SKKU Exam Manager

Test Plan Document

2021. 05. 30.

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

1. Introduction **5**
1.1. Purpose 5
1.2. Scope 5
1.3. Definitions, Acronyms, and Abbreviation 5
1.4. References 7
1.5. Overview 7
2. Approach **7**
2.1. Test method 7
2.1.1. Software unit test methods 8
2.1.2. Software interface test methods 14
3. Software Unit Test **14**
3.1. Login 14
3.1.1. Login test case 15
3.1.2. Login test case – Negative 15
3.2. Insert Test code 16
3.2.1. Test code case 16
3.2.2. Test code case – Negative 17
3.3. Represent classes 17
3.3.1. Login test case 17
3.3.2. Login test case – Negative 18
3.4. Manage exam paper 18
3.4.1. Manage exam paper test case 18
3.4.2. Manage exam paper test case – Negative 19
3.5. Show the video for the administrator 19
3.5.1. Show the video for the administrator test case 19
3.5.2. Show the video for the administrator test case – Negative 19
3.6. Showing Message when the administrator terminates the exam 20
3.6.1. Showing Message when the administrator terminates the exam test case 20
3.6.2. Showing Message when the administrator terminates the exam test case – Negative 20
3.7. Showing Message when the students submit their own answer 21
3.7.1. Showing Message when the students submit their own answer test case 21
3.7.2. Showing Message when the students submit their own answer test case – Negative 21
3.8. Motion Detection 21
3.8.1. Motion Detection test case 21
3.8.2. Motion Detection test case – Negative 22

3.9. Motion Detection warning message....23
3.9.1. Motion Detection warning message test case....23
3.9.2. Motion Detection warning message test case – Negative....23
3.10. Save exam data....24
3.10.1. Save exam data test case....24
3.10.2. Save exam data test case – Negative....25
4. Software Interface test....25
4.1. MySQL....25
4.1.1. Connection test....25
4.1.2. Connection Nodejs to MySQL....26
4.1.3. Database Validation test....27
4.2. AWS API to DynamoDB....27
4.2.1. S3 upload....27
4.2.2. AWS DynamoDB connect....29
4.3. Video Analyzing....29
4.3.1. Connection test....29
4.3.2. Load test....29
4.4. Sharing information between DB....30
4.4.1. Connection test....30
4.4.2. Load test....30
5. Supporting Information....31
5.1. Document History....31

List of Tables

[Table 1] Table of acronyms and abbreviations 5
[Table 2] Table of terms and definitions 6
[Table 3] Test input of Login 15
[Table 4] Negative Test input of Login 16
[Table 5] Test input of Test code 16
[Table 6] Negative Test input of Test code 17
[Table 7] Test input of Motion Detection 22
[Table 8] Test input of Motion Detection warning message 23
[Table 9] Test case of Exam paper 24
[Table 10] Test case of Exam video 24
[Table 11] Test case of Exam paper 25
[Table 12] Test case of Exam video 25
[Table 13] Document History 31

1. Introduction

1.1. Purpose

This document contains a detailed description of the approach used for testing and the general framework that would be used for this project, which is meant to show that the

requirements for testing this service were determined and planned ahead of time. This document is meant to be read and used by the members of Team 2 for their project; however, other students, professors, and TAs can read this document as well.

1.2. 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.

1.3. Definitions, Acronyms, and Abbreviation

The following table explains the acronyms and abbreviations used in this document.

[Table 1] Table of acronyms and abbreviations

Acronyms & Abbreviations	Explanation
API	Application Programming Interface
DB	Database
DBMS	Database Management System
AWS	Amazon Web Services
S/W	Software
SKKU	Sungkyunkwan University
PC	Personal Computer
TC	Test Code
JS	Javascript

The following table defines certain technical terms used in this document.

