SKKU Exam Manager

Software Requirement Specification

2021. 04. 25.

Introduction to Software Engineering 41

Team 2 (SKKUEM)

Team Leader	Kyunghee Ko
Team Member	Luke (Yeongseung An)
Team Member	Jeongmin Lee
Team Member	Eunsu Kang
Team Member	Seungyeon Cho
Team Member	Hyungjun Joo

Contents

List of Figures

List of Tables

1. Preface

This chapter contains information about the readership, scope, objective, and structure for the Software Design Document that would be used for this project.

1.1. Readership

This document is composed of 10 sections, each with its subsections. The structure for this document is found in Section 1.4. Team 2 is the main reader for this document, but students, professors, and TAs of the Introduction to Software Engineering course can also be one.

1.2. Scope

This document is used to provide descriptions of the designs that would be used to implement the test-taking system for courses that offer their exams online.

1.3. Objective

The main objective of this Software Design Specification document is to describe the design aspects for the SKKU Exam Manager. This document describes the software architecture and design decisions. Also, it depicts the structure and design of functions also referred to in the Software Requirement Specification document with some use cases and diagrams.

1.4. Document Structure

1. Preface: The chapter being read right now; provides information about the readership, scope, objective, and structure for this document

2. Introduction: Provides definitions and descriptions of diagrams and tools used for this project; also provides the scope and references that may be necessary to understand this document

3. System Architecture - Overall: Provide information on how the system is organized

4. System Architecture - Frontend: Provides the conceptual model that defines the structure, behavior, and views of the frontend of a system

5. System Architecture - Backend: Provides the conceptual model that defines the structure, behavior, and views of the backend of a system

6. Protocol Design: Provides descriptions of the structure for the protocol used for the interaction of subsystems and the protocol for defining interfaces.

7. Database Design: Provides descriptions of the system data structures and how said data structures are to be represented in a database.

8. Testing Plan: Provides plans for development testing, release testing, and user testing.

9. Development Plan: Provides information on the development environment and tools used during development.

10. Supporting Information: Provides document history.

2. Introduction

The project is to develop and design a mobile application and web service used to complement courses that require the exams to be taken online due to the current pandemic. The program should allow students to use a one-to-one test-taking system with cheat detection and recording capability. The designs that would be used when implementing the system would be provided in this document and would follow the requirements specified in the Software Requirements Specifications document provided before.

2.1. Objectives

The objective of this chapter is to provide descriptions and definitions of the tools and diagrams used for this project during the design phase. The definitions for the program are provided by Visual Paradigm, Creately, and EdrawMax, which are all tools used to create UML diagrams.

2.2. Applied Diagrams

This subsection aims to provide definitions for UML and the types of UML diagrams used for this project.

2.2.1. UML

According to Visual Paradigm, UML (Unified Modeling Language) is a "general-purpose, developmental, modeling language in the field of software engineering, that is intended to provide a standard way to visualize the design of a system". In other words, it is a way to model and document software that is commonly used by business analysts, and developers as a modeling technique to visualize the design of a system that makes it easier to understand potential flaws with the system when it comes to software or business processes.

2.2.2. Use Case Diagram

According to Visual Paradigm, use case diagrams are "a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved…[It] is the primary form of system/software requirements for a new software program underdeveloped". In order words, it serves as a simplified guide for what the system should do, and does not denote the order in which steps are performed, and "only summarizes some of the relationships between use cases, actors, and systems."

2.2.3. Class Diagram

According to Visual Paradigm, class diagrams are "a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationship among objects." In other words, they are the cornerstone of object-oriented modeling as they provide the static structure of classifiers in a system through diagrams that consist of a set of classes and a set of relationships between classes, which help developers and other team members when working on the system.

2.2.4. Sequence Diagram

According to Visual Paradigm, sequence diagrams are “interaction diagrams that detail how operations are carried out…[that] capture the interaction between the objects in the context of a collaboration.” They depict high-level interactions that occur between sub-systems, systems, or the system and its user that either visualize a use case or an operation.

2.2.5. Context Diagram

According to EdrawMax, context diagrams are used to show the “flow of information between the system and external components.” They are also the highest or most basic level of a Data Flow Diagram, which is why context diagrams are sometimes referred to as Level 0 Data Flow Diagrams, which are diagrams that tries to document how information flows within a system as a whole by specifying the processes that are involved with transferring data.

2.2.6. Entity Relationship Diagram

According to Creately, an Entity Relation Diagram (ERD) is a way to visually represent how various entities within a system relate to each other. Entities can be any type of business objects used in the system, such as people, roles, events, places, products, and logs.

3. Applied Tools

3.1. StarUML

According to their website, StarUML is a "sophisticated software modeler for agile and concise modeling" with UML2 and SysML Support. It is a convenient way to create diagrams and models.

3.1.1. diagrams.net

According to their website, diagrams.net is a security-first diagramming tool for teams that focuses on privacy.

3.2. Project Scope

This service was made to accommodate non-face-to-face lectures that were conducted online due to the current pandemic as an online test program that provides a UI that is easy to use and a cheat detection system using open-source AI. There will also be a database server that would store the recorded videos, exam papers, and the answers of the students, which would interact with the university database.

3.3. References

[1] "Chapter 5. UML & Requirement Diagram," *Visual Paradigm Community Circle*. https://circle.visual-paradigm.com/docs/uml-and-requirement-diagram/.
[2] Visual Paradigm, "What is Entity Relationship Diagram (ERD)?," *Visual Paradigm*, 2019. https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/.
[3] N. Silva, "ER Diagram Tutorial | Complete Guide to Entity Relationship Diagrams," *Creately Blog*, Apr. 30, 2019. https://creately.com/blog/diagrams/er-diagrams-tutorial/.
[4] "What is a Context Diagram – Explain with Examples," *Edrawsoft*. https://www.edrawmax.com/context-diagram/.
[5] "Diagram Software and Flowchart Maker," *diagrams.net*. https://www.diagrams.net/.
[6] "StarUML," *StarUML*. https://staruml.io/.
[7] "What is Sequence Diagram?," *Visual Paradigm*, 2019. https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-sequence-diagram/.

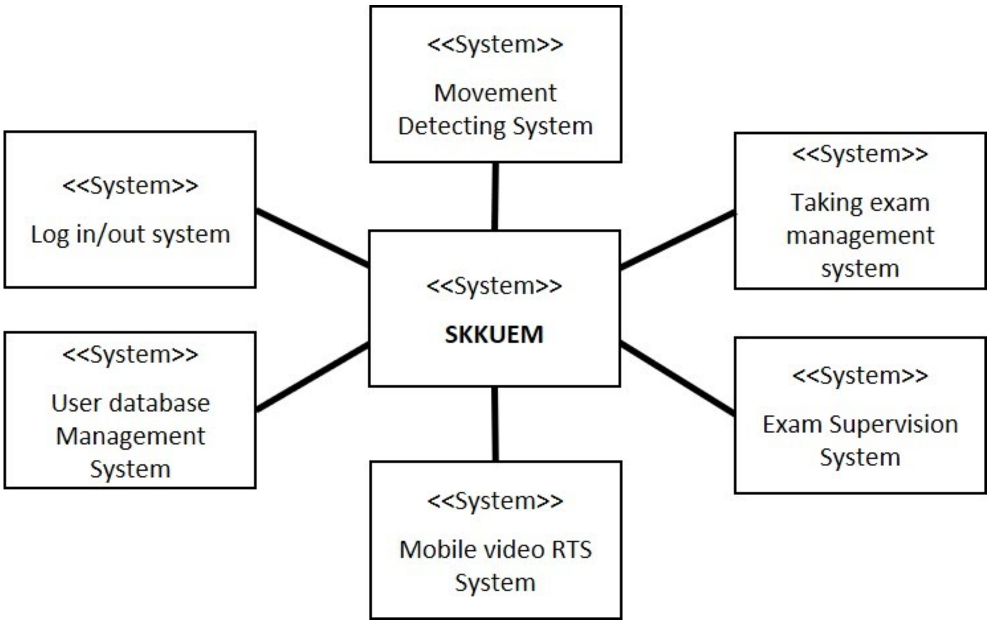
4. System Architecture – Overall

4.1. Objectives

The objective of this chapter is to provide information on how the system is organized.

