Survey data:** 설문조사 데이터
- **Survey date:** 설문조사 날짜(연/월/일)

4.2.5.2. **Methods**

설문조사 결과 클래스가 가지는 메소드는 다음과 같다.

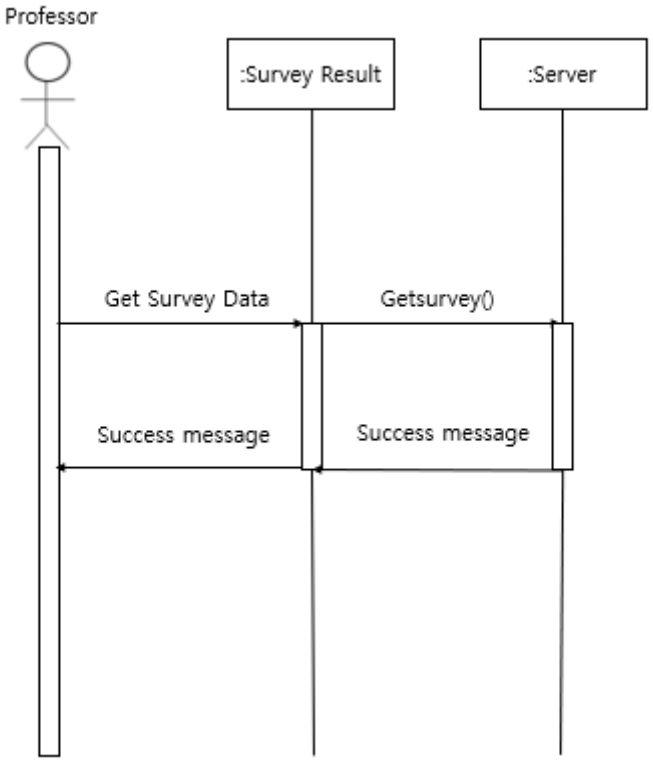
- GetSurvey()

4.2.5.3. **Class Diagram**



[Figure 16] Class diagram – Survey Result

4.2.5.4. **Sequence Diagram**



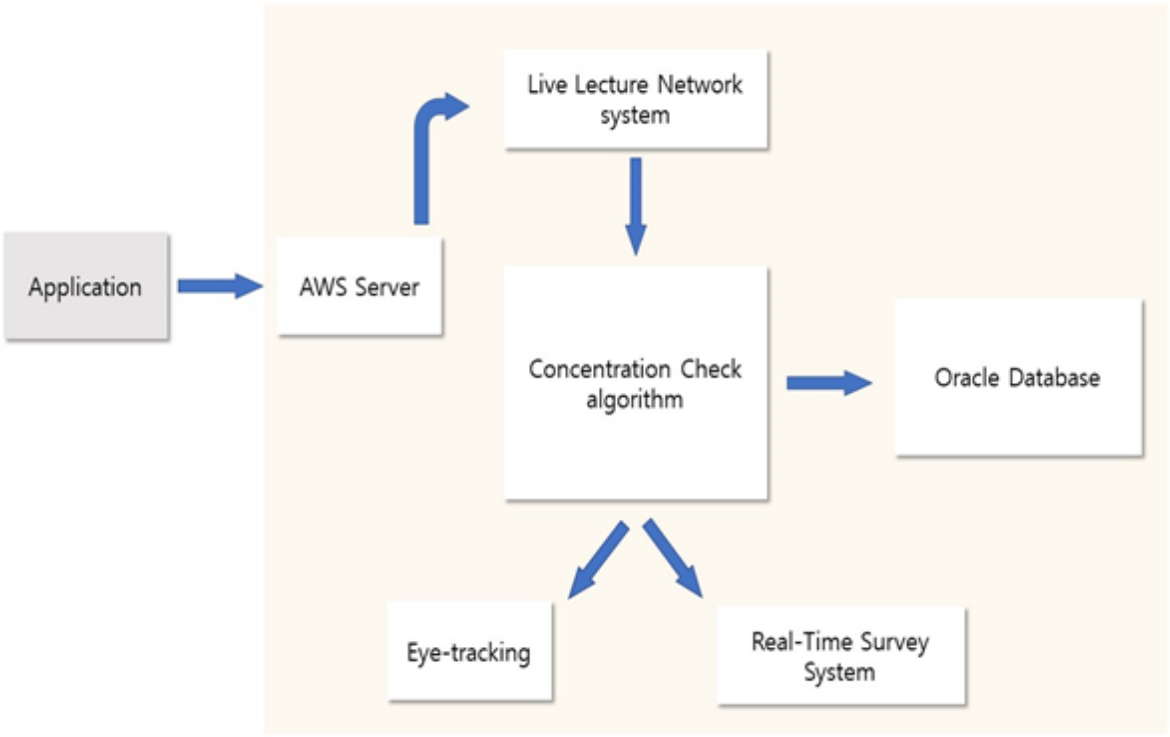
[Figure 17] Sequence diagram – Survey Result

5. System Architecture – Backend

5.1. Objectives

이 챕터에서는 back-end 시스템의 구조에 대해서 설명하도록 하겠습니다.

5.2. Overall Architecture



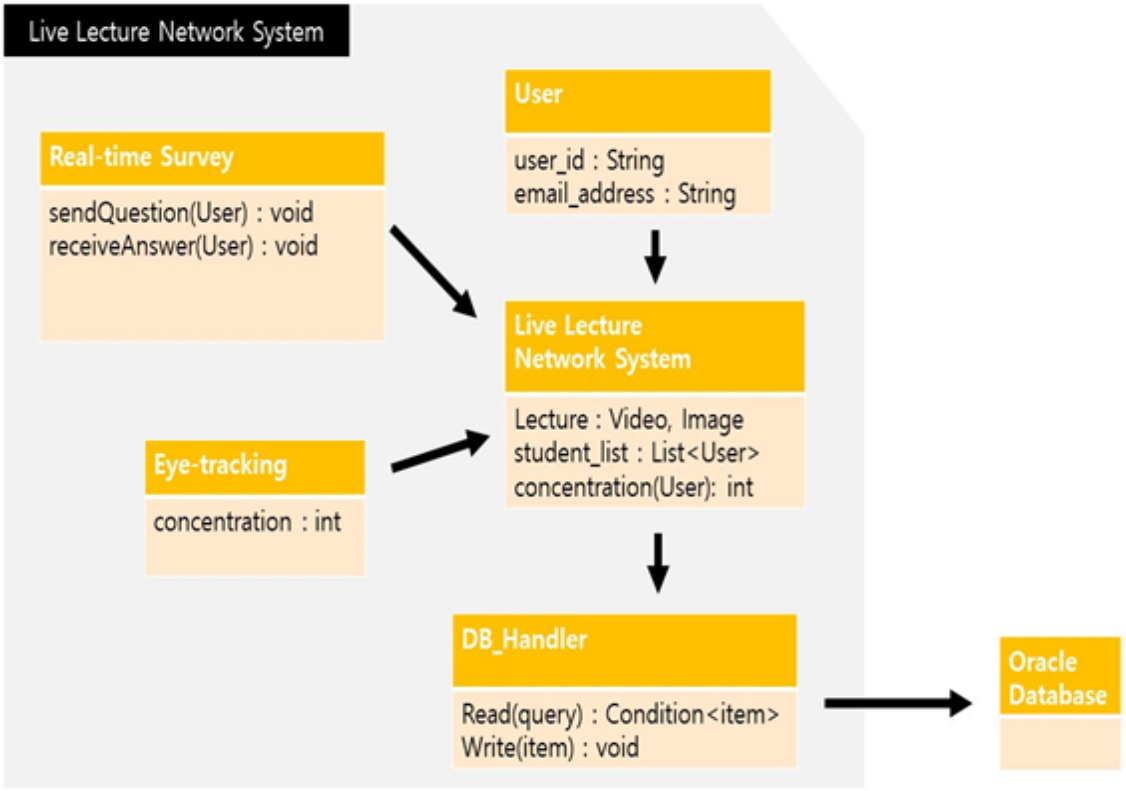
[Figure 18] Overall architecture

실시간 강의를 제공하기 위한 본 시스템의 전반적인 구조는 위와 같다. Front-end로부터 요청이 들어오면 AWS 서버를 통해 온라인 수업 환경이 생성되게 된다. 이 라이브 강의 네트워크 시스템은 집중도 체크 알고리즘에 기반을 하고 있다. 집중도 체크 알고리즘은 eye-tracking과 실시간 설문 시스템으로 구성되어 있는데 이러한 것들을 통해 얻게 되는 집중도 데이터 및 여러 시스템의 데이터들은 오라클 데이터베이스 서비스를 통해 관리되며 실시간 수업이 시작한 이후 로그 정보는 로컬 데이터베이스에 연동된다. 교수는 이러한 데이터베이스를 통해서 수업 중에 획득한 학생들의 집중도에 대한 데이터 및 실시간 설문에 대한 답을 확인하고 얻어올 수 있다.