[Table 2] Table of terms and definitions

Terms	Explanation
User	Someone who uses a system
System administrator	Someone who hosts class and exams like a professor.
Java	A class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible
Javascript (JS)	A high-level, multi-paradigm programming language that conforms to the ECMAScript specification
Database	An organized collection of data, generally stored and accessed electronically from a computer system.
MySQL	An open-source relational DBMS.
NoSQL	A database that generally stores data in a format that is not a relational table.
DynamoDB	A proprietary NoSQL database that is a part of AWS
Runtime environment	A collection of resources needed to run a program
Node.js	An open-source back-end JS runtime environment
OpenPose	First real-time multi-person system to jointly detect the human body, hand, facial, and foot key-points on single images.
OpenCV	An open-source computer vision and machine learning software library.
Motion Detection	A way to detect the motion of an object relative to its surroundings or a change in the surrounding relative to the object; OpenCV/OpenPose is used for this project
Database Validation	A process that checks the data entered in the database to make sure that the data is consistent, logical and valid
Software Interface	A testing type used in software that verifies whether the two different software systems

Testing	communicate correctly, or not.
Software Unit Testing	A testing type used in software that tests the components of the software to confirm that each unit of the software code works as intended.
Test Case	A collection of actions and conditions used to verify compliance against a specific requirement
JUnit	A unit testing framework for Java

1.4. References

[1] "AWS S3 Nodejs documents," [Online]. Available: https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/s3-node-examples.html.

[2] "JUnit 5 User Guide," [Online]. Available: https://junit.org/junit5/docs/current/user-guide/.

1.5. Overview

The rest of this document is divided into three main sections that describe the tests in detail:

- Approach: how the tests are implemented for this service
- Software Unit Test: how the units of the product function of the service are tested
- Software Interface Test: how the external software interface of this service is tested

2. Approach

2.1. Test method

Make various examples of testing our applications with JUnit. JUnit is a famous framework that is used to test Java. It provides lots of flexibility to test the application from a developer's point of view.

2.1.1. Software unit test methods

2.1.1.1. Login function

Make list of various input(id)

Make list of various input(password)

Make set of various input(id and password)

Insert randomly generated with only character and number of length 5 to id list

Insert randomly generated with only character and number of length 6 to id list

Insert randomly generated with only character and number of length 12 to id list

Insert randomly generated with only character and number of length 13 to id list

Insert randomly generated with character and number and any punctuation of length 6 to id list

Insert randomly generated with character and number and any punctuation of length 12 to id list

Insert randomly generated with character and number and any punctuation of length 7 to password list

Insert randomly generated with character and number and any punctuation of length 8 to password list

Insert randomly generated with character and number of length 9 to password list

Insert randomly generated with character and punctuation of length 9 to password list

Insert randomly generated with number and punctuation of length 9 to password list

Insert randomly generated with character and number of length 10 to password list

Insert randomly generated with character and punctuation of length 10 to password list

Insert randomly generated with number and punctuation of length 10 to password list

For id input in Id list

    For password input in Password list

        Match with id input and password input and insert to set

Repeat

    Get one input from set

    Make class which extends AppCompatActivity with input

    Validate

    Compare with expected output

2.1.1.2. Insert Test code function

Make list of various inputs(Class id)

Make list of various inputs(Paper id)

Make list of various inputs(Administrator id)

Make list of various inputs(Test code)

Insert some of class id from I-Campus to class id list

Insert randomly generated with character and number of length 8 to class id list

Insert randomly generated with character and number of length 9 to class id list

Insert randomly generated with character and number of length 10 to class id list

Insert randomly generated with character of length 2 to paper id list

Insert randomly generated with character of length 3 to paper id list

Insert randomly generated with character and number of length 2 to paper id list

Insert randomly generated with only character and number of length 5 to administrator id list

Insert randomly generated with only character and number of length 6 to administrator id list

Insert randomly generated with only character and number of length 12 to administrator id list

Insert randomly generated with only character and number of length 13 to administrator id list

Insert randomly generated with character and number and any punctuation of length 6 to administrator id list

Insert randomly generated with character and number and any punctuation of length 12 to administrator id list

For class id input in Class Id list

    For paper id input in Paper Id list

        For Administrator id input in Administrator Id list

            Combine all inputs and insert to test code list

Repeat

    Get one input from test code list

    Make class which extends AppCompatActivity with input

    Validate

    Compare with expected output

2.1.1.3. Represent all the administrator classes

Get classes from SKKU DB and make a list

Get the length of list

Get classes from our application and make a list

Get the length of list

If lengths of two lists are equal

jump to 2nd condition

else

validation failed

2nd Condition:

for name in SKKU DB classes list

if name is not in application list

validation failed

validation succeeded

2.1.1.4. Manage exam paper

Get the number of exam paper and store to “p” variable

Upload any empty exam paper

Get the number of exam paper and store to “c” variable

if p + 1 == c

validation succeeded

else

validation failed

Get the number of exam paper and store to “p” variable

Delete any exam paper

Get the number of exam paper and store to “c” variable

if p - 1 == c

validation succeeded

else

validation failed

2.1.1.5. show the video for administrator

Retrieve the number of images taken on mobile and store them in nm

Get the number of images taken on PC and save to np

Get total number of students and save to sum

```
if( nm > sum)
        validation failed
if( np > sum)
        validation failed
if(np != nm)
        validation failed
```