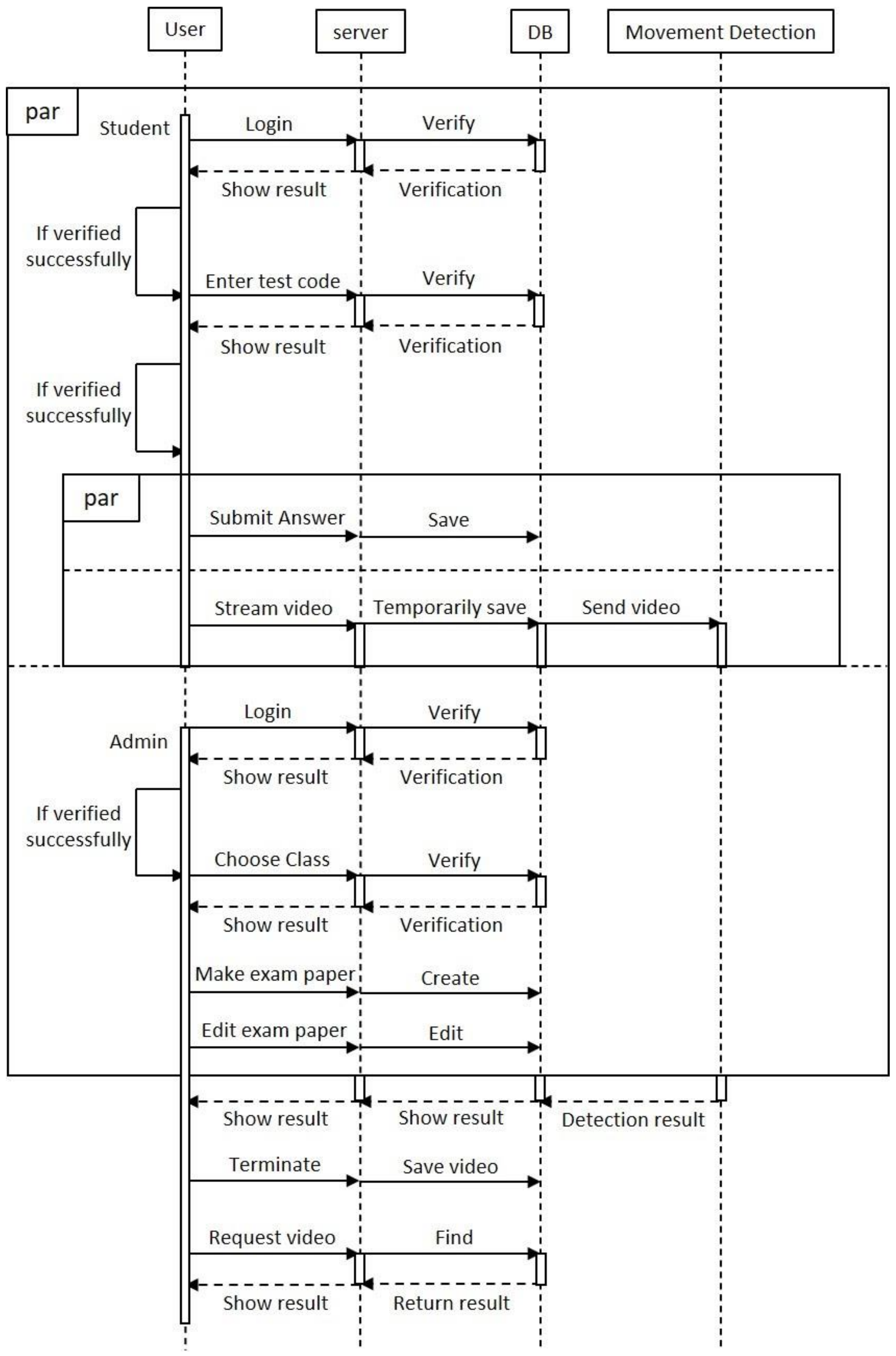
4.2. System Organization

4.2.1. Context Diagram



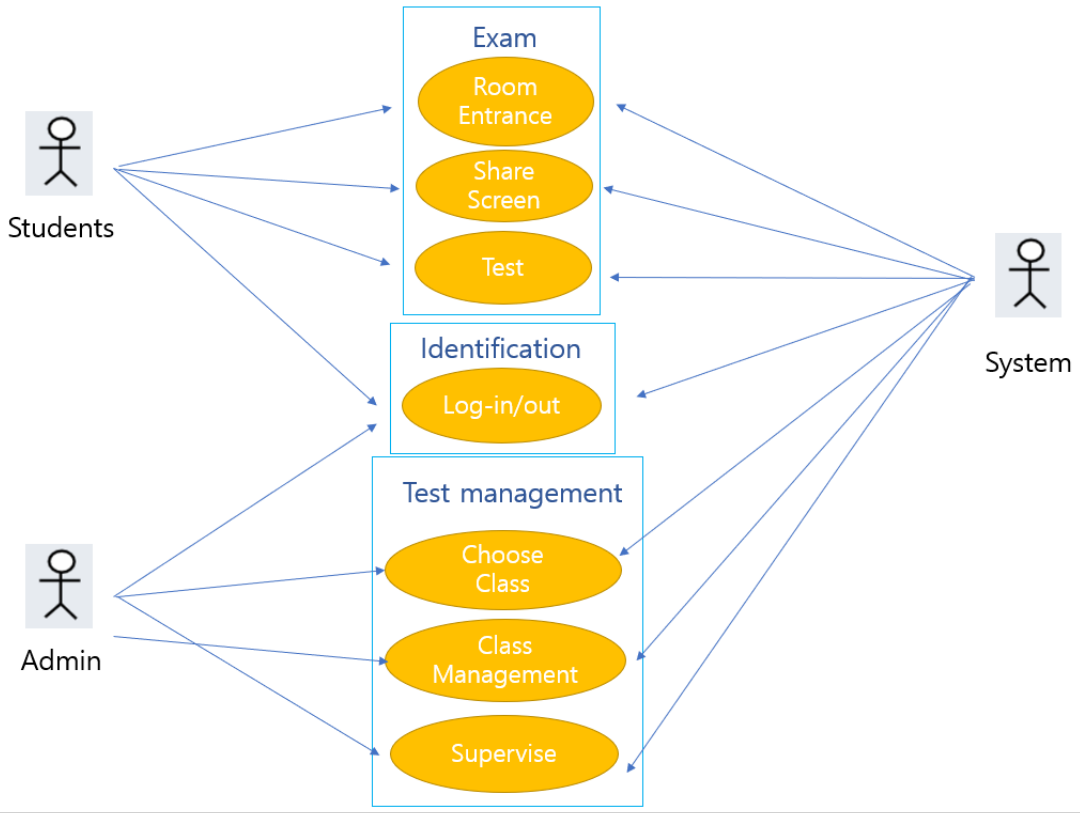
[Figure 1] Overall Context Diagram

4.2.2. Sequence Diagram



[Figure 2] Overall Sequence Diagram

4.2.3. Use Case Diagram



[Figure 3] Use Case Diagram

5. System Architecture - Frontend

5.1. Objectives

This chapter describes the conceptual model that defines the structure, behavior, and more views of the frontend of a system. It consists of system components and the sub-systems developed, that will work together to implement the overall system.

5.2. Subcomponents

5.2.1. User Profile

The user profile class deals with basic user information. All users are already registered. Users can log in/log out. Profile class is one of the constraints to use this service, so a person can be rejected to log in.

5.2.1.1. Attributes

These are the attributes that the user profile class has.

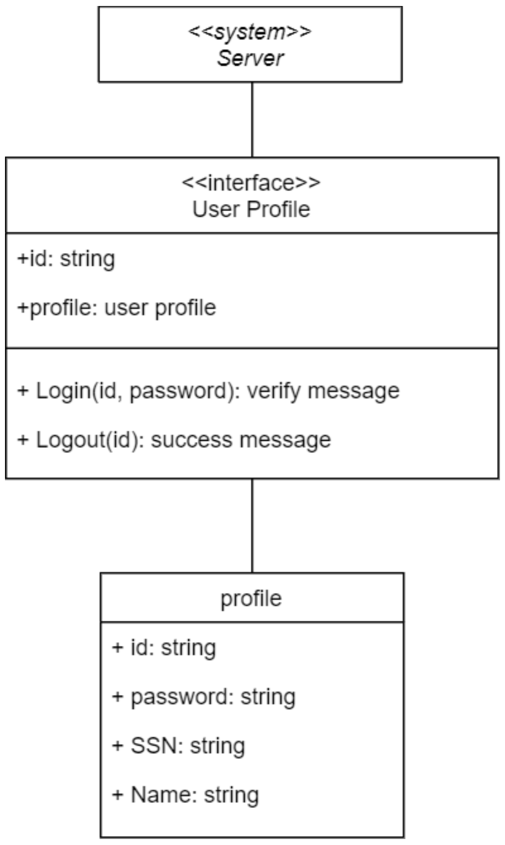
- User id: Id from I-Campus
- User Password: Password from I-Campus
- SSN : student own number
- Name : Name of the user

5.2.1.2. Methods

These are the methods that the user profile class has.

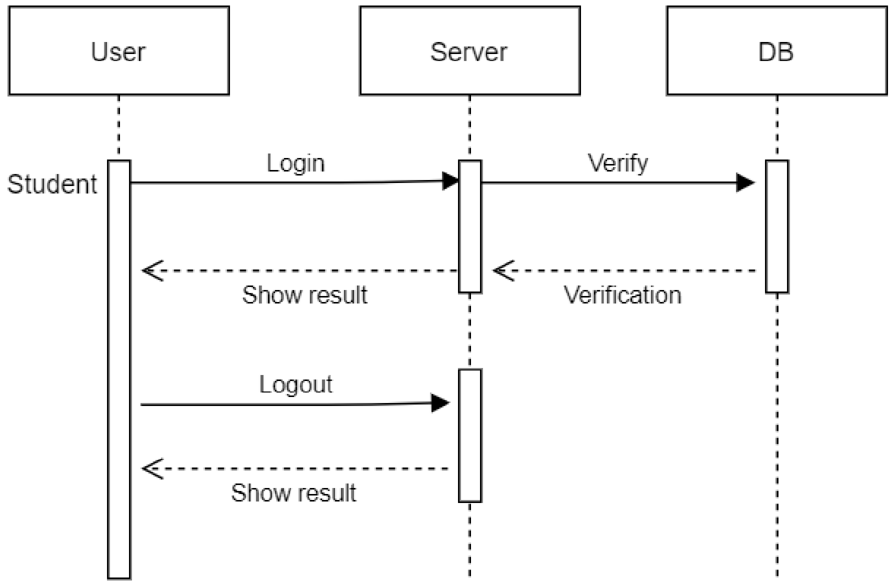
- Login()
- Logout()

5.2.1.3. Class Diagram



[Figure 4] Class Diagram - User Profile

5.2.1.4. Sequence Diagram



[Figure 5] Sequence Diagram - User Profile

5.2.2. Administrator Profile

The administrator profile class deals with basic user information. All users are already registered. Administrators can log in/log out, choose class, manage class and supervise. Profile class is one of the constraints to use this service, so a person can be rejected to log in, choose class, manage class, and supervise if they don't have any authority.

5.2.2.1. Attributes

These are the attributes that the administrator profile class has.

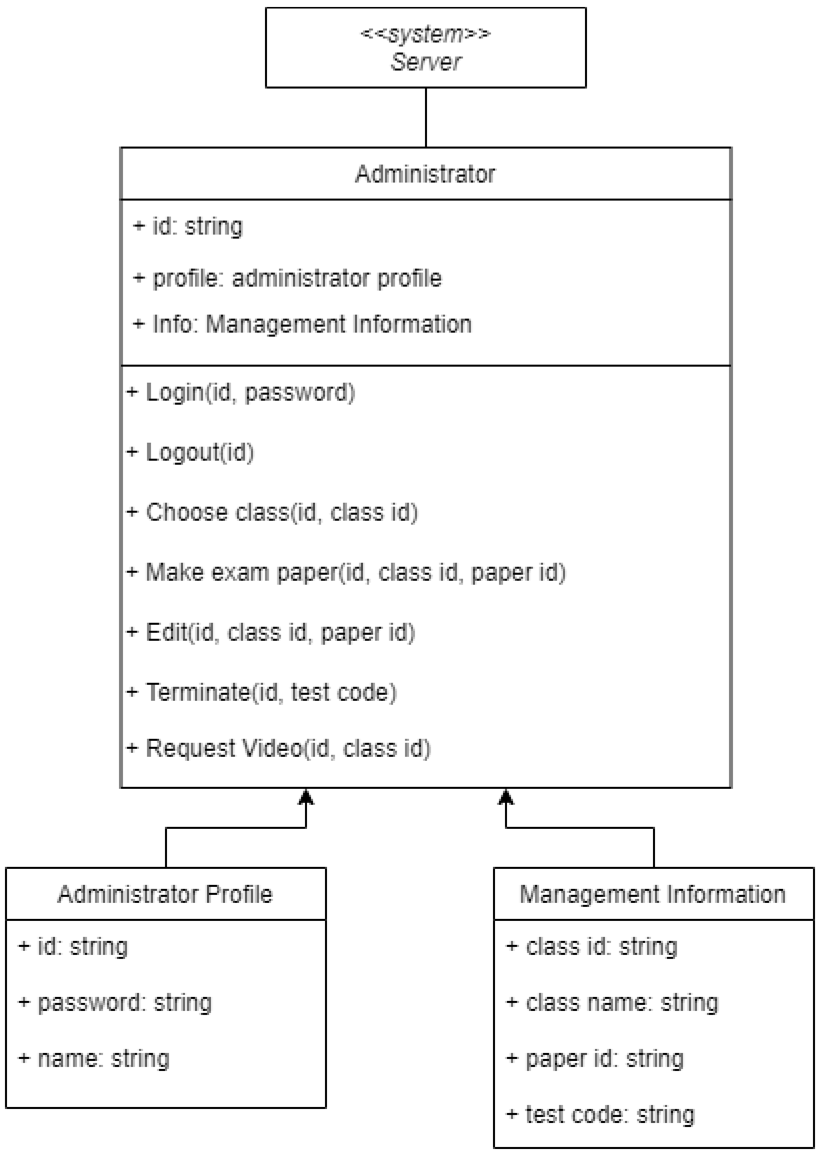
- Administrator Id: Id from I-Campus
- Password: Password from I-Campus
- Name: Administrator name
- Class Id: List of classes id which they have authority
- class name: List of classes id which they have authority
- Paper Id: Id of the exam paper
- Test code: Test code that distinguishes test

5.2.2.2. Methods

These are the methods that the administrator profile class has.