5.3. Subcomponents

5.3.1. Live Lecture Network System

5.3.1.1. Class Diagram



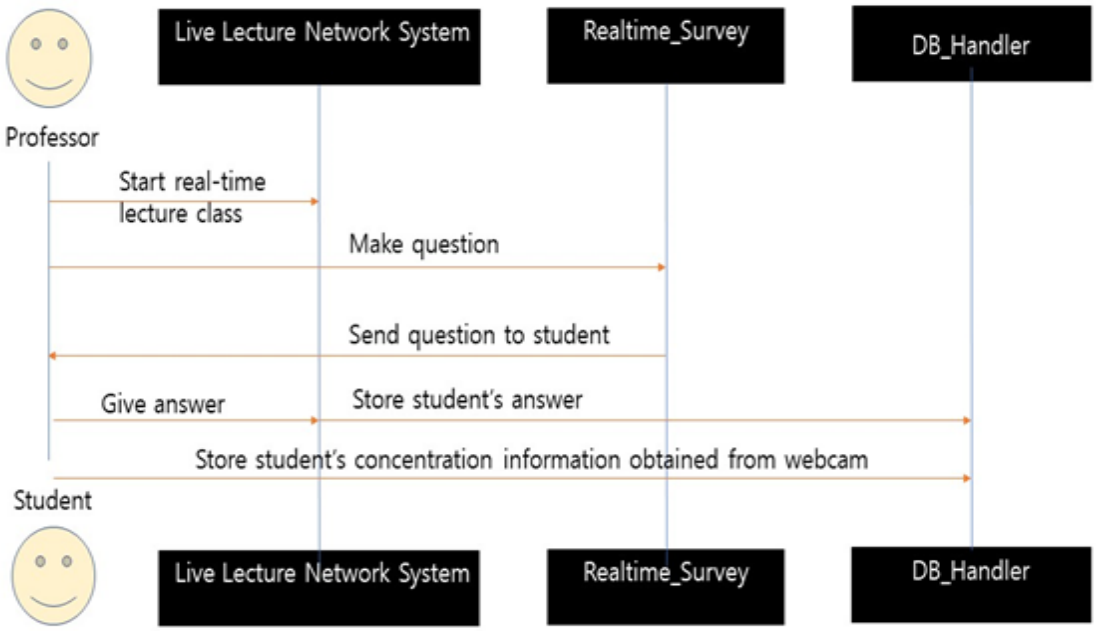
[Figure 19] Class Diagram - Live Lecture Network system

- Class Description

- Live lecture Network System class: 이 클래스는 실시간 설문과 eye-tracking을 통해 획득한 설문의 답과 concentration을 받고 이를 집중도로 환산하고 DB_Handler를 통해서 각 학생의 오라클 데이터베이스에 저장할 수 있도록 해준다. 또한 비디오 및 이미지로 제공되는 강의를 제공하고 학생 리스트에 대한 정보를 가지고 있다. 이 클래스는 유저, 즉 교수가 강의를 제공하고자 할 때 사용될 것이다.
- Real-time Survey: 실시간 설문을 전송하고 받을 수 있도록 해준다. 이 시스템에 대한 자세한 설명은 뒤에서 자세히 다루도록 하겠다.
- Eye-tracking class: eye-tracking 기술을 기반으로 하는 웹캠으로부터 실시간으로

얻을 수 있는 학생들의 집중도에 대한 데이터를 정수형태로 저장한다. 이 데이터는 오라클 데이터베이스에 저장된다.