Creating a video list of length "sum" named “imsi”.

```
while(1){
if(terminating command)
        break;
        for id in SKKU DB classes list
                (show the video in each id)

}
```

2.1.1.6. Showing Message when Administrator terminates the exam

```
while(!(sig = Administrator presses the end button));
//(Infinite loop continues until administrator presses quit)
if(sig == 1)
        goto check_point1
check_point1:
        for id in SKKU DB classes list
                Go to the student's test homepage
                Forced storage of test information
                kill the process;


}for id in SKKU DB classes list
        if( the process of the test homepage for student[id])
                goto check point1;
```

2.1.1.7. Showing Message when Students submit their own answer

```
Import file information from students
Get the file size and save it in "size"
if(size == 0)
        validation failed

Receiving file information to DB

Requesting the file information from the DB again and saving the file again in "imsi"

if("imsi" == student answer)
        show the message
        validation succeed
else
      validation failed.
```

2.1.1.8. Motion Detection

```
while(1){
if(terminate command )
        exit()
else{
maction = mobile realtime video
paction = pc realtime video

if(motion_dection(maction) → abnormal)
        Motion_detection_warning(maction);
if(motion_detection(paction) → abnormal)
        Motion_detection_warning(paction);
}
}
```

2.1.1.9. Motion Detection warning

```
Motion_detection_warning(action){

while(! isempty(action)) ;

Red error message is formed on the administrator screen
Importing action video on the network
for name in SKKU DB
        print(name and action)
}
```

2.1.2. Software interface test methods

2.1.2.1. MySQL

Input code refers to section 4.1.2.

2.1.2.2. AWS API to DynamoDB

Input code refers to section 4.2.1. and 4.2.2.

2.1.2.3. Video Analyzing

```
while(ping_status == connected){
        motion_detection()

        if (terminate_command) exit()
}
```

2.1.2.4. Sharing information between DB

```
while(ping_status == connected){
        send data query
        if (error) report and recover the network

        if (terminate_command) exit()
}
```

3. Software Unit Test

Unit testing is a type of software testing where individual units or components of the software are tested. The purpose is to validate that each unit of the software code performs as expected. Unit Testing is done during the development (coding phase) of an application by the developers. Unit Tests isolate a section of code and verify its correctness. A unit may be an individual function, method, procedure, module, or object.

3.1. Login

Our login system equals with I-Campus which is the online campus of our university, Sungkyunkwan University. The data of students, administrator information is from the SKKU database.

3.1.1. Login test case

3.1.1.1. Test case object

To check correct login function when positive input inserted.

3.1.1.2. Test inputs

Total: 2 x 4 = 8

[Table 3] Test input of Login

	Id	Password
1	3skku3	jEs#2a38
2	gheuK382fhje	jr29fhjs39
3		938@3^$$21
4		fs#%@BdS()

3.1.1.3. Expected Results

Login success message

3.1.2. Login test case – Negative

3.1.2.1. Test case object

To check correct login function when negative input inserted.

3.1.2.2. Test Inputs

Total: 4 x 5 = 20

[Table 4] Negative Test input of Login

	Id	Password
1	sKe98	ghJ#293
2	hj!93d	672@39$$!
3	@#fe382@4s3$	fejf39Ae9
4	cxCVb932hfe8f	fjs#@fad!
5		672@39$$!

3.1.2.3. Expected Results

Login failed message

3.2. Insert Test code

3.2.1. Test code case

3.2.1.1. Test case object

To check correct insert test code function when positive input inserted.