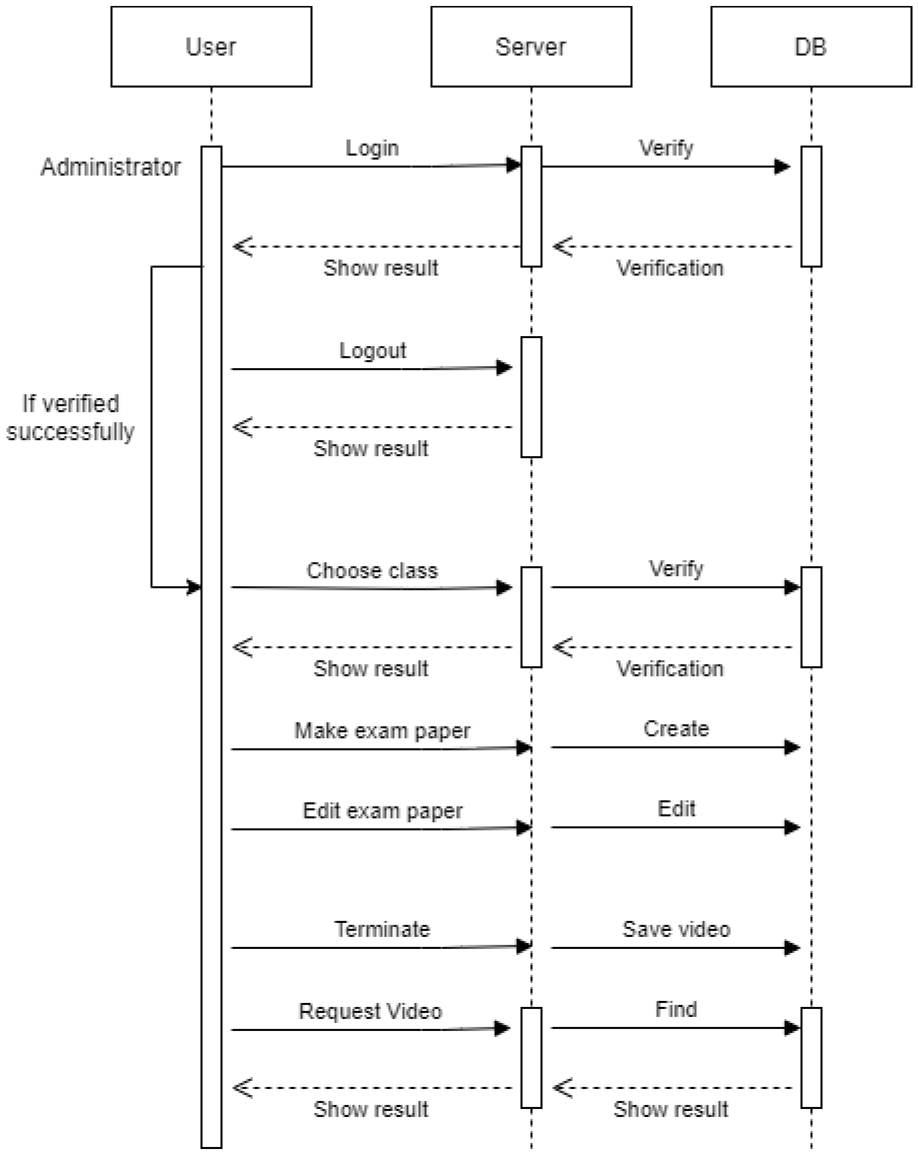
- Login()
- Logout()
- Choose Class()
- Make exam paper()
- Edit exam paper()
- Terminate()
- Request Video()

5.2.2.3. Class Diagram



[Figure 6] Class Diagram - Administrator Profile

5.2.2.4. Sequence Diagram



[Figure 7] Sequence Diagram - Administrator Profile

5.2.3. Share screen

Share screen class get videos of students and show results to administrators. By independence between students which is our service characteristic, results are only shown to administrators.

5.2.3.1. Attributes

These are the attributes that the share screen class has.

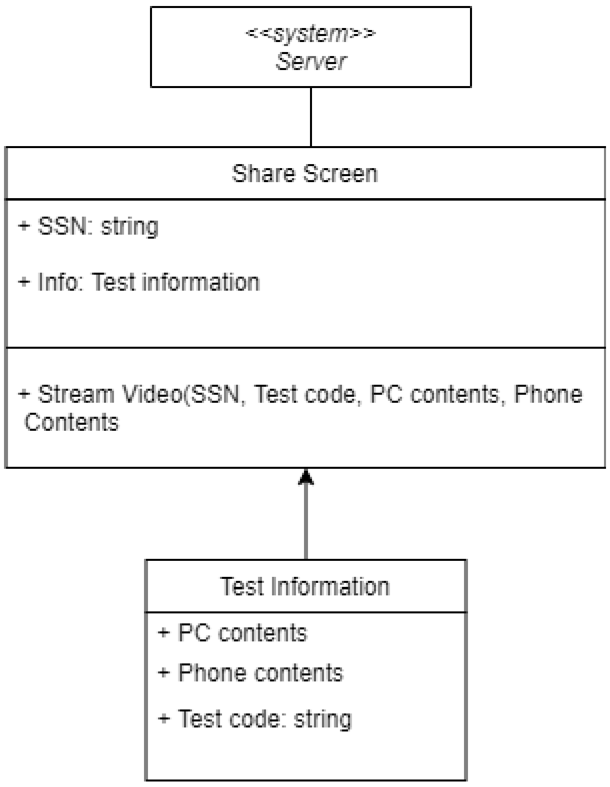
- SSN: Student own number
- Test code: Test code that distinguishes test
- PC Contents : Video from PC screen
- Phone Contents: Video from mobile phone which shows student

5.2.3.2. Methods

This is the method that the share screen class has.

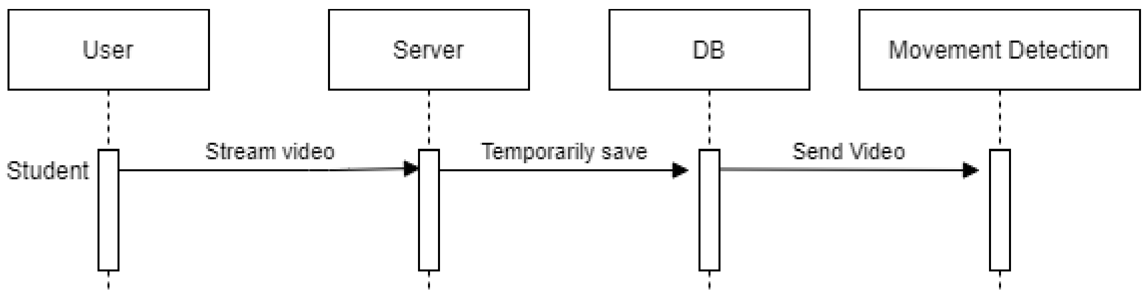
- Stream Video()

5.2.3.3. Class Diagram



[Figure 8] Class Diagram - Share screen

5.2.3.4. Sequence Diagram



[Figure 9] Sequence Diagram - Share screen

5.2.4. Test

Test class deals with students who are taking exams. This class gets the students' information and checks the test code to give authority to take an exam. Students enter the room, take an exam, and submit the answers.

5.2.4.1. Attributes

These are the attributes that the test class has.

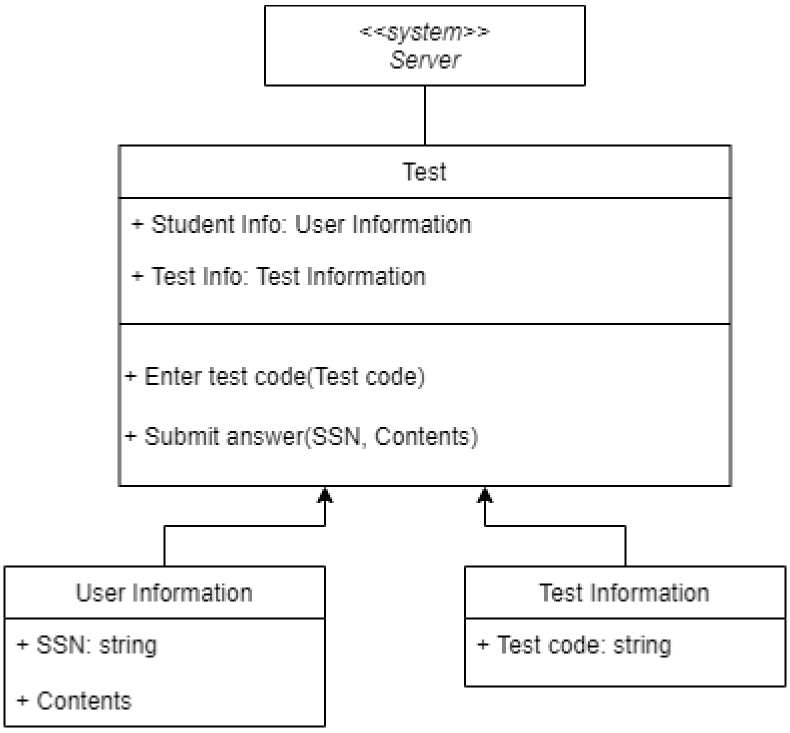
- SSN: Student own number
- Test code: Test code that distinguishes test
- Contents: Answer of exam paper from student

5.2.4.2. Methods

These are the attributes that the test class has.

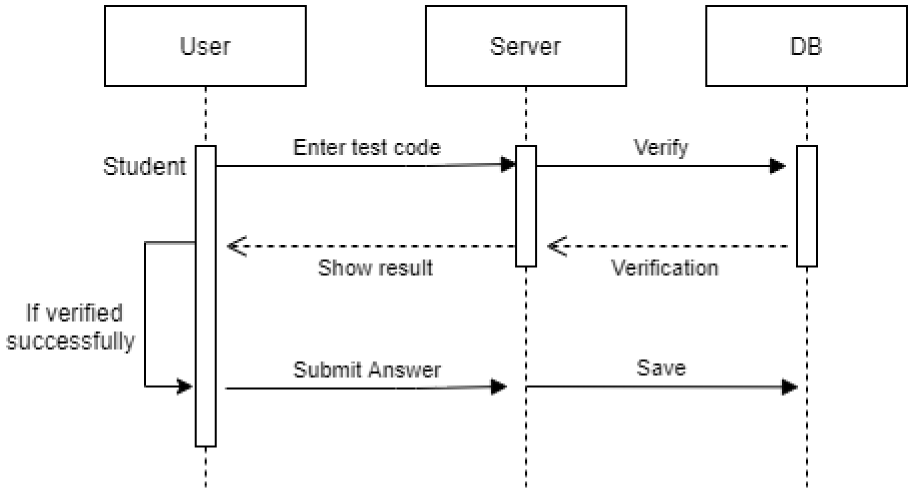
- Enter test code()
- Submit answer()

5.2.4.3. Class Diagram


<<system>>
Server
Test
+ Student Info: User Information
+ Test Info: Test Information
+ Enter test code(Test code)
+ Submit answer(SSN, Contents)
User Information
+ SSN: string
+ Contents
Test Information
+ Test code: string

[Figure 10] Class Diagram - Test

5.2.4.4. Sequence Diagram



[Figure 11] Sequence Diagram - Test

5.2.5. Test management

Test management deals with administrators. Tests cannot be closed until administrators terminate the test. Also, all behaviors of students are monitored by our system for checking whether there is strange behavior like cheating.

5.2.5.1. Attributes

These are the attributes that the test management class has.