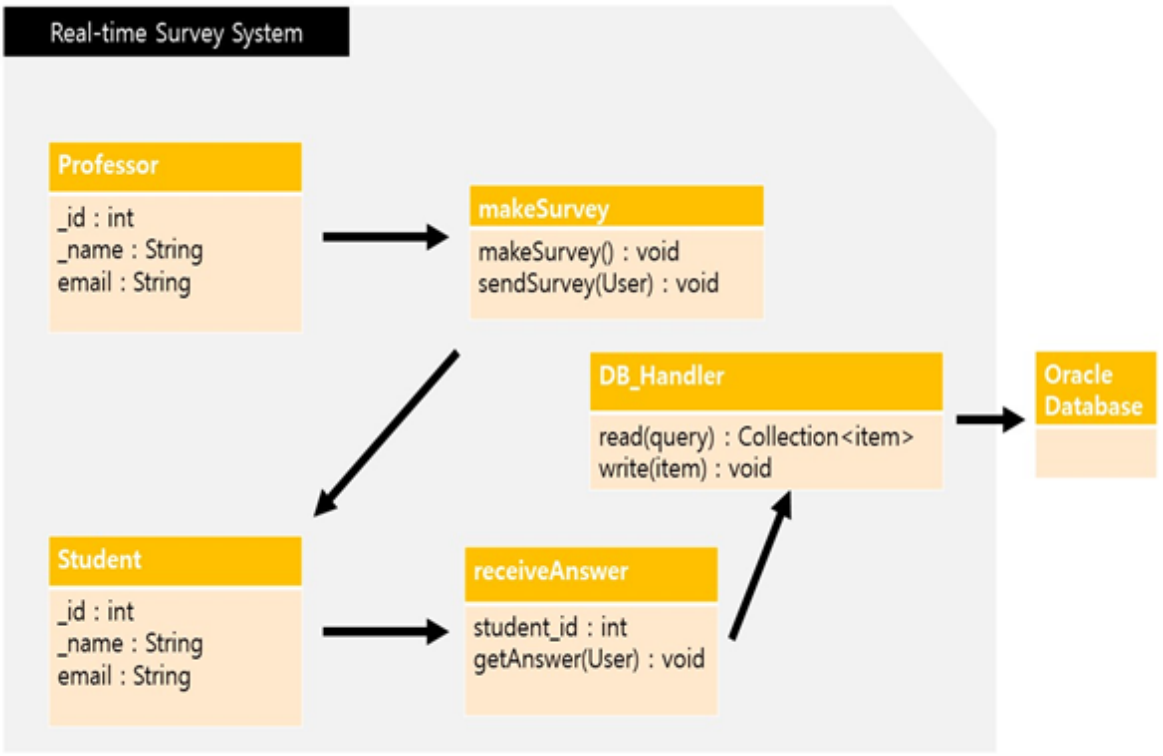
5.3.1.2. **Sequence Diagram**



[Figure 20] Sequence Diagram - Live Lecture Network system

5.3.2. Real-time Survey System

5.3.2.1. Class Diagram

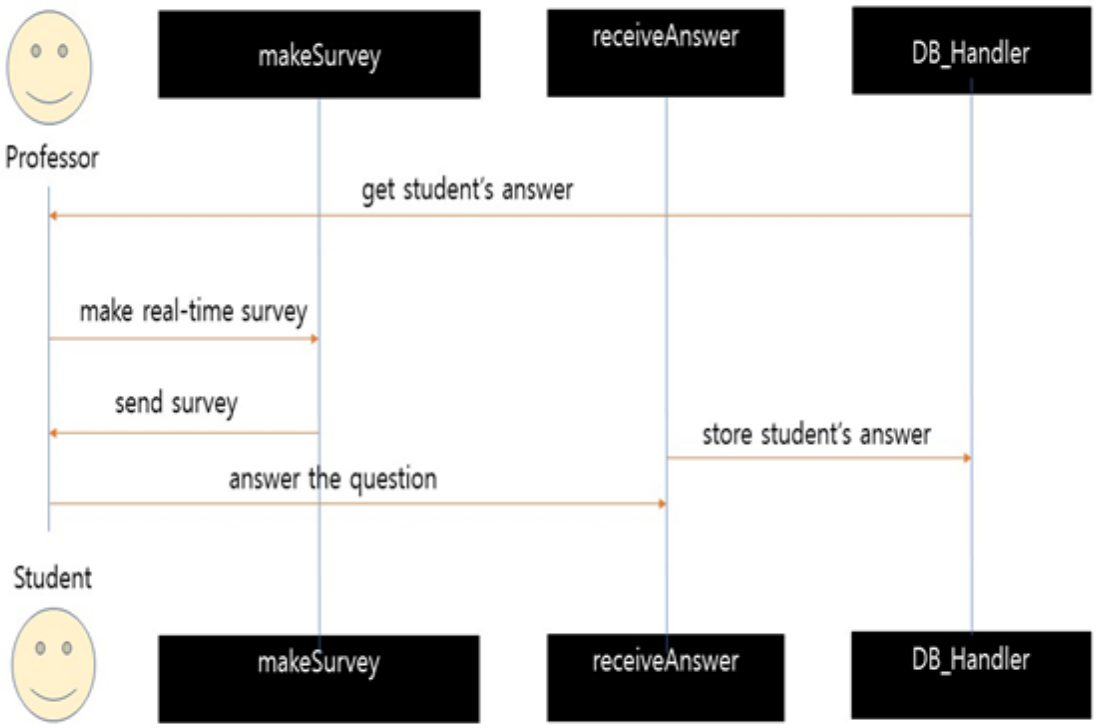


[Figure 21] Class diagram – Real-time Survey system

- Class Description

- makeSurvey Class: 교수가 수업 중에 학생들에게 설문을 보내고 싶을 때 이를 가능하도록 해준다. 설문을 다양한 텍스트, 선택형 등의 다양한 형태로 만들 수 있도록 해준다. 만들어진 설문은 학생에게 전송된다.
- receiveAnswer Class: 교수로부터 전송받은 설문에 대한 학생들의 답을 받아온다. 어떤 학생이 무슨 대답을 하였는 지에 대한 정보를 알 수 있도록 학번 데이터를 가지고 있으며 이렇게 수집된 학생들의 대답은 DB_Handler를 통해서 오라클 데이터베이스에 저장된다. 교수는 이 데이터베이스를 통해서 설문에 대한 학생들의 대답을 확인할 수 있다.

5.3.2.2. **Sequence Diagram**

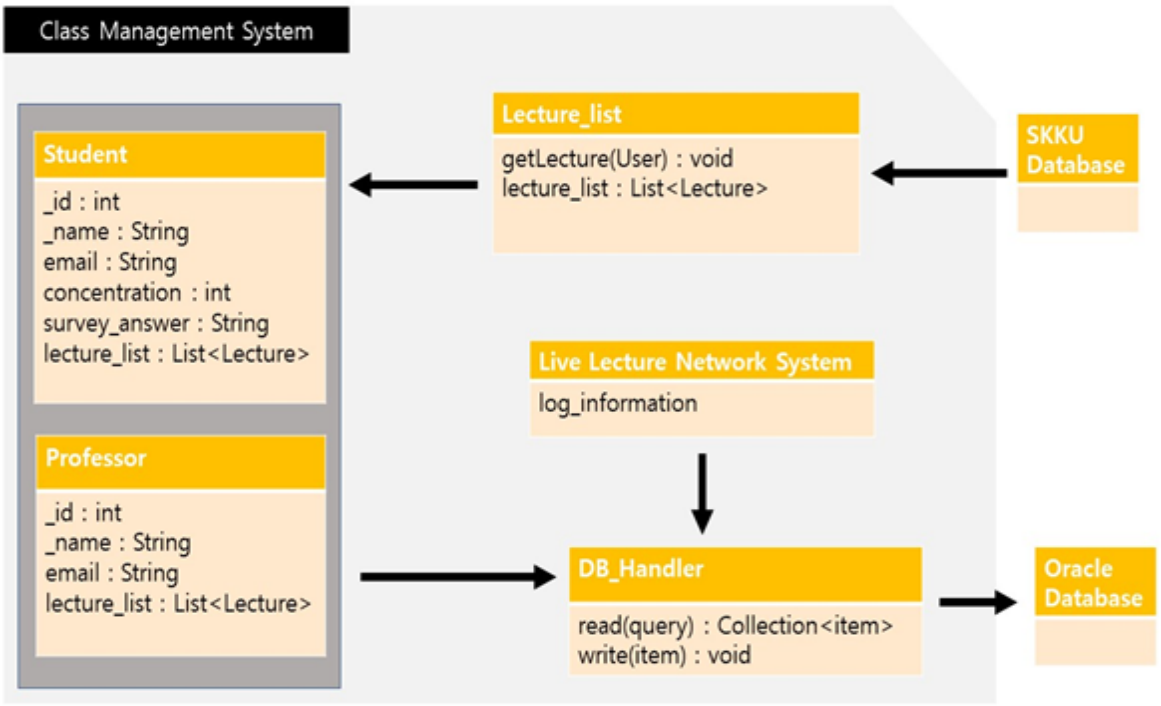


[Figure 22] Sequence diagram – Real-time Survey system

5.3.3. Class Management System

실시간 미팅룸 개설 및 집중도 체크 시스템 이외의 학생이나 교수와 같은 유저 및 강의 목록에 대한 데이터베이스를 관리하는 시스템에 대하여 추가적으로 설명하겠다.

5.3.3.1. **Class Diagram**

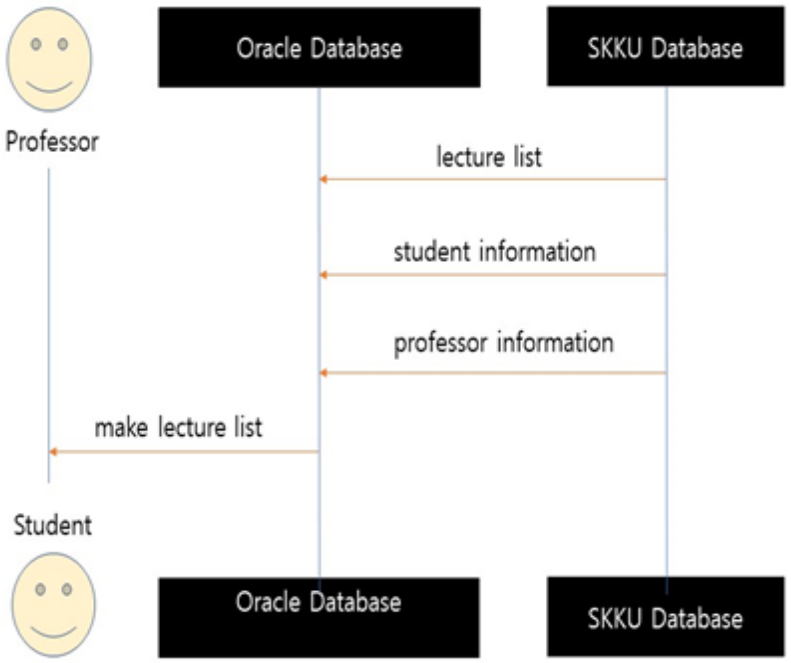


[Figure 23] Class diagram – Class Management system

- Class Description

- Lecture List class: 성균관대학교의 강의 데이터베이스에서 시스템을 이용하는 학생 및 교수가 현재 학기에 수강 및 강의 중인 수업에 대한 목록을 가져와서 리스트의 형태로 저장할 수 있도록 해준다. 얻어온 데이터는 학생과 교수의 강의 목록을 형성할 수 있도록 시스템의 데이터베이스, 즉 오라클 데이터베이스에 저장된다. 강의 목록은 추후에 본 시스템에서 제공하는 서비스를 이용하기 더 쉽도록 돕는다.