3.2.1.2. Test inputs

Total: 3 x 1 x 2 = 6

[Table 5] Test input of Test code

	Class id	Paper id	Administrator id
1	BIZ202101	se	3skku3
2	CSE303641		gheuK382fhje
3	EEE200843		

3.2.1.3. Expected Results

Success message

3.2.2. Test code case – Negative

3.2.2.1. Test case object

To check correct insert test code function when negative input inserted.

3.2.2.2. Test inputs

Total: 3 x 2 x 4 = 24

[Table 6] Negative Test input of Test code

	Class id	Paper id	Administrator id
1	S93SEIF1	fko	sKe98
2	DKGE0293B	s3	cxCVb932hfe8f
3	38REJF03HF		hj!93d
4			@#fe382@4s3$

3.2.2.3. Expected Results

Failed message

3.3. Represent classes

3.3.1. Login test case

3.3.1.1. Test case object

To check whether all classes are represented correctly.

3.3.1.2. Test inputs

Make SKKU DB for classes empty and represented classes empty.

Make SKKU DB for classes of 1 and represented classes of 1.

Make SKKU DB for classes of 30 and represented classes of 30.

3.3.1.3. Expected Results

1st condition and 2nd condition are succeeded and assert success message.

3.3.2. Login test case – Negative

3.3.2.1. Test case object

To check system catches the error when representing classes function has defected.

3.3.2.2. Test inputs

1)

Make SKKU DB for classes empty and represented classes.

Make SKKU DB for classes and represented classes empty.

Make SKKU DB for classes of 1 and represented classes of 2.

Make SKKU DB for classes of 31 and represented classes of 30.

2)

Make SKKU DB for classes of 2 and represented classes of 2 and each name is different.

3.3.2.3. Expected Results

1) 1st condition has to be failed.

2) 1st condition has to be succeeded and 2nd condition has to be failed.

3.4. Manage exam paper

3.4.1. Manage exam paper test case

3.4.1.1. Test case object

To check whether uploading and deleting exam papers correctly.

3.4.1.2. Test inputs

Upload empty exam paper

Upload pre-written exam paper

Delete one of the exam papers

3.4.1.3. Expected Results

After upload and delete, the number of exam papers must be different by one.

3.4.2. Manage exam paper test case – Negative

3.4.2.1. Test case object

To check whether the action is rejected when deleting exam papers where there is empty.

3.4.2.2. Test inputs

Delete exam paper where there is empty.

3.4.2.3. Expected Results

Error message and the action should be rejected.

3.5. Show the video for the administrator

3.5.1. Show the video for the administrator test case

3.5.1.1. Test case object

To verify that students' videos are visible to the administrator

3.5.1.2. Test inputs

Upload empty Video

Upload Video of Mobile

Upload Video of PC

3.5.1.3. Expected Results

After Upload the videos, the videos from PC and Mobile are displayed concurrently.

3.5.2. Show the video for the administrator test case – Negative

3.5.2.1. Test case object

To check whether the message is shown when Nothing is upload.

3.5.2.2. Test inputs

None

3.5.2.3. Expected Results

The message will announce the video slot is empty.

3.6. Showing Message when the administrator terminates the exam

3.6.1. Showing Message when the administrator terminates the exam test case

3.6.1.1. Test case object

To verify that the system shows the message when the administrator terminates the exam.

3.6.1.2. Test inputs

terminate command

3.6.1.3. Expected Results

The message of successful terminating will be shown.

3.6.2. Showing Message when the administrator terminates the exam test case – Negative

3.6.2.1. Test case object

To check whether the message is not shown.

3.6.2.2. Test inputs

Same Command with Positive case

3.6.2.3. Expected Results

The message will be shown what is the error. (Network, Disk ……)

3.7. Showing Message when the students submit their own answer

3.7.1. Showing Message when the students submit their own answer test case

3.7.1.1. Test case object

To verify that the system shows the message when Students submit their own answer.

3.7.1.2. Test inputs

Upload empty answer text.

Upload pre-written answer text.

Reload answer text

3.7.1.3. Expected Results

The message of successfully submitting the answer.

3.7.2. Showing Message when the students submit their own answer test case – Negative

3.7.2.1. Test case object

To check whether the submitting error has existed or the Message is not shown.

3.7.2.2. Test inputs

(a) Upload Nothing

(b) Upload the Same answer text when an external problem exists(Network…..)

3.7.2.3. Expected Results

(a) A message will inform you that there are no submissions.