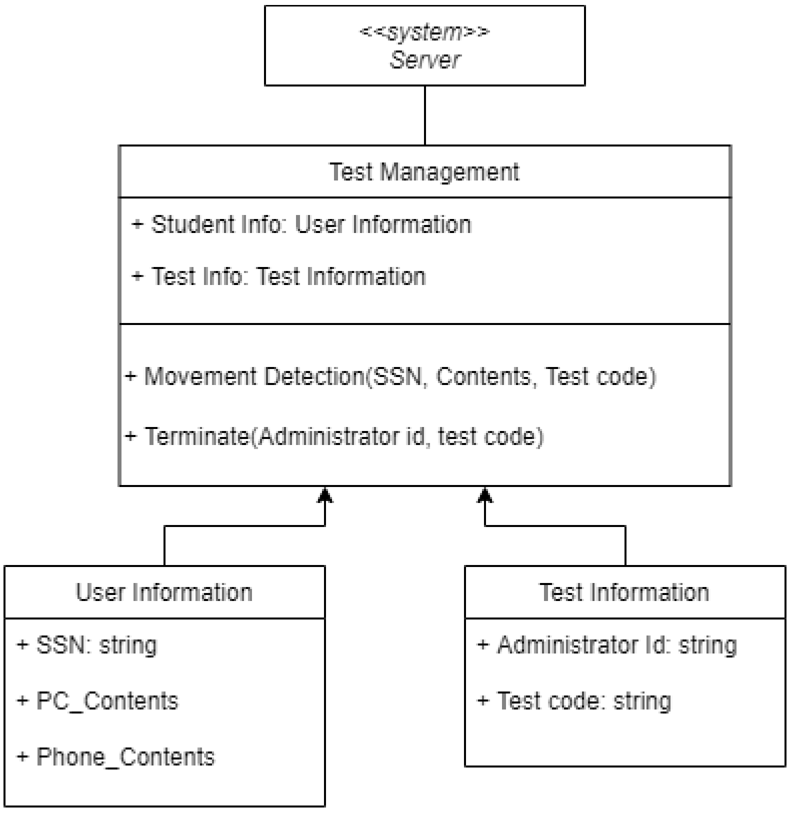
- Administrator Id: Id from I-Campus
- Test code: Test code that distinguishes test
- SSN: Student own number
- PC Contents: Video from PC screen
- Phone Contents: Video from mobile phone which shows student

5.2.5.2. Methods

These are the methods that the test management class has.

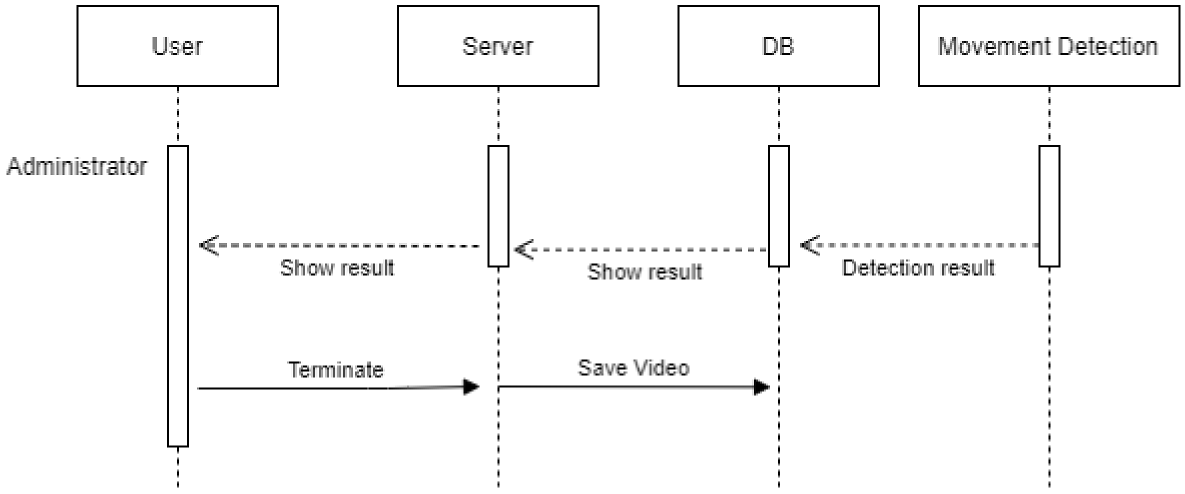
- Movement Detection()
- Terminate()

5.2.5.3. Class Diagram


<<system>>
Server
Test Management
+ Student Info: User Information
+ Test Info: Test Information
+ Movement Detection(SSN, Contents, Test code)
+ Terminate(Administrator id, test code)
User Information
+ SSN: string
+ PC_Contents
+ Phone_Contents
Test Information
+ Administrator Id: string
+ Test code: string

[Figure 12] Class Diagram - Test Management

5.2.5.4. Sequence Diagram



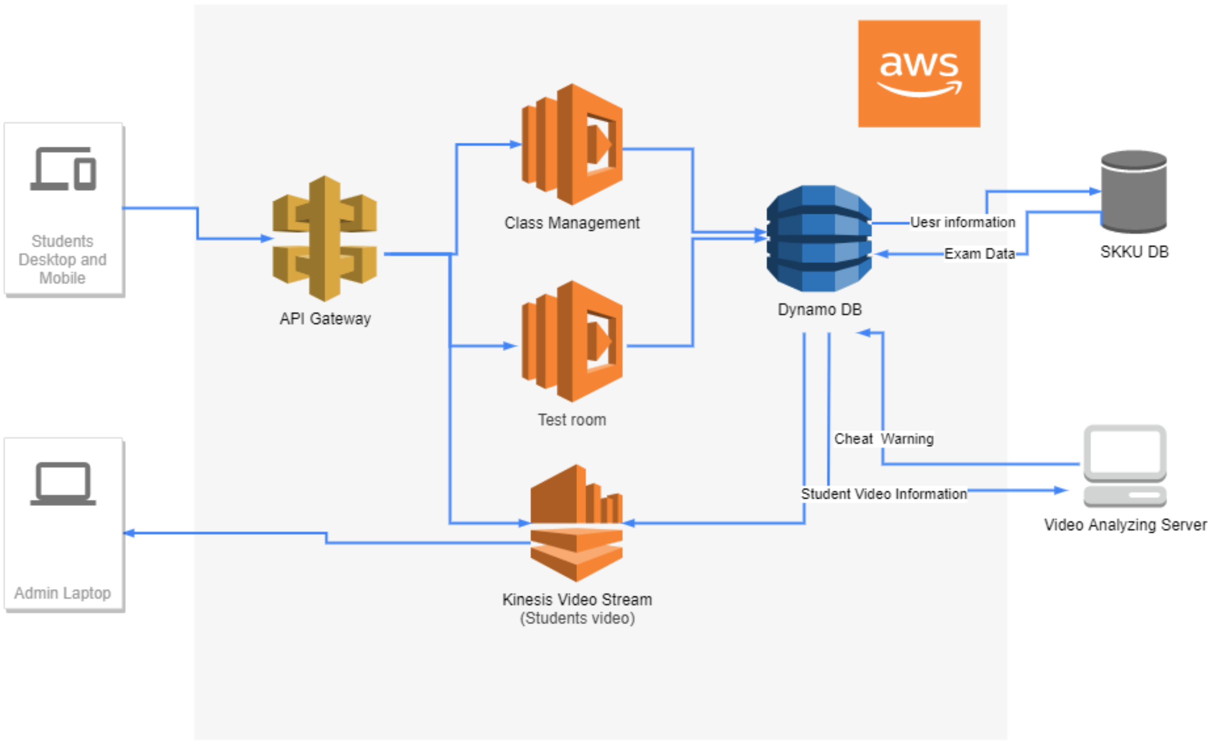
[Figure 13] Sequence Diagram - Test Management

6. System Architecture – Backend

6.1. Objectives

This chapter describes the structure of the back-end system include DB and API Cloud.

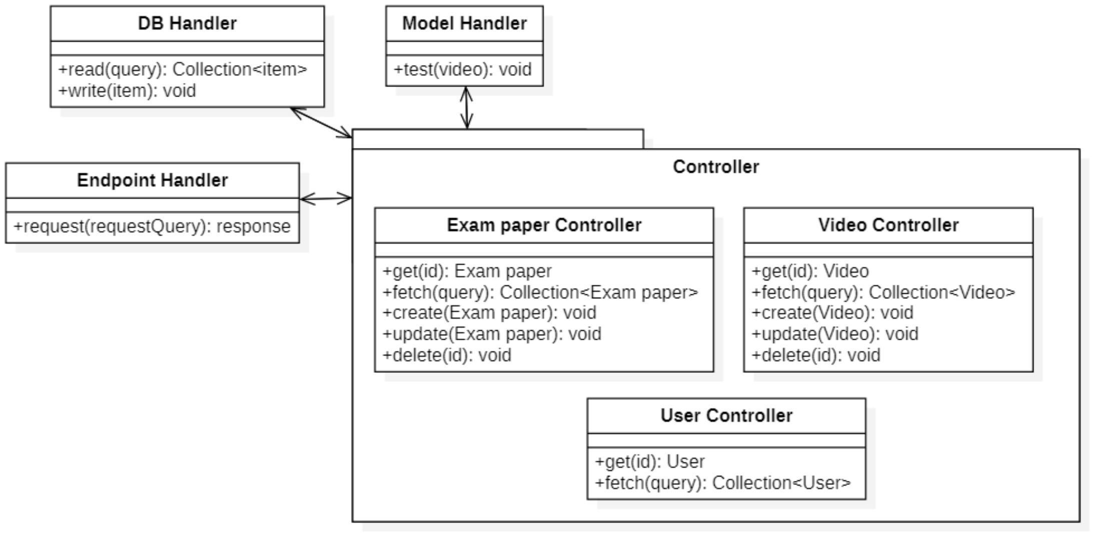
6.2. Overall Architecture



[Figure 14] Overall Architecture

6.3. Subcomponents

6.3.1. Cloud Function



[Figure 15] Class Diagram - Cloud functions

6.3.1.1. Endpoint Handler Class

API gateway. Send requests from the frontend to the appropriate controller or API.

6.3.1.2. DB_Handler Class

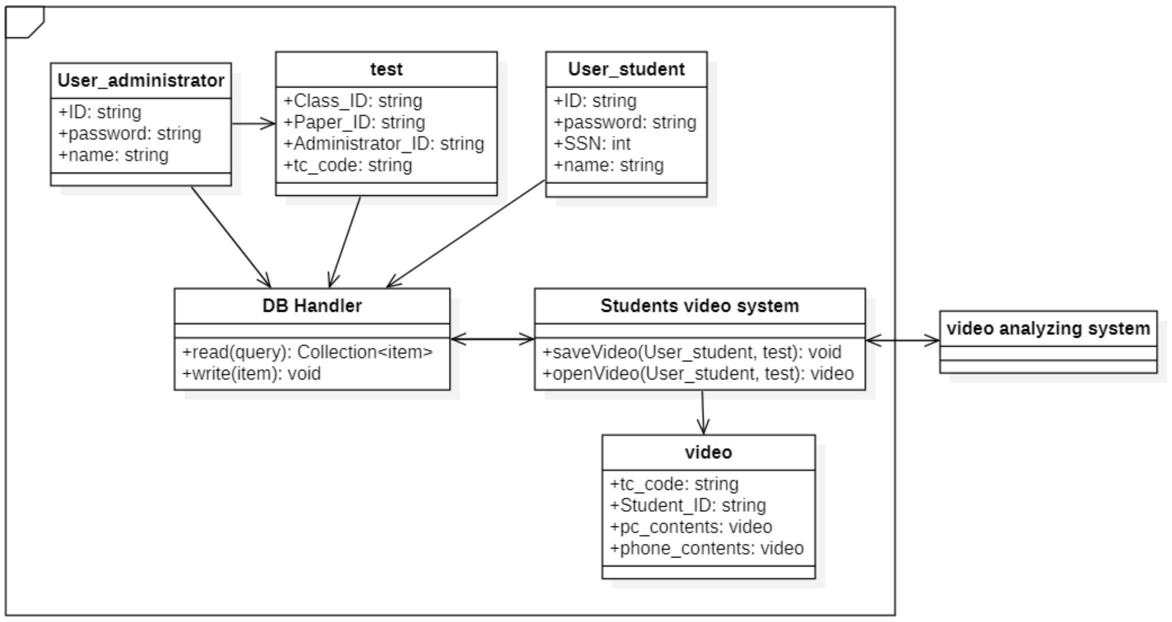
Interface to communicate with DB. Read and write the objects like User, video, exam paper object.