5.3.3.2. **Sequence Diagram**



[Figure 21] Sequence diagram – Class Management system

6. Protocol Design

6.1. Objectives

이 장에서는 각 서브 시스템들 사이에 사용된 프로토콜의 구조에 대해서 설명한다. 또한 각 인터페이스들이 어떻게 정의되었는지에 대해서 설명한다.

6.2. RDBMS

SKKU-MEET는 oracle의 데이터베이스 서비스를 이용한다. Oracle은 Relational Database Management System(이하 RDBMS)를 지원하는 데이터베이스로 SKKU-MEET는 모든 정보를 2차원 테이블의 형태로 데이터베이스에 저장한다. 수많은

강의들, 교수 및 학생들의 정보를 관리해야 하는 본 어플리케이션의 특성상 효율적으로 데이터를 저장, 구성 및 관리할 수 있는 RDBMS를 사용하게 되었습니다.

6.3. Authentication

6.3.1. Register

- Request

[Table 1] Table of register request

Attribute	Detail	
Method	POST	
URI	/register	
Parameter	Id	회원가입 요청된 아이디.
	Pass	회원가입 요청된 비밀번호.
	Type	회원가입 요청된 계정의 타입, 교수 혹은 학생.
	Affiliation	회원가입 요청된 계정의 소속 학교 정보.
Header		

- Response

[Table 2] Table of register response

Attribute	Detail
Success Code	200 OK
Failure Code	HTTP error code = 400

Success Response Body	Access Token	액세스 토큰
	Message	회원가입 성공 메시지
Failure Response Body	Message	회원가입 실패 메시지

6.3.2. Sign In

- Request

[Table 3] Table of sign in request

Attribute	Detail	
Method	POST	
URI	/signin	
Parameter	Id	로그인 요청된 아이디
	Pass	로그인 요청된 비밀번호
Header		

- Response

[Table 4] Table of sign in response

Attribute	Detail
Success Code	200 OK

Failure Code	HTTP error code = 400	
Success Response Body	Access Token	액세스 토큰
	Message	로그인 성공 메시지
Failure Response Body	Message	로그인 실패 메시지

6.4. Class list

6.4.1. View class list

- Request

[Table 5] Table of view class list request

Attribute	Detail	
Method	GET	
URI	/user/:id/classlist	
Parameter	User	사용자의 정보
Header	Authorization	액세스 토큰

- Response

[Table 6] Table of view class list response

Attribute	Detail

Success Code	200 OK	
Failure Code	HTTP error code = 403 (Forbidden)	
Success Response Body	Class list	사용자가 운영/수강 중인 강의들의 목록 및 정보.
Failure Response Body	Message	강의 리스트 불러오기 실패 메시지.

6.5. Meeting Room

6.5.1. Create meeting room

- Request

[Table 9] Table of create meeting room request

Attribute	Detail	
Method	POST	
URI	/user/:id/create	
Parameter	Class_id	미팅룸 생성 요청된 강의의 id값.
Header	Authorization	액세스 토큰

- Response

[Table 10] Table of create meeting room response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 403 (Forbidden)	
Success Response Body	Meeting_Room	생성된 미팅룸의 url을 비롯한 정보.
Failure Response Body	Message	미팅룸 생성 실패 메시지

6.5.2. Join meeting room

- Request

[Table 9] Table of join meeting room request

Attribute	Detail	
Method	POST	
URI	/user/:id/join	
Parameter	Class_id	미팅룸 참여 요청된 강의의 id값.
Header	Authorization	액세스 토큰

- Response

[Table 10] Table of join meeting room response

Attribute	Detail

Success Code	200 OK	
Failure Code	HTTP error code = 403 (Forbidden)	
Success Response Body	Meeting_Room	참가 요청된 강의의 미팅룸 url 및 정보.
Failure Response Body	Message	미팅룸 참여 실패 메시지.

6.6. Real-time survey

6.6.1. Start survey

- Request

[Table 11] Table of start survey request

Attribute	Detail	
Method	Post	
URI	/:meetingroom/startsurvey	
Request Body	Survey_Info	실시간 설문의 내용이 담긴 정보
Header	Authorization	액세스 토큰

- Response

[Table 12] Table of start survey response

Attribute	Detail

Success Code	200 OK	
Failure Code	HTTP error code = 403 (Forbidden)	
Success Response Body	Message	실시간 설문 시작 성공 메시지
Failure Response Body	Message	실시간 설문 시작 실패 메시지

6.6.2. Answer survey

- Request

[Table 13] Table of answer survey request

Attribute	Detail	
Method	POST	
URI	/:meetingroom/:survey/answer	
Parameter	Answer	실시간 설문에 대한 응답
Header	Authorization	액세스 토큰

- Response

[Table 14] Table of answer survey response

Attribute	Detail
Success Code	200 OK

Failure Code	HTTP error code = 400 (Bad Request, overlap)	
Success Response Body	Message	실시간 설문 응답 성공 메시지
Failure Response Body	Message	실시간 설문 응답 실패 메시지

6.6.3. End survey

- Request

[Table 15] Table of end survey request

Attribute	Detail	
Method	POST	
URI	/:meetingroom/:survey/end	
Request Body	Survey_Info	종료 요청된 실시간 설문에 대한 정보.
Header	Authorization	액세스 토큰

- Response

[Table 16] Table of end survey response

Attribute	Detail
Success Code	200 OK
Failure Code	HTTP error code = 403 (Forbidden)

Success Response Body	Message	실시간 설문 종료 성공 메시지
Failure Response Body	Message	실시간 설문 종료 실패 메시지

6.7. Concentration Report

6.7.1. View report

- Request

[Table 17] Table of view report request

Attribute	Detail	
Method	GET	
URI	/user/:id/classlist/report	
Parameters	Class_id	집중도 리포트.
Header	Authorization	User authentication

- Response

[Table 18] Table of view report response

Attribute	Detail
Success Code	200 OK
Failure Code	HTTP error code = 404 (Not Found)

Success Response Body	Report	조회 요청 받은 집중도 리포트의 내용.
Failure Response Body	message	집중도 리포트 조회 실패 메시지.