(b) A message will inform you what the external problem is.

3.8. Motion Detection

3.8.1. Motion Detection test case

3.8.1.1. Test case object

To verify that the system can detect cheating actions.

3.8.1.2. Test inputs

[Table 7] Test input of Motion Detection

	Action Video
1	Captured by more than one person on the screen
2	Wearing earphones
3	The act of moving your hand off the screen
4	Frequently changing the angle of the body
5	The act of shedding snow
6	The act of keeping an eye on a specific direction
7	Going out of the screen (when no one is on the screen)
8	The act of taking seat off
10	Touching an electromechanical during the test

3.8.1.3. Expected Results

The abnormal action would be captured with a red border over the student's video.

3.8.2. Motion Detection test case – Negative

3.8.2.1. Test case object

To ensure that the test taker without cheating during the test is not erroneously warned

3.8.2.2. Test inputs

the typical test video without cheating

3.8.2.3. Expected Results

show the video without any additional information.

3.9. Motion Detection warning message

3.9.1. Motion Detection warning message test case

3.9.1.1. Test case object

Check whether messages about cheating are well displayed

3.9.1.2. Test inputs

(same with 3.8)

[Table 8] Test input of Motion Detection warning message

	Action Video
1	Captured by more than one person on the screen
2	Wearing earphones
3	The act of moving your hand off the screen
4	Frequently changing the angle of the body
5	The act of shedding snow
6	The act of keeping an eye on a specific direction
7	Going out of the screen (when no one is on the screen)
8	The act of taking seat off
10	Touching an electromechanical during the test

3.9.1.3. Expected Results

A red warning message appears on the screen of the abnormal behavior

3.9.2. Motion Detection warning message test case – Negative

3.9.2.1. Test case object

Does not properly generate warnings about cheating

3.9.2.2. Test inputs

the typical test video without cheating

3.9.2.3. Expected Results

No warning messages are displayed during the test.

3.10. Save exam data

3.10.1. Save exam data test case

3.10.1.1. Test case object

To check push Exam Paper and Video Code info to MySQL database when enrolling in a new Exam Paper and Video Code.

3.10.1.2. Test inputs

[Table 9] Test case of Exam paper

	Class_ID	Paper_ID	Student_ID	Contents
1	SW3295	SW3295_05	abc	Image format
2	BA6904	BA6904_11	aaaa	Image format
3	SW0012	SW00012_01	asdf	Image format

[Table 10] Test case of Exam video

	TC_Code	Student_ID	PC_Contents	Phone_Contents
1	5902	a	Video data type	Video data type
2	1211	aaaa	Video data type	Video data type
3	A094	fdsaf42332	Video data type	Video data type

3.10.1.3. Expected Results

All test case should show flawless operation and contents should be saved in AWS DynamoDB

3.10.2. Save exam data test case – Negative

3.10.2.1. Test case object

To check program can handle invalid code or contents

3.10.2.2. Test inputs

[Table 11] Test case of Exam paper

	Class_ID	Paper_ID	Student_ID	Contents
1	abcde	SW3295_05	abc	Image format
2	BA6904	1234	aaaa	Image format
3	SW0012	SW00012_01	asdf	Video data type

[Table 12] Test case of Exam video

	TC_Code	Student_ID	PC_Contents	Phone_Contents
1	5902	a	Image format	Video data type
2	1211	aaaa	Video data type	Image format
3	@ACEW$	fdsaf42332	Video data type	Video data type

3.10.2.3. Expected Results

program denied the save test files and program operate properly

4. Software Interface test

4.1. MySQL

4.1.1. Connection test

4.1.1.1. Test case object

To check external software interfaces of MySQL.

4.1.1.2. Test inputs

Ping command

4.1.1.3. Expected Results

Ping status

4.1.2. Connection Nodejs to MySQL

4.1.2.1. Test case object

To check connection between Nodejs framework and MySQL.

4.1.2.2. Test input code

```
var mysql = require('mysql');

module.exports = function () {
  return {
    init: function () {
      return mysql.createConnection({
        host: 'localhost',
        port: '3306',
        user: 'username',
        password: 'password',
        database: 'test'
      })
    },

    test_open: function (con) {
      con.connect(function (err) {
        if (err) {
          console.error('mysql connection error :' + err);
        } else {
          console.info('mysql is connected successfully.');
        }
      })
    }
  }

};
```

4.1.2.3. Expected Results

If successful, console.log(mysql is connected successfully)

if failed, console.log(mysql connection error)

4.1.3. Database Validation test

4.1.3.1. Test case object

To check data in MySQL database.