6.3.1.3. Model Handler Class

Interface to communicate with machine learning motion detective model.

6.3.2. Students video system

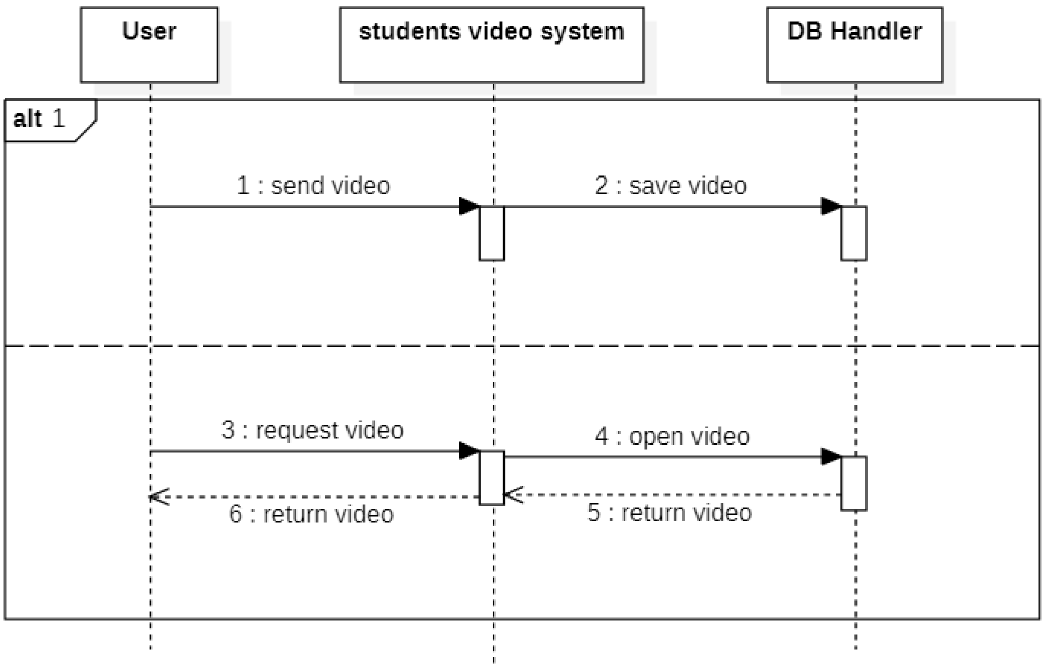
6.3.2.1. Class Diagram



[Figure 16] Class Diagram - Students video system

- Students video system : interface for controlling the video objects. It saves video with its appropriate attributes like tc_code, and student_ID, etc. When an administrator who has the authority request to open the video, it shows the stored video file. Also, This interacts with the video analyzing system.

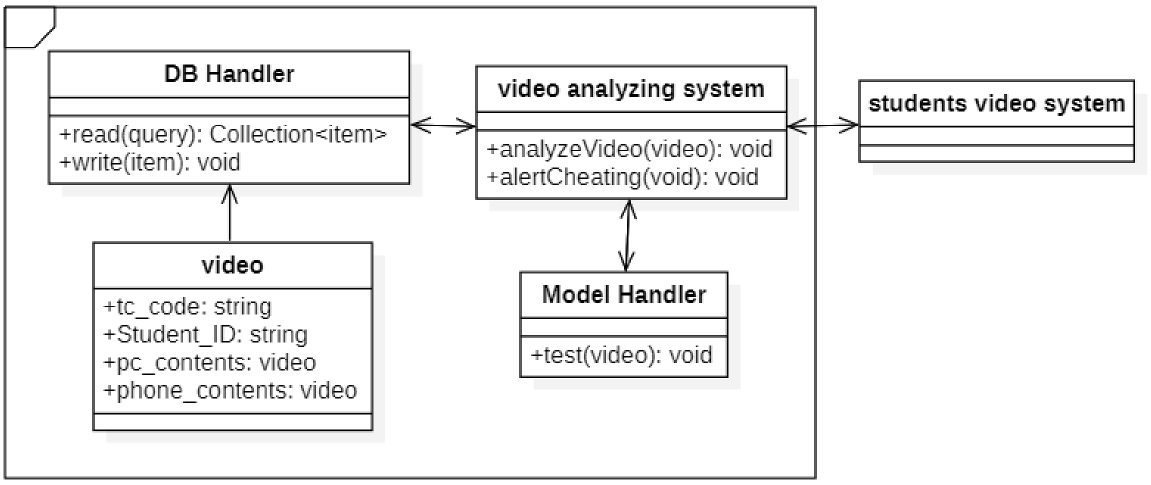
6.3.2.2. Sequence Diagram



[Figure 17] Sequence Diagram - Students video system

6.3.3. Video analyzing system

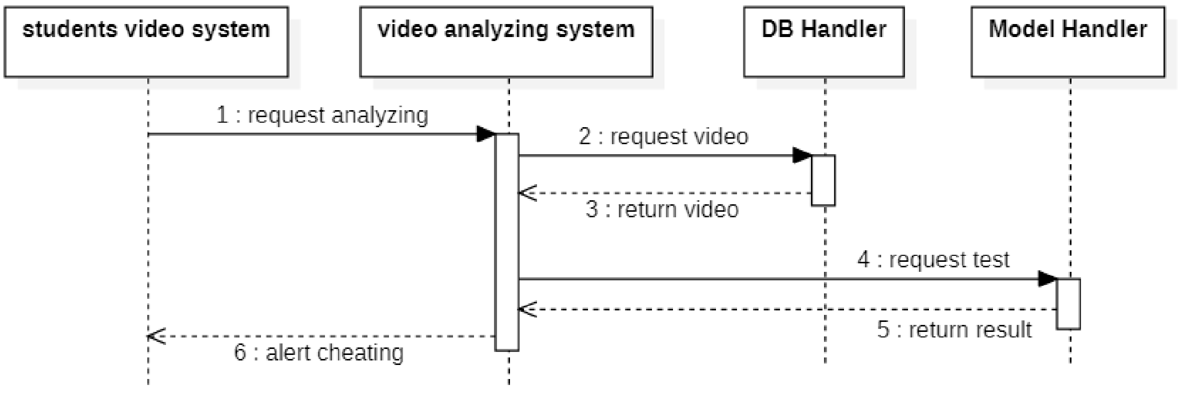
6.3.3.1. Class Diagram



[Figure 18] Class Diagram - Video analyzing system

- Video analyzing system : interface for analyzing the video objects. It analyzes videos and sends alerts to the students video system if there is a suspicious movement.

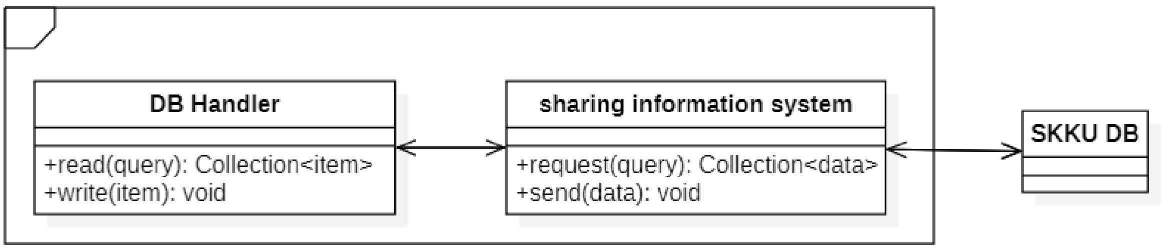
6.3.3.2. Sequence Diagram



[Figure 19] Sequence Diagram - Video analyzing system

6.3.4. Sharing information system

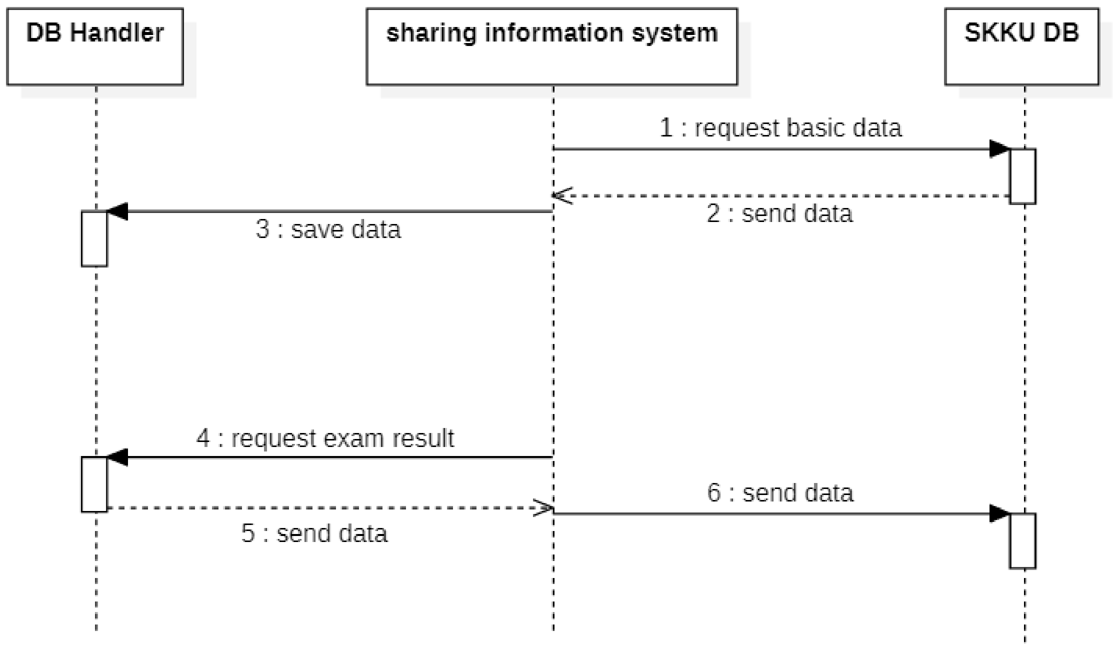
6.3.4.1. Class Diagram


DB Handler
+read(query): Collection<item>
+write(item): void
sharing information system
+request(query): Collection<data>
+send(data): void
SKKU DB

[Figure 20] Class Diagram - Sharing information system

- Sharing information system : interface for data transportation between SKKU DB and our DB. It requests data at the time and sends it to the appropriate DB.

6.3.4.2. Sequence Diagram



[Figure 21] Sequence Diagram - Sharing information system

7. Protocol Design

7.1. Objectives

In this chapter, we cover the structure for the protocol used for the interaction of subsystems such as web pages, servers, DBs, and applications. This chapter also talks about how to define interfaces.