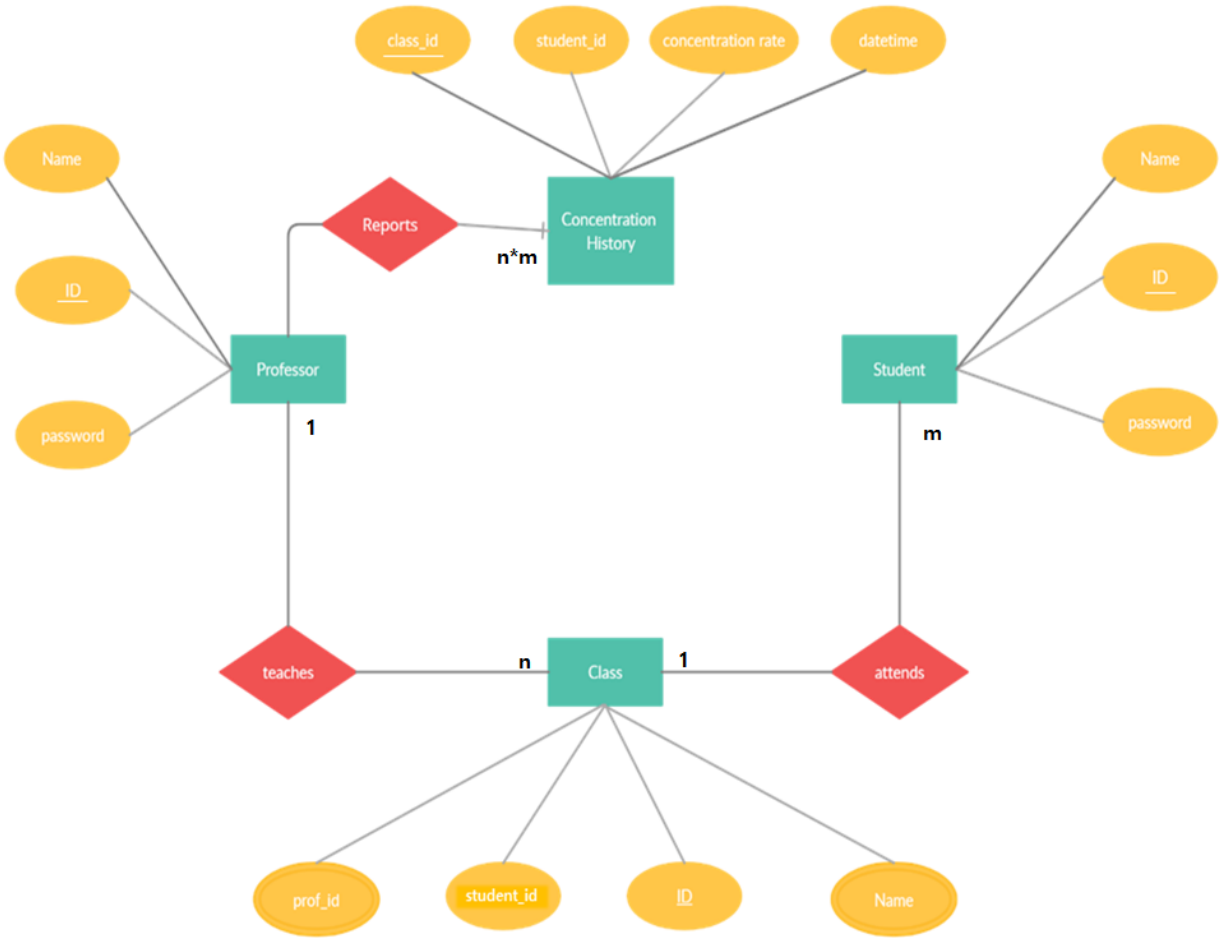
7. Database Design

7.1. Objectives

7장에서는 시스템 데이터 구조와 이러한 구조가 데이터베이스로 어떻게 구현되었는지에 대해 설명한다. 먼저 ER 다이어그램(Entity Relationship diagram)을 통해 entity와 그 관계를 식별한다. 그런 다음 관계형 스키마 및 SQL DDL(Data Definition Language)을 작성한다.

7.2. ER Diagram

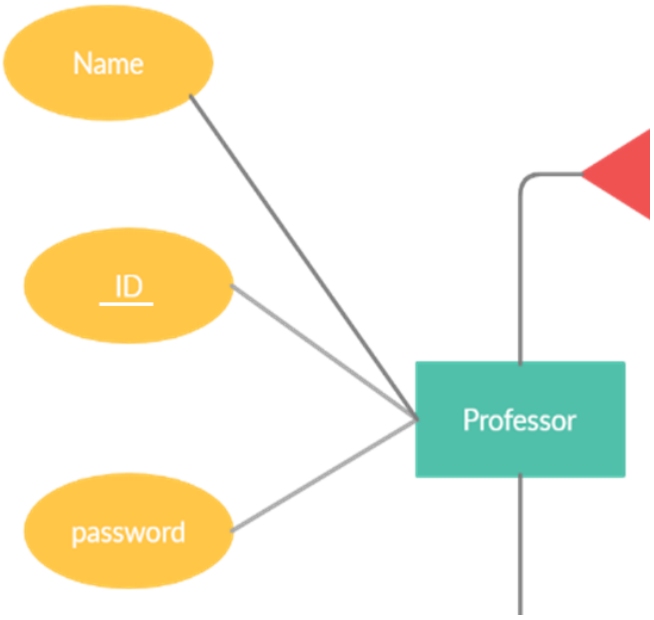
본 어플리케이션 시스템은 총 4가지 entity로 이루어져 있다; Professor, Student, Class, Concentration_History. ER-Diagram은 entity간의 관계, 그리고 entity와 attribute의 관계를 다이어그램으로 설명한다. 각 entity의 primary key는 밑줄로 표시되어 있다. 각 entity마다 대응되는 개수는 entity를 연결하는 선 주변에 표기되어 있어 확인 가능하다.



[Figure 25] ER-diagram

7.2.1. Entities

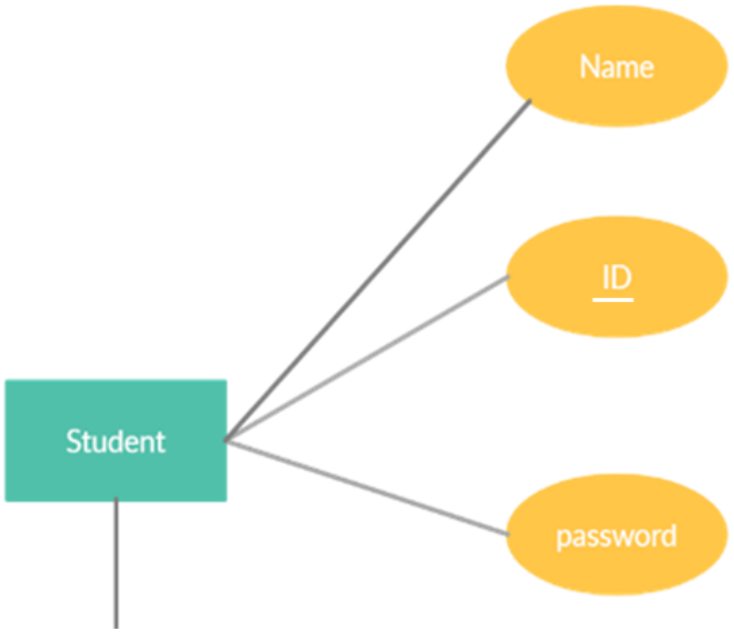
7.2.1.1. **Professor**



[Figure 26] ER diagram, Entity, Professor

Professor entity는 SKKU:MEET 사용자 중 교수에 해당한다. Professor entity는 name, ID 그리고 password를 attribute로 가지며, ID가 primary key이다. 이 attribute는 애플리케이션 회원 가입할 때 사용자들이 기입한 정보이다. 또한 professor entity는 class와 concentration_history entity와 연관이 있다.