4.1.3.2. Test inputs

SQL query command

4.1.3.3. Expected Results

if requested data existed, the data will be shown in screen, but if requested data doesn't exist, the data will not be shown in screen.

4.2. AWS API to DynamoDB

4.2.1. S3 upload

4.2.1.1. Test case object

To upload some data to AWS dynamo DB

4.2.1.2. Test input code

```
// Load the AWS SDK for Node.js
var AWS = require('aws-sdk');
// Set the region
AWS.config.update({region: 'REGION'});

// Create S3 service object
s3 = new AWS.S3({apiVersion: '2006-03-01'});

// call S3 to retrieve upload file to specified bucket
var uploadParams = {Bucket: process.argv[2], Key: '', Body: ''};
var file = process.argv[3];

// Configure the file stream and obtain the upload parameters
var fs = require('fs');
var fileStream = fs.createReadStream(file);
fileStream.on('error', function(err) {
  console.log('File Error', err);
});
uploadParams.Body = fileStream;
var path = require('path');
uploadParams.Key = path.basename(file);

// call S3 to retrieve upload file to specified bucket
s3.upload (uploadParams, function (err, data) {
  if (err) {
    console.log("Error", err);
  } if (data) {
    console.log("Upload Success", data.Location);
  }
});
```

4.2.1.3. Expected Results

Data in S3(in success). If fail, console log error message will be appear

4.2.2. AWS DynamoDB connect

4.2.2.1. Test case object

To connect to AWS DynamoDB

4.2.2.2. Test input code

```
const { DynamoDBClient, ListTablesCommand } = require("@aws-sdk/client-dynamodb");

(async () => {
  const client = new DynamoDBClient({ region: "us-west-2" });
  const command = new ListTablesCommand({});
  try {
    const results = await client.send(command);
    console.log(results.TableNames.join("\n"));
  } catch (err) {
    console.error(err);
  }
})();
```

4.2.2.3. Expected Results

If successful, console.log(results.TableNames).

If fail, console.log(err).

4.3. Video Analyzing

4.3.1. Connection test

4.3.1.1. Test case object

To check software interfaces of the Video Analyzing system.

4.3.1.2. Test inputs

Ping command

4.3.1.3. Expected Results

Ping status

4.3.2. Load test

4.3.2.1. Test case object

To check software interfaces of the Video Analyzing system’s load.

4.3.2.2. Test inputs

According to the system's numerical requirements, it should be able to accommodate about 5000 simultaneous connections and video information transmitted and backed up once every 10 seconds.

Set the sample environment with the same load with the above things(5000 simultaneous video input to the video analyzing system) and test.

4.3.2.3. Expected Results

If successful, the system runs normally.

If fail, time delay, network error, load error, etc.

4.4. Sharing information between DB

4.4.1. Connection test

4.4.1.1. Test case object

To check external software interfaces that connect SKKU DB and own DB.

4.4.1.2. Test inputs

Ping command

4.4.1.3. Expected Results

Ping status

4.4.2. Load test

4.4.2.1. Test case object

To check software interfaces of sharing information system’s load.

4.4.2.2. Test inputs

According to the system’s numerical requirements, it should be able to accommodate about 5000 simultaneous connections. Also, it should be able to transmit and back up video/answer information once every 10 seconds.

Send the sample query with the same load with about things and test.

4.4.2.3. Expected Results

If successful, the system runs normally.

If fail, time delay, network error, load error, etc.

5. Supporting Information

5.1. Document History

[Table 13] Document History

Date	Description	Writer
5.24	Add components to 2.1.1 and 3	Eunsu Kang
5.25	Add components to 2.1.1 and 3	SeungYeon Cho
5.26	Add components to 3.10	HyungJoon Joo
5.27	Add components about Database	JeongMin Lee
5.27	Add things to 4.3 ~ 4.4	Kyunghee Ko
5.28	Add things to 2.1.2	Kyunghee Ko
5.28	Add things to 1	Luke