7.2. Audio Video Interleave file(AVI)

A file with the .avi file extension is an Audio Video Interleave file. AVI is a commonly used video format that contains both audio and video. An AVI file uses less compression to store files and takes up more space than many other video formats—like MPEG and MOV. A lossless file will not lose quality over time, regardless of how many times you open or save the file. Additionally, this allows for playback without the use of any codecs.

In our system, transmitting video in real-time and analyzing video would follow this protocol.

7.3. JSON

JSON (JavaScript Object Notation) is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute-value pairs and arrays (or other serializable values). It is a very common data format, with a diverse range of applications, one example being web applications that communicate with a server. The DB type of iCampus is saved in the form of the JSON object. Using a JSON object in the process of importing test papers and student information is very useful in editing, adding, and deleting tasks.

7.4. Student user

7.4.1. Login

- Request

[Table 1] Table of student user login request

Attribute	Detail	
Method	POST	
URL	/user/login	
Parameter	ID	Student Icampus ID
	password	Student Icampus Password

- Response

[Table 2] Table of student user login response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Message	Message for Success
	URL	URL for checking Test code
Failure Response Body	Message	message for wrong ID or PW

7.4.2. Logout

- Request

[Table 3] Table of student user logout request

Attribute	Detail	
Method	GET	
URL	/user/login	
Parameter	ID	Student Icampus ID
	password	Student Icampus Password

- Response

[Table 4] Table of student user logout response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Message	message for successful logout
	URL	URL for homepage
Failure Response Body	Message	message for abnormal logout(exit error)

7.4.3. Checking Test code

- Request

[Table 5] Table of checking test code request

Attribute	Detail	
Method	POST	
URL	/user/checkingTestcode	
Parameter	Testcode	Class own code
	ClassID	Class own ID

- Response

[Table 6] Table of checking test code response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400 (wrong Test code)	
Success Response Body	URL	URL for Testing board
Failure Response Body	Message	message for wrong Test code

7.4.4. Submit Test answer

- Request

[Table 7] Table of submit test answer request

Attribute	Detail	
Method	POST/GET	
URL	/user/submitanswer	
Parameter	Text	Test answer
	ClassID	Class own ID
	StudentID	Student own ID

- Response

[Table 8] Table of submit test answer response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Message	Message for success submission
Failure Response Body	Message	Message for the failure of submission

7.4.5. Recording Test in mobile

- Request

[Table 9] Table of recording test in mobile request

Attribute	Detail	
Method	POST	
URL	/user/recordingmobile	
Parameter	Video	the video that taken using a mobile phone
	Video type	mobile
	StudentID	Student own ID
	Studentname	Student Name

- Response

[Table 10] Table of recording test in mobile response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Message	Success message
Failure Response Body	Message	Fail message

7.4.6. Recording Test in computer

- Request

[Table 11] Table of recording test in computer request

Attribute	Detail	
Method	POST	
URL	/user/recordingcomputer	
Parameter	Video	the video that taken using the webcam
	Video type	Computer
	Student ID	Student own ID
	Studentname	Student Name

- Response

[Table 12] Table of recording test in computer response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Message	Success message
Failure Response Body	Message	Fail message

7.5. Administrator

7.5.1. Login

- Request

[Table 13] Table of administrator login request

Attribute	Detail	
Method	POST	
URL	/admin/login	
Parameter	Administrator ID	Administrator ID
	Administrator PW	Administrator PW

- Response

[Table 14] Table of administrator login response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Message	success message
	URL	URL for select class for testing
Failure Response Body	Message	Fail message

7.5.2. Logout

- Request

[Table 15] Table of administrator logout request

Attribute	Detail	
Method	POST	
URL	/admin/login	
Parameter	Administrator ID	Administrator ID
	Administrator PW	Administrator PW

- Response

[Table 16] Table of administrator logout response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Message	Success message
	URL	URL for select class for testing
Failure Response Body	Message	Fail message

7.5.3. Choose Class

- Request

[Table 17] Table of choose class request

Attribute	Detail	
Method	POST	
URL	/admin/class	
Parameter	classID	Class own ID
	Administrator ID	Administrator own ID

- Response

[Table 18] Table of choose class response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	URL	URL for Upload Exam paper
Failure Response Body	Message	Fail message

7.5.4. Make exam paper

- Request

[Table 19] Table of make exam paper request

Attribute	Detail	
Method	POST	
URL	/admin/exam/create	
Parameter	classID	Class own ID
	Administrator ID	Administrator own ID
	TestpaperID	Test paper own ID

- Response

[Table 20] Table of make exam paper response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	URL	Success message
Failure Response Body	Message	Fail message

7.5.5. Request video

- Request

[Table 21] Table of request video request

Attribute	Detail	
Method	GET	
URL	/admin/monitor	
Parameter	classID	Class own ID
	Administrator ID	Administrator own ID

- Response

[Table 22] Table of request video response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Video of Mobile	the video that taken using a mobile phone
	Video of Computer	the video that taken using a mobile phone
Failure Response Body	Message	Fail message

7.5.6. Reviewing student video

- Request

[Table 23] Table of reviewing student video request

Attribute	Detail	
Method	GET	
URL	/admin/Review	
Parameter	classID	Class own ID
	Administrator ID	Administrator own ID
	TestpaperID	Test paper own ID

- Response

[Table 24] Table of reviewing student video response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Video of Mobile	the video that taken using a mobile phone
	Video of Computer	the video that taken using a mobile phone
Failure Response Body	Message	Fail message

7.5.7. Terminating Exam

- Request

[Table 25] Table of terminating exam request

Attribute	Detail	
Method	POST	
URL	/admin/terminate	
Parameter	classID	Class own ID
	Administrator ID	Administrator own ID
	TestpaperID	Test paper own ID
	time	Deadline for terminating

- Response

[Table 26] Table of terminating exam response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	URL	return to start page
	Message	success message for showing termination and successful submission
Failure Response Body	Message	Fail message

7.6. Test management

7.6.1. Movement detection

- Analyzing student action during exam

[Table 27] Table of movement detection

Attribute	Detail	
Method	GET	
URL	/admin/analyzing	
Parameter	classID	Class own ID
	Administrator ID	Administrator own ID
	TestpaperID	Test paper own ID
	Video	the video that taken using a mobile phone
	Test code	Test own code

- Response

[Table 28] Table of movement detection response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Message	success detection for abnormal behavior
	-	NO abnormal behavior
Failure Response Body	Message	Fail message

7.7. Sharing information system with icampus and SKKU

7.7.1. Get info

- Request

[Table 29] Table of get info request

Attribute	Detail	
Method	GET	
URL	/sharing_info	
Parameter	Query	The query for getting information of skku and icampus

- Response

[Table 30] Table of get info response

Attribute	Detail	
Success Code	200 OK	
Failure Code	HTTP error code = 400	
Success Response Body	Data	student data, administrator class information
	Message	Success message
Failure Response Body	Message	Fail message

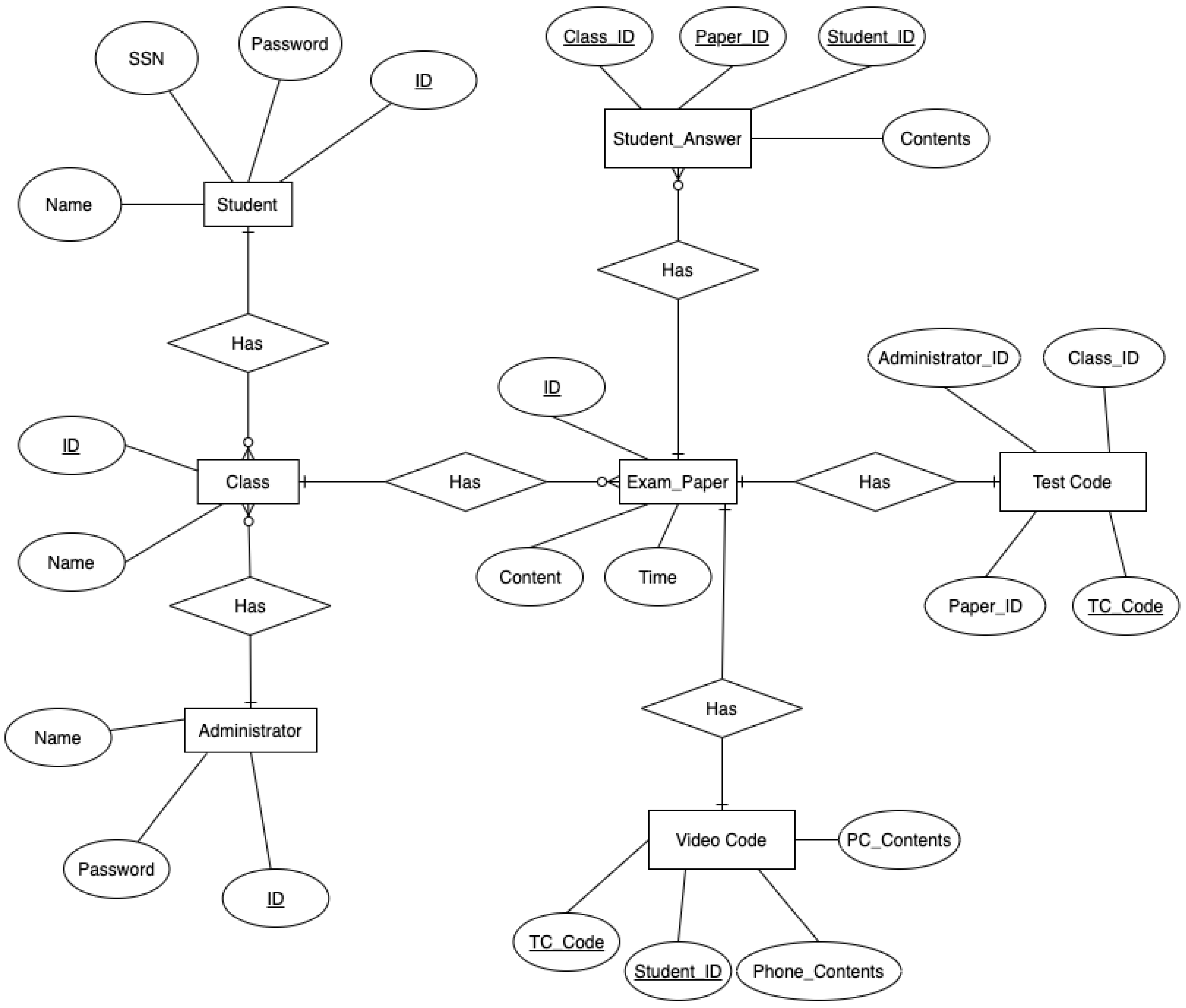
8. Database Design

8.1. Objectives

This section describes the system data structures and how these are to be represented in a database. This part describes entities first, and their relationship through ER-diagram (Entity Relationship diagram). Then, it provides Relational Schema and SQL DDL (Data Description Language) specifications.