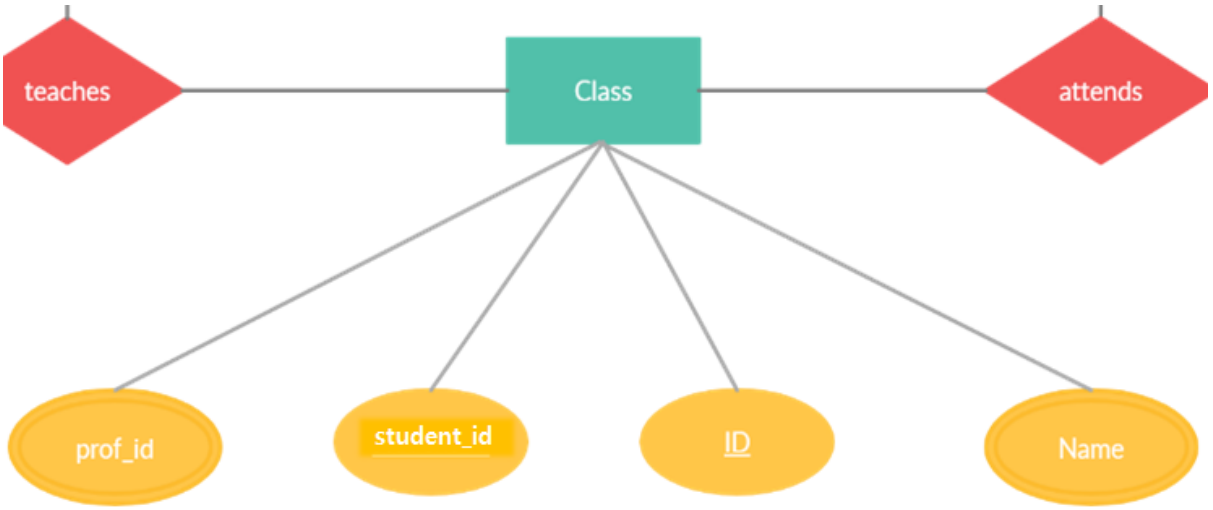
7.2.1.2. **Student**



[Figure 27] ER diagram, Entity, Student

Student entity는 SKKU:MEET 사용자 중 수업을 듣는 학생에 해당한다. Student entity는 name, ID 그리고 password를 attribute로 가지며, ID가 primary key이다. 이 attribute는 애플리케이션 회원 가입할 때 사용자들이 기입한 정보이다. 또한 student entity는 class와 concentration_history entity와 연관이 있다.

7.2.1.3. **Class**

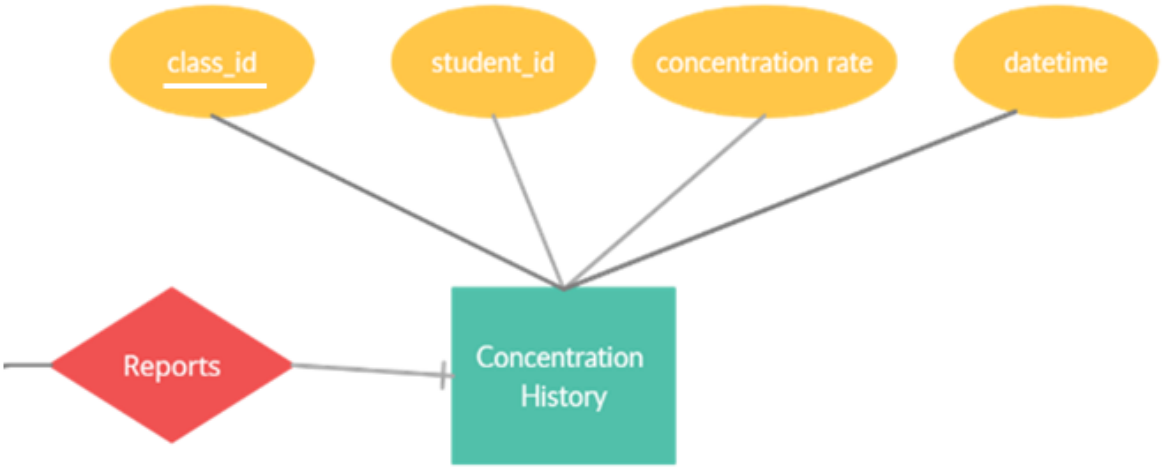


[Figure 28] ER diagram, Entity, Class

Class entity는 교수가 강의중인 수업에 해당한다. 4개의 attributes로 이루어져 있다: prof_id, student_id, id, 그리고 name이며 ID가 primary key이다. 또한, class는 professor, student 그리고 concentration_history entity.와 관계가 있다. prof_id와 student_id는 foreign key로 Professor.ID와 Student.ID에서 참조한다.

.

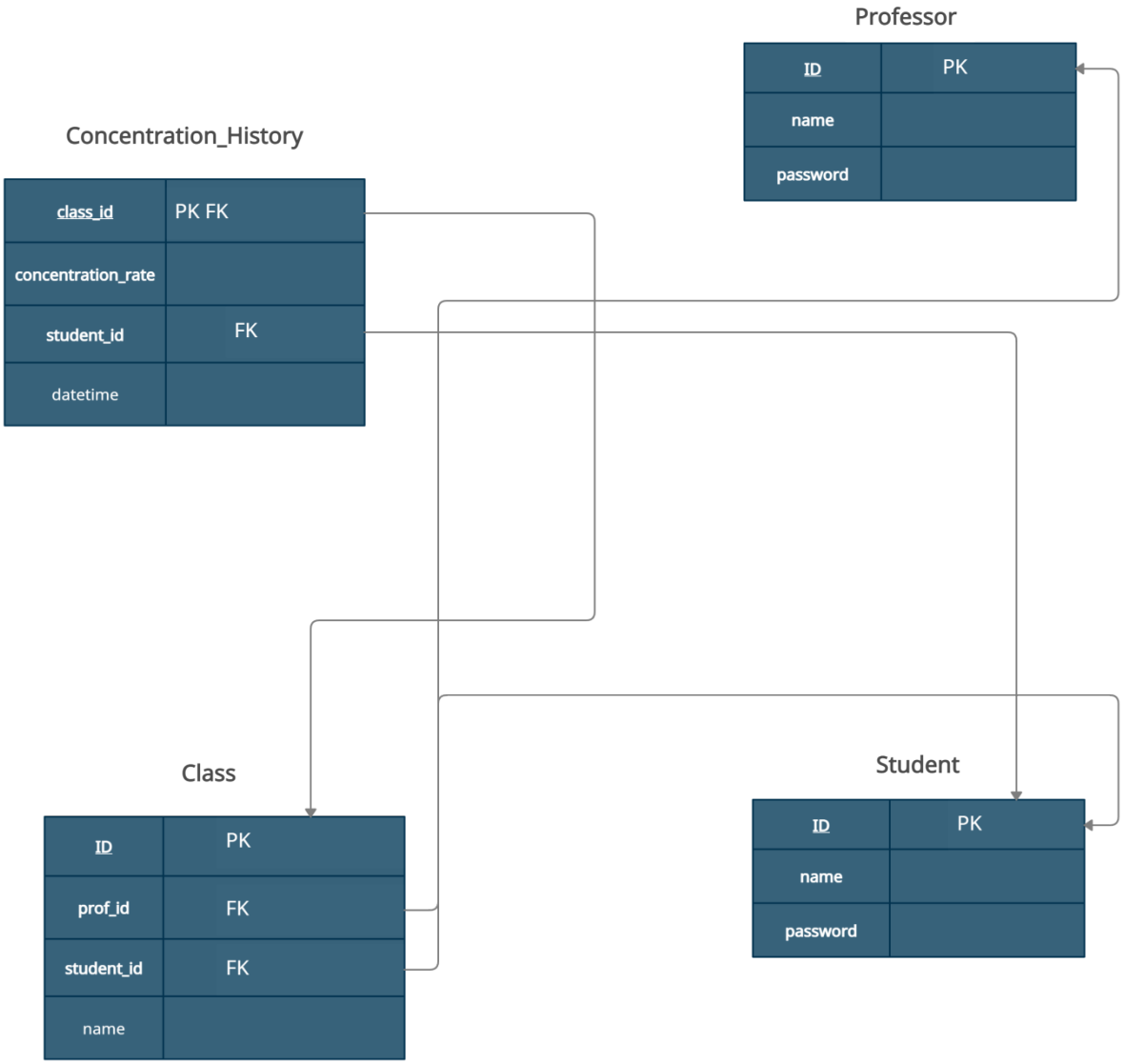
7.2.1.4. **Concentration_History**



[Figure 29] ER diagram, Entity, Concentration_History

Concentration_History entity는 교수가 강의중인 수업에 해당한다. 4개의 attributes로 이루어져 있다: class_id, student_id, concentration_rate 그리고 date_time이며 class_id가 primary key이다. 또한, concentration_history는 student과 class entity와 관계가 있다. class_id와 student_id는 foreign key로 Class.ID와 Student.ID에서 참조한다.

7.3. Relational Schema



[Figure 30] Relational Schema

7.4. SQL DDL

7.4.1. Professor

```
CREATE TABLE Professor
(
 ID INT NOT NULL,
 name INT NOT NULL,
 password INT NOT NULL,,
 PRIMARY KEY (ID)
  )) Engine=InnoDB;
```

7.4.2. Student

```
CREATE TABLE Student
(
 ID INT NOT NULL,
 name INT NOT NULL,
 password INT NOT NULL,
   PRIMARY KEY (ID)
  )) Engine=InnoDB;
```

7.4.3. Class

```
CREATE TABLE Class
(
 ID INT NOT NULL,
 FOREIGN KEY (student_id) REFERENCES Student(ID),
  FOREIGN KEY (prof_id) REFERENCES Professor(ID)
  name INT,
  PRIMARY KEY(ID)) Engine=InnoDB;
```

7.4.4. Concentration_History

```
CREATE TABLE Concentration_History
(
 date_time datetime NOT NULL,
 class_id INT NOT NULL,
  FOREIGN KEY (class_id) REFERENCES Class(ID),
  FOREIGN KEY (student_id) REFERENCES Student(ID),
  PRIMARY KEY(class_id)
  )) Engine=InnoDB;
```

8. Testing Plan

8.1. Objectives

8장에서는 개발 테스트, 릴리스 테스트 및 사용자 테스트의 세 가지 주요 하위 그룹을 포함하는 테스트에 대한 계획을 설명한다. 이러한 테스트는 어플리케이션의 잠재적인 오류와 결함을 감지하고 애플리케이션의 완성도를 높이고 안정적인 시스템을 보장하기 위해 총 세 가지 테스팅을 실행한다.

8.2. Testing Policy

8.2.1. Development Testing

Development Testing은 주로 소프트웨어 개발 과정에서 광범위하게 발생하는 잠재적 오류를 줄이고 시간과 비용을 절약하기 위해 수행된다.
이 단계에서는 소프트웨어가 테스트 단계에 있기 때문에 불안정할 수 있으며, 구성 요소(backend 시스템, frontend 시스템 등)가 서로 충돌하여 제대로 작동하지 않을 수 있다. 따라서 이 단계에서는 static 코드 분석, 데이터 흐름 분석, 개발팀원간 코드 검토, 단계별 테스트를 수행해야 한다. 이러한 프로세스를 통해 우리는 주로 소프트웨어의 성능과 신뢰성, 그리고 보안을 보장하게 된다.

8.2.1.1. Performance

화상 강의에 사용되는 본 시스템의 특성상 사용자들 간의 동영상 공유에 소모되는 지연시간이 5초 미만이어야 하며, eye-tracking의 역할이 학생의 이해도를 실시간으로

제공하는 것인 만큼 관련된 연산에 소모되는 시간 또한 5초를 넘어서는 안 된다. AWS 서버의 성능을 사전에 확인하고 실제 테스트를 다양한 시간 동안 여러 번 수행하여 99% 지연시간(latency)가 5초 미만인지 확인한다.

8.2.1.2. **Reliability**

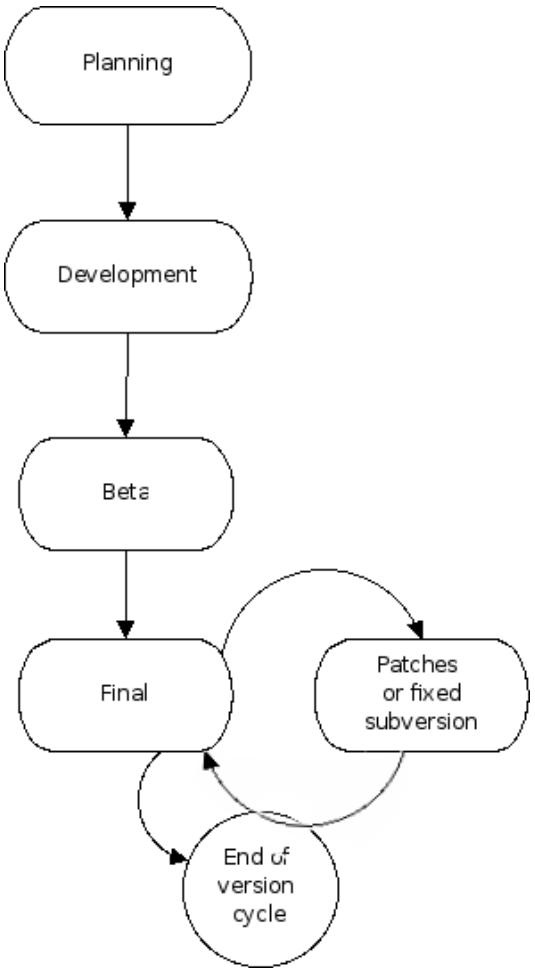
시스템이 고장 없이 안전하게 작동하려면 먼저 시스템을 구성하는 하위 구성 시스템과 장치가 정상적으로 작동한 뒤, 전체 시스템으로 연결되어야 한다. 따라서 하위 시스템 개발 단계부터 개발 테스트를 거쳐 각 하위 시스템이 전체 시스템에 통합되는 동안 오류를 반복적으로 확인하고 그에 따라 수정해야 한다.

8.2.1.3. **Security**

사용자들은 시스템을 이용하기 전에 올바른 절차를 통해 인증을 받고 시스템에 접속하기 위한 계정을 생성해야 한다. 사용자들 중에 교수, 조교 등 미팅룸을 생성하는 권한이 필요한 사용자는 회원 가입 단계에서 적합한 인증을 받도록 하는데, 인증 후 권한이 제대로 부여되는지 테스트한다. 일반 사용자(학생 권한)는 시스템 관리자만큼의 권한을 가질 수 없고, 시스템을 통하는 방법 외에는 DB에 직접적으로 접근하여 데이터를 획득하는 등의 작업을 할 수 없는지 권한을 체크한다. 또한 전체 사용자의 개인 정보가 담긴 DB는 시스템 관리자만 접근할 수 있도록 권한 설정이 되었는지 확인한다.

8.2.2. Release Testing

소프트웨어 개발 프로젝트의 가장 중요한 단계 중 하나는 제품을 출시하는 단계이다. 잘못된 릴리스 방식으로 인해 잘 작동하던 시스템도 오류가 발생할 수 있다. 따라서, 제품과 시장 간의 더 나은 연결을 위해 릴리스 테스트가 불가피하다. 릴리스 테스트는 소프트웨어/애플리케이션의 업데이트된 버전을 테스트하여 소프트웨어가 완벽한 상태로 릴리스되어 작동에 아무런 하자가 없는지 확인하는 작업이며, 실제 릴리스 전에 수행해야 한다.
소프트웨어 릴리스 수명 주기에 따라 테스트는 일반적으로 소프트웨어의 기본 구현이 완료된 '알파' 버전에서 시작된다. 알파 버전에서 개발 테스트를 시작하고 사용자 및 릴리스 테스트를 포함한 추가 테스트를 위해 베타 버전을 릴리스 할 예정이다. 베타 버전이 출시되면 개발자는 물론 실제 사용자로부터 피드백을 받을 예정이다.



[Figure 31] Software Release Life Cycle

8.2.3. User Testing

사용자 테스트를 진행할 수 있는 가능한 시나리오와 상황을 현실적으로 설정해야 한다. 본 애플리케이션을 사용하는 사용자를 50명을 모아 해당 베타버전 사용자에게 베타 버전을 배포하고 안드로이드 시뮬레이터로 자체 사용자 테스트를 수행하면서 사용자 리뷰를 수집할 예정이다.

8.2.4. Testing Case

Test case는 기능, 성능 그리고 보안 세 가지 측면을 중심으로 설계된다. 각 측면에서 최소 5개의 test case를 만들고 본 애플리케이션에 대한 테스트를 진행하며 평가 보고서를 작성한다.

9. Development Plan

9.1. Objectives

해당 챕터에서는 시스템의 개발 환경 및 기술에 대하여 설명한다..