8.2. ER Diagram

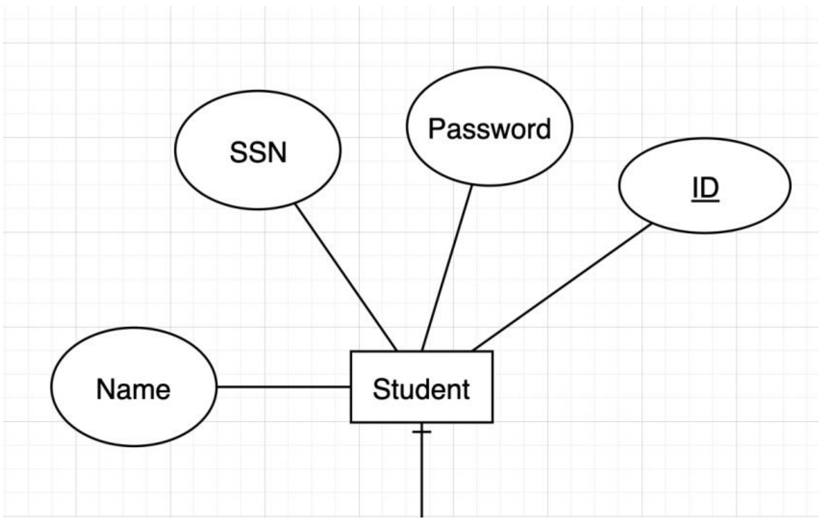
The system consists of seven entities: Student, Class, Administrator, Exam_Paper, Student_Answer, Test Code, and Video Code. ER-diagram expresses each entity as rectangular and their relationship as a rhombus. When an entity has multiple relationships with another entity, three lines are used to indicate it. The attribute of an entity is expressed as an ellipse. The primary key(unique attribute) which uniquely identifies an entity is underlined. There are also entities with multiple primary keys.



[Figure 22] ER-diagram

8.2.1. Entities

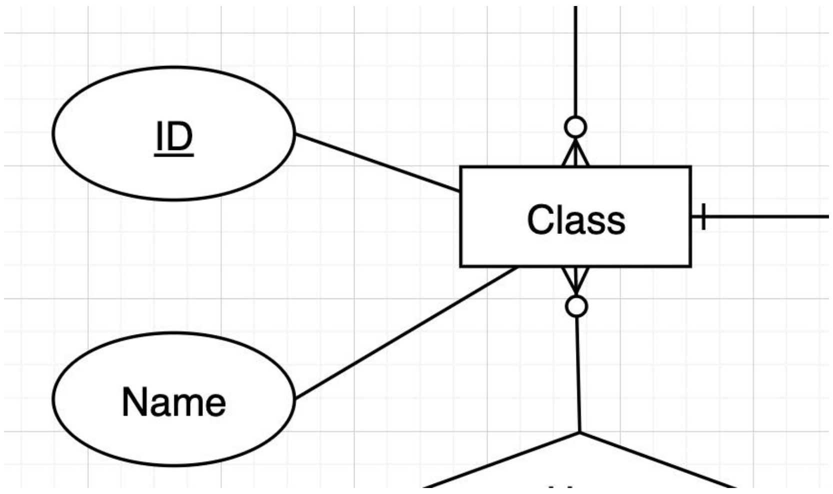
8.2.1.1. Student



[Figure 23] ER diagram, Entity, Student

Student entity represents the user of SKKUEM. It consists of ID, Password, SSN, Name. In these attributes, ID is the primary key. It can have multiple Classes.

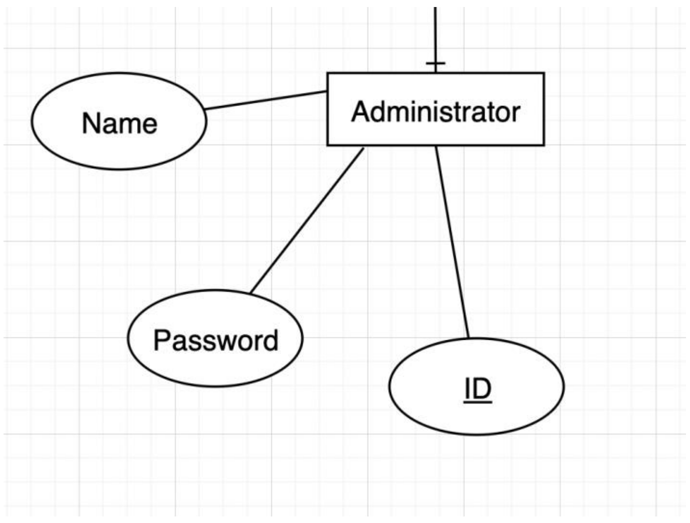
8.2.1.2. Class



[Figure 24] ER diagram, Entity, Class

The Class entity represents the class that the professor hosted that semester. It can have a relationship with a student and administrator. It consists of Class_ID and Class_Name. The primary key is Class_ID. It can have multiple Exam_Paper.

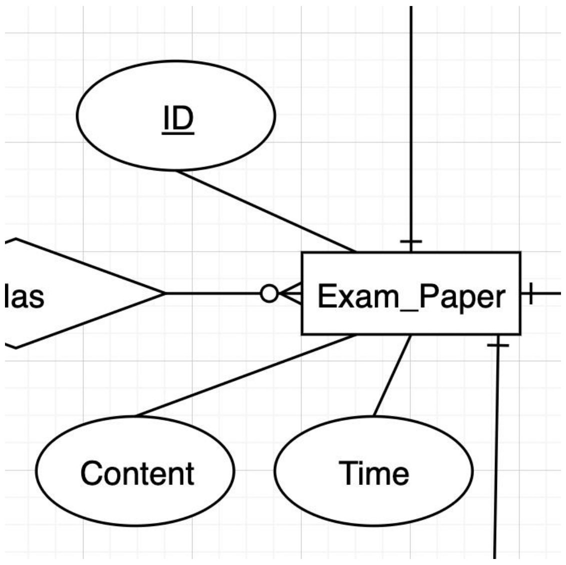
8.2.1.3. Administrator



[Figure 25] ER diagram, Entity, Administrator

The administrator entity represents the host of SKKUEM. The person who becomes an administrator will be a professor. It consists of Name, Password, and ID that is the primary key. The Administrator can have multiple Classes.

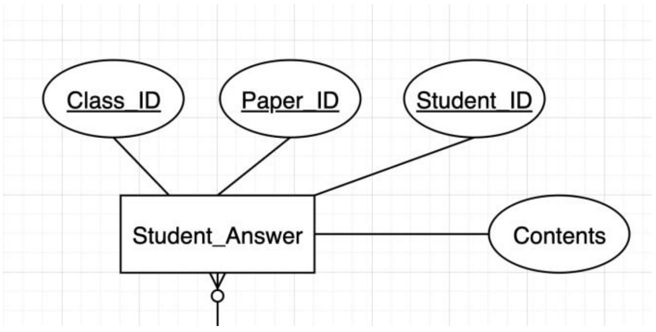
8.2.1.4. Exam_Paper



[Figure 26] ER diagram, Entity, Exam_Paper

Exam_Paper entity represents the exam in Class. This entity has a relationship with Student_Answer, Test Code, and Video Code. It consists of exam_ID, Class_ID(FK), Content, Time.

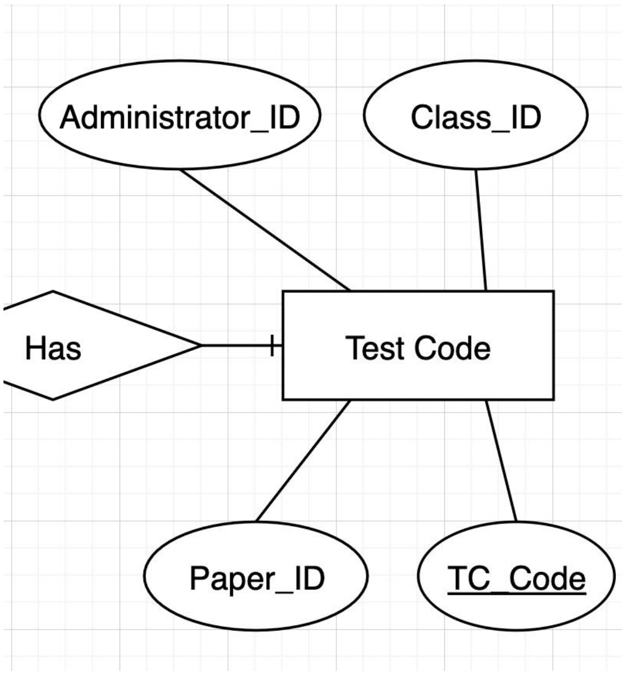
8.2.1.5. Student_Answer



[Figure 27] ER diagram, Entity, Student_Answer

Student_Answer entity represents the answer that students submit for the exam. It consists of Class_ID, Paper_ID, Student_ID and Contents. It has the composite key with Class_ID, Paper_ID, and Student ID

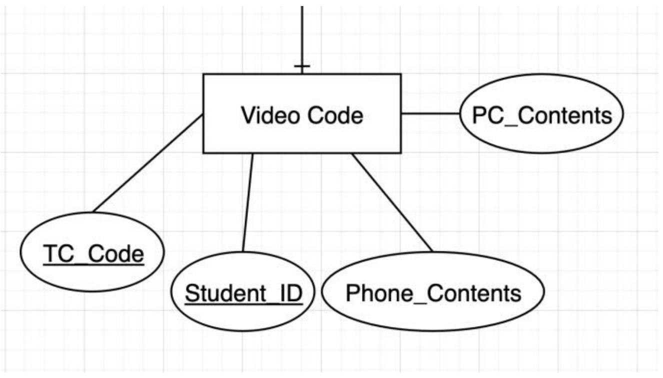
8.2.1.6. Test Code



[Figure 28] ER diagram, Entity, Test Code

Test Code entity represents the code that students should input for seeing test data. It consists of TC_Code that primary key, Class_ID, Paper_ID and Administrator_ID.

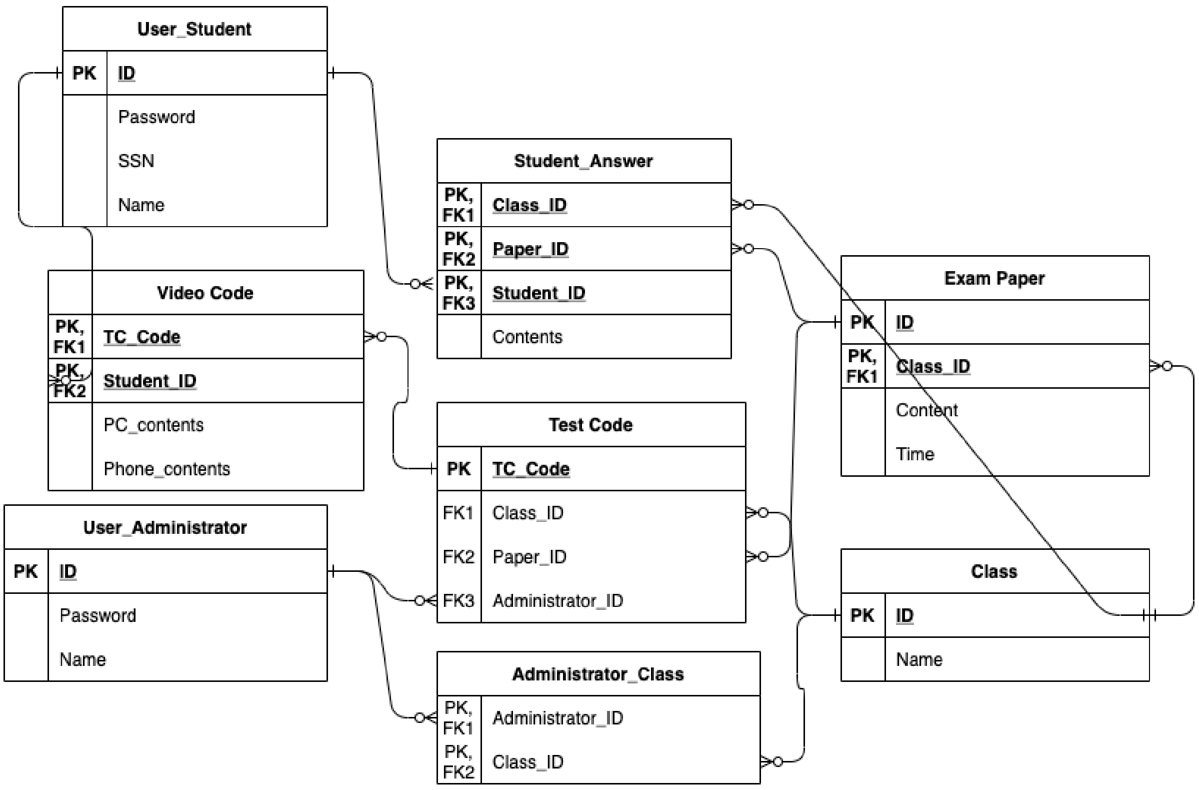
8.2.1.7. Video Code



[Figure 29] ER diagram, Entity, Video Code

Video Code entity represents the video data about recorded student behavior during the test. It has a composite key made up of TC Code and Student ID. It also has Phone_Contents that recorded video from mobile and PC_Contents that recorded video from PC.