9.2. Frontend Environment

9.2.1. Java Server Pages

[Figure 32] Java Server Pages logo

JSP는 JAVA를 이용한 서버 사이드 스크립트 언어로 HTML 코드에 JAVA 코드를 넣어 동적 웹페이지를 생성하는 웹 어플리케이션 도구이다. JSP가 실행되면 자바 서블릿(Servlet)으로 변환되며 웹 어플리케이션 서버에서 동작되면서 필요한 기능을 수행하고 그렇게 생성된 데이터를 웹페이지와 함께 클라이언트로 응답하게 된다. JSP를 호스팅 해주는 업체는 찾기 어려운 편이지만 AWS (Amazon Web Services)에서 이를 지원한다. 따라서 뒤 부분의 backend environment에서 소개하겠지만 우리는 JSP를 지원해주는 AWS를 사용할 것이다. JSP는 유지보수가 쉬우며 안정적이라는 장점이 있지만 무겁고 느리다는 단점이 존재한다.

9.2.2. Hypertext Preprocessor

[Figure 33] Hypertext Preprocessor logo

PHP는 대표적인 서버 사이드 스크립트 언어로 전 세계 수많은 웹 시스템의 기반이 되는 언어이다. 현재 The PHP Group이라는 단체에서 개발 및 관리를 맡고 있다. C-like 문법을 사용하며 소규모 웹 페이지를 제작할 때 기본적으로 웹 관련 함수들이 많아 생산성이 높다는 점에서 사용자가 많다. 본 프로젝트의 SKKU:MEET 웹 어플리케이션을 개발하기 위하여 JSP와 함께 도구로 사용될 것이다.

9.2.3. HyperText Markup Language

[Figure 34] HyperText Markup Language logo

HTML은 웹 페이지를 위한 지배적인 마크업 언어이며 W3C가 HTML과 CSS 표준의 공동 책임자이다. HTML을 통해 제목, 단락, 목록 등 본문을 위한 구조적 의미를 나타내는 것뿐만 아니라 링크, 인용과 그 밖의 항목으로 구조적 문서를 만들 수 있는 방법을 제공받을 수 있다. 이는 태그로 되어있는 HTML요소 형태로 작성되며 웹

브라우저와 같은 HTML 처리 장치의 행동에 영향을 주는 자바스크립트를 포함하거나 불러올 수 있다. 따라서 우리는 SKKU:MEET 웹 어플리케이션을 구현할 때 각각의 웹 페이지를 위해서 HTML을 사용할 것이다.

9.2.4. Adobe Photoshop

[Figure 35] Adobe Photoshop logo

Adobe Photoshop은 어도비 시스템즈에서 개발 및 배급된 레스터 그래픽 편집기이다. 이 프로그램을 통해서 유저에게 편의성을 제공할 수 있도록 SKKU:MEET의 레이아웃 및 여러 아이콘을 디자인하고 그래픽 형태로 만들어 사용할 수 있다.

9.2.5. Adobe Xd

[Figure 36] Adobe Xd logo

Adobe XD는 어도비 시스템즈에서 개발 및 배급된 사용자 체험 디자인 소프트웨어이다. 벡터 디자인 및 와이어프레임, 그리고 클릭을 통한 단순한 상호작용 프로토타입 제작을 지원한다. 이를 통해 팀원 간의 커뮤니케이션 및 피드백의 반영을

가능하게 해준다는 장점이 있다.

9.3. Backend Environment

9.3.1. Amazon AWS

[Figure 37] Amazon AWS logo

AWS는 아마존닷컴의 클라우드 컴퓨팅 사업부이다. AWS는 다른 웹 사이트나 클라이언트 측 응용 프로그램에 대해 온라인 서비스를 제공하고 있으며 어플리케이션 서비스 및 모바일 서비스 역시 포함된다. AWS에서 제공하는 퍼블릭 클라우드 컴퓨팅 서비스는 물리 장비 구매나 임대 계약 없이도 사용 가능하고 원하는 시간 동안 원하는 만큼 컴퓨팅 자원을 사용할 수 있으며 스케일 업/다운, 스케일 인/아웃이 자유롭게 가능하다는 특징이 있다. AWS를 통해 우리는 SKKU:MEET의 실시간 수업을 시작하기 위한 미팅룸을 개설할 수 있다.

9.3.2. Oracle Database

[Figure 38] Oracle database logo

Oracle Database는 미국 오라클 사에서 판매하는 관계형 데이터베이스 관리 시스템이다. 현재 유닉스 환경에서 가장 널리 사용되는 RDBMS이며 검색이나 업데이트용 언어로는 국제표준화기구의 표준 구조화 조회 언어와 PL/SQL을 지원한다.

본 프로젝트에서는 오라클 데이터베이스 서비스를 이용해서 시스템 사용자들의 계정 정보 및 집중도 데이터를 관리할 것이다.

9.4. Constraints

시스템은 본 문서에서 언급된 내용들에 기반하여 디자인 및 구현될 것이다. 많은 고려해야 할 제약사항이 있고 이를 나열하면 다음과 같다.

- 기존에 배포된 eye-tracking 기술을 사용한다.
- eye-tracking을 통해 사용자의 집중도를 계산하는 시간 및 결과를 데이터베이스에 저장하는 시간은 5초를 넘으면 안 된다.
- 로열티를 지불해야 하거나 separate license를 요구하는 기술 및 소프트웨어의 사용은 지양한다.
- 전반적인 시스템 성능을 향상시킬 수 있는 방향으로 개발을 진행한다.
- 유저 친화적이며 시스템을 사용하기 편하도록 개발을 진행한다.
- 가능한 오픈소스 소프트웨어를 사용한다.
- 시스템 비용과 유지보수 비용에 대해 고려하며 개발을 진행한다.
- 시스템 자원의 낭비를 막도록 소스 코드를 최적화한다.
- 소스 코드를 작성할 때 유지보수에 대하여 충분히 고려하며 이를 주석으로 남겨놓는다.
- 개발은 최소 안드로이드 6.0 버전 및 윈도우 7 이상에서 이루어져야 하며 각각 안드로이드 10, 윈도우 10 환경을 타겟으로 한다.
- 안드로이드 10 버전, 윈도우 10 버전에서 emulate한다.

9.5. Assumptions and Dependencies

본 문서의 모든 시스템은 안드로이드 장치 및 데스크탑 환경에 기반하여 디자인 및 구현되었다고 가정하며 작성되었다. 또한 eye-tracking 기술이나 여러 시스템들은 오픈소스를 활용하여 구현된다고 가정하고 있다. 본 문서의 모든 내용은 안드로이드 버전 6.0 및 윈도우 7 이상의 OS 환경을 기반으로 하여 작성되었으며 따라서 다른 OS나 조건을 만족하지 않는 버전에서 시스템은 지원되지 않는다.

10. Supporting Information

10.1. Software Design Specification

이 소프트웨어 디자인 명세서는 IEEE 권장 사항(IEEE Recommended Practice for Software Design Description, IEEE-Std-1016)에 따라 작성되었다.

10.2. Document History

[Table 19] Document History

Date	Version	Description	Writer
2020/05/03	0.1	Style and overview	Lee, Bohyun
2020/05/05	1.0	Addition of 1, 2	Lee, Jaehun
2020/05/06	1.1	Addition of 7	Lee, Bohyun
2020/05/07	1.2	Addition of 6	Rim, Seungjae
2020/05/09	1.3	Addition of 5, 9	Jo, Hyojung
2020/05/11	1.4	Addition of 3, 4	Kang, Byeongnam
2020/05/12	1.5	Addition of 8	Lee, Bohyun
2020/05/13	1.6	Revision of 2.1, 2.2	Lee, Jaehun
2020/05/13	1.7	Revision of 3.2, 4.2	Kang, Byeongnam
2020/05/14	1.8	Revision of 9.3	Jo, Hyojung
2020/05/15	1.9	Addition of 10	Rim, Seungjae