8.3. Relational Schema



[Figure 30] Relational Schema

8.4. SQL DDL

8.4.1. Student

```
Create Table Student
(
   Student_ID Varchar(20) NOT NULL,
   Password INT NOT NULL,
   SSN INT NOT NULL,
   Name Varchar(20) NOT NULL,
   PRIMARY KEY (Student_ID)
);
```

8.4.2. Class

```
Create Table Class
(
   Class_ID Varchar(20) NOT NULL
   Class_Name Varchar(30) NOT NULL
   PRIMARY KEY(Class_ID)
);
```

8.4.3. Administrator

```
Create Table Administrator
(
   Administrator_ID Varchar(20) NOT NULL,
   Password INT NOT NULL,
   Administrator_Name Char(20) NOT NULL,
   PRIMARY KEY (Administrator_ID)
);
```

8.4.4. Exam_Paper

```
Create Table Exam_Paper
(
    Paper_ID Varchar(20) NOT NULL,
    Class_ID Varchar(20) NOT NULL,
    Content    Long    NOT NULL,
    Time DATETIME NOT NULL
    PRIMARY KEY (Paper_ID, Class_ID)
    FOREIGN KEY (Class_ID) REFERENCES CLASS(Class_ID)
);
```

8.4.5. Student_Answer

```
Create Table Student_Answer
(
    Paper_ID Varchar(20) NOT NULL,
    Class_ID    Varchar(20)    NOT NULL,
    Student_ID Varchar(20) NOT NULL,
    Content Long NOT NULL,
    PRIMARY KEY (Paper_ID, Class_ID, Student_ID),
    FOREIGN KEY (Class_ID) REFERENCES CLASS (Class_ID),
    FOREIGN KEY (Paper_ID) REFERENCES Exam_Paper (Paper_ID),
    FOREIGN KEY (Student_ID) REFERENCES Student (Student_ID)
);
```

8.4.6. Test_Code

```
Create Table Test_Code
(
    Class_ID Varchar(20) NOT NULL,
    Paper_ID Varchar(20) NOT NULL,
    Administrator_ID Varchar(20) NOT NULL,
    TC_Code Varchar(20) NOT NULL,
    PRIMARY KEY (TC_Code),
    FOREIGN KEY (Class_ID) REFERENCES CLASS (Class_ID),
    FOREIGN KEY (Paper_ID) REFERENCES Exam_Paper (Paper_ID),
    FOREIGN KEY (Administrator_ID) REFERENCES Administrator (Admnistrator_ID)
);
```

8.4.7. Video Code

```
Create Table Video Code
(
    TC_Code Varchar(20) NOT NULL,
    Student_ID Varchar(20) NOT NULL,
    PC_Contents Long NOT NULL,
    Phone_Contents Long NOT NULL,
    PRIMARY KEY (TC_Code, Student_ID)
    FOREIGN KEY (TC_Code) REFERENCES Test_Code (Class_ID)
    FOREIGN KEY (Student_ID) REFERENCES Student (Student_ID)
);
```

9. Testing Plan

9.1. Objectives

This phase will explain plans for development testing, release testing, and user testing. These tests play an important role in increasing program completeness and enabling stable deployment by detecting program errors and supplementing flaws.

9.2. Testing Policy

9.2.1. Development Testing

Development Testing is a test that takes place in the development phase, and the main goal is to stabilize the program. At this stage, the program is stabilized through static code analysis, data flow analysis, peer code review, unit testing, etc. because the program is still incomplete. In this process, we will focus on three main factors. The first is to evaluate how efficiently the program achieves what it does in terms of performance. Secondly, we will evaluate how stable the program works in terms of Reliability. The third is to evaluate Security.

9.2.1.1. Performance

The process of delivering student video information and video analyzing algorithms are quite time-consuming, so we should try to improve this subsystem`s performance in dealing with the process. We would prepare test cases on various gestures and features and evaluate the speed of the video analyzing function and improve the flow of code regarding the video analyzing algorithm and communication with the server.

9.2.1.2. Reliability

Our system is a non-functional requirement where reliability is the most important. This is because a certain malfunction during the test causes a serious situation if the program is terminated or a specific error prevents access to the test room in time. the sub-components and units comprising the system should operate and be connected correctly. therefore we need to unit test first and iteratively integrate to the system checking failure iteratively.

9.2.1.3. Security

In our system, test information, etc. can be the desired target for malicious-intentioned visitors. So we need to be concerned about maintaining the security of this information. There are some free tools to check out web security vulnerabilities like SUCURI, Qualys, etc. These tools would let developers know overlooked vulnerable points in the application. Also, there are some free tools to check out app security vulnerabilities like Ostorlab, Appvigil, etc.

9.2.2. Release Testing

Release testing is the process of testing a particular release of a system that is intended for use outside of the development team. The primary goal of the release testing process is to convince the customer of the system that it is good enough for use. Testing is generally initiated from the first version which a basic implementation of the software is completed and We will look at the responses of users. Users will compare the existing programs (zoom, Webex) and release the second version of them, fixing what we identify as our weaknesses.

9.2.3. User Testing

We should set up possible scenarios and realistic situations that can proceed with necessary user tests. We assume that there would be 15 users for one test room and 60 people can use total. And admin manages at least three classes and the exam paper could be a word file. After setting this situation, we would distribute the Beta version to them and collect user reviews while carrying out our use cases test.

9.2.4. Testing Case

Testing cases would be set according to 3 fundamental aspects of the Reliability Performance, and security. We would set 10 test cases for the Reliability aspect and set 5 test cases for others and proceed with testing on the system and would make an evaluation sheet.

10. Development Plan

10.1. Objectives

This chapter illustrates the development environment and tools we used during development. The limitations and assumptions considered during development will also be described.

10.2. Frontend Environment

10.2.1. Adobe photoshop

[Figure 31] Adobe photoshop logo

Adobe Photoshop is a raster graphics editor developed and published by Adobe Inc. for Windows and macOS. It was used to edit and design visual components such as frames and icons in the program.

10.2.2. Figma

Figma is a vector graphics editor and primarily web-based prototyping tool, with additional offline features enabled by desktop applications for macOS and Windows. The Figma Mirror companion apps for Android and iOS allow viewing Figma prototypes in real-time on mobile devices.

It would be used to design UI/UX like Layout and structure, color and shape, and user experience.

10.2.3. React Native

React Native is an open-source mobile application framework created by Facebook, Inc. It is used to develop applications for Android, Android TV, iOS, macOS, tvOS, Web, Windows, and UWP by enabling developers to use React's framework along with native platform capabilities. It is used for web publishing and ios/android development.

10.3. Backend Environment

10.3.1. WebRTC

WebRTC is a free, open-source project providing web browsers and mobile applications with real-time communication via simple application programming interfaces. Through this API, our program can transmit and check video images in real-time. Almost essential elements in our program can be implemented through basic chatting functions and video call functions.

10.3.2. Node.js

Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. The test manager carries out various analysis programs and real-time video transmission. That is why the server is essential. Through this, the image information and analysis programs are implemented so that they can operate normally on the server.

10.3.3. OpenCV & OpenPose

OpenCV is a library of programming functions mainly aimed at real-time computer vision. OpenPose is the first real-time multi-person system to jointly detect the human body, hand, facial, and foot key-points (in total 135 key-points) on single images. Through this, we will analyze the videos of students taking exams in real-time. These APIs play a very important role in implementing the function of detecting students' cheating or abnormal situations.

10.3.4. AWS

Amazon Web Services is a subsidiary of Amazon providing on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered pay-as-you-go basis. It is used here to collect and analyze data. It is also responsible for server deploying.

10.4. Constraints

The system will be designed and implemented based on the contents mentioned in this document. Other details are designed and implemented by selecting the direction preferred by the developer, but the following items are observed.

- The following are the constraints when developing the system:
  - Use free/libre open-source software as much as possible
  - Avoid the use of proprietary software when possible
  - Keep the system as dependable as possible

- Optimize the system while minimizing premature optimization
- Consider future scalability of the system where the system would be offered in other universities and institutions.
- Test the system as much as possible to reduce any unexpected bugs
- Make the system as secure as possible to prevent any exploits
- Keep the "Don't Repeat Yourself" principle by using the same backend and mostly similar frontend for the system

- The following are the constraints for users:
  - An internet connection is required.
  - Access to a camera is required.
  - The minimum specifications required are an Intel Core 2 Duo CPU 2.XX GHz or an AMD processor, with at least 2GB RAM for the PC and the phone.
  - The operating system required for the PC is any operating system that supports screen-sharing and has a browser that is HTML-5 compliant and supports at least ES6. The operating system required for the phones is any operating system supported by React Native, such as Android, and iOS. The minimum version required for Android is Android 5.0 (Lollipop) while the minimum version required for iOS is iOS 11.0.

10.5. Assumptions and Dependencies

- The assumptions and dependencies for this system are the following:
  - Assumptions
    - It is assumed that the system is allowed to use the database of Sungkyunkwan University to authenticate the user.
    - It is assumed that the exam code would be given to the students by the administrators themselves via the iCampus system before the exam.
  - Dependencies
    - As a web application, the user has to have access to the internet and have a client that supports the application. This client can be the native application for mobile devices and a web browser that is HTML5- and ES6-compliant for other devices.
    - The following are dependencies required for the project:
      - React would be used for developing the website.
      - React Native would be used to create a native application for mobile devices.
      - WebRTC, NodeJS, AWS, OpenCV, and OpenPose would be used for the backend server.

11. Supporting Information

11.1. Software Design Specification

This software design specification was written in accordance with the IEEE Recommendation (IEEE Recommended Practice for Software Design Description, IEEE-Std-1016).

11.2. Document History

[Table 31] Document history

Date	Description	Writer
5/2	define subsystem(4,5)	Hyungjoon Joo
5/2	define and describe development plan (9)	SeungYeon cho
5/6	Define protocol (6)	SeungYeon cho
5/6	Fill database section(7)	JeongMin Lee
5/6	System architecture - Frontend(4)	Eunsu Kang
5/9	System architecture - Frontend(4)	Eunsu Kang
5/9	Write 5. Backend	Kyunghee Ko
5/9	Testing plan (8), Overall Architecture(5.2)	Hyungjoon Joo
5/11	Work on Sections 2.0-2.1	Luke
5/12	Work on Section 1	Luke
5/12	Fix some formatting issues (e.g. Section 7 not appearing on ToC)	Luke
5/13	Add and revise chapter 5.	Kyunghee Ko
5/14	Add and revise section part of chapter 6	SeungYeon cho
5/15	Finish filling out section 2	Luke
5/16	Correct overall typo	Kyunghee Ko