The Hello Team Project Platform

Software Test Plan Document

2021.05.30.

Introduction to Software Engineering 41

TEAM 14 (Hello Team Project)

Team Leader Jaehyun Ju
Team Member Min Jang
Team Member Heesoo Jung
Team Member Byeongsu Woo
Team Member Seongwook Lim

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. Introduction

This chapter contains the purpose, scope, definitions, acronyms, abbreviation, references, and overview of this Software Test Plan Document for Hello Team Project.

1.1. Purpose

This Software Test Plan Document is for appropriate and specific testing to Hello Team Project. The main reader of this document is Team 14. Additionally, professors, TAs, and team members in the Introduction to Software Engineering class can be the main readers.

1.2. Scope

This Software Test Document is to be used by Software Engineering and Software Quality Engineering as a testing system which is the University online team project platform.

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**
UI	User Interface
AB	Apache Bench
API	Application Programming Interface
CRUD	Create, Read, Update, Delete

AWS	Amazon Web Service
ECS	Elastic Container Service
COTS	Commercial off the shelf
YCSB	Yahoo Cloud Serving Benchmark

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

[Table 2] Table of terms and definitions

Terms	Definitions
User	Someone who uses a system. Professors, TAs and students.
Classmate	Student who attends the same lecture.
Mattermost	An open source collaboration platform
Back-End	Application part that is not directly accessed by the user, such as the server and database
Front-End	The user interface, also known as the presentation layer of an application

Algorithm	A set of rules or procedures followed by a computer in problem-solving operations
Server	A computer or computer program which manages access to a centralized resource or service in a network
Software	The programs and other operating information used by a computer
Network	Connect devices together so that they can share information. In this system, it usually means internet

1.4. References

- Test plan document format

1.5. Overview

The remainder of this Software Test Plan Document includes five chapters. The second chapter provides an overall approach for all test methods, including software unit test methods and software interface test methods. The third chapter provides the details of the unit test methods, which is the validation of the various units. And the fourth chapter provides the details of interface test methods. The fourth chapter provides the details of the interface test methods. It includes checking the function and interfaces. All members contributed equally to the production of this project. We hope that you, the reader, enjoy viewing this document.

2. Approach

The project is to develop and design a web platform used for the introduction which makes students find team members easier, for the communication which makes students do a team project online easier, and for the rating which reduces free riders. The system should provide the information management place for individual students, information searching place for students who find team members, team making request function for students who want to make a team, team management place for professors who want to manage the teams in their lecture, communication place for professors and students to communicate with team members and professor during the project, and rating place for students who want to evaluate their team members after the team project. This test document presents the test methods for the functional and nonfunctional requirements that are described above, throughout unit test and interface test.

2.1. Test method

In this chapter, we describe the various tools and methods which we have applied to this project in the test phase.

2.1.1. Software unit test methods

We will use the partition testing for unit tests for the login page. The input and output group of the unit can be grouped by similar characteristic features. The group is called by domain, where we take the representee of them as test cases.

Inspection is the main method for checking the units. There are advantages of inspection over the actual test. The people -examining source representation- can check where the bugs and defects are easily, because errors cannot hide other errors. And there are many properties that we can’t check by test such as maintainability, portability and compliance with standards, so we have to inspect the code.

Hello project uses ‘go’ as backend language. Because COTS(mattermost) is made of ‘go’ language, using ‘go’ will make compatibility better. We will use Next.js which is the server side render framework of React.js. Functions and components which are made of ‘go’ will be tested by test command in ‘go’. Each domain group has their test case and will be tested using them. Because Next.js runs based on node.js, we will use the testing tool of node.js named Jest for testing functions and components.

2.1.2. Software interface test methods

Many of our functions, that are used for passing the value or parameter, thus we should check the error that misuses call by value and call by references. And we should check the name of the interfaces are properly named, for preventing the error on maintaining.

Users use REST API when they request CRUD works and get data from front-end to backend. If the user requests correctly, he will get 200 OK request. On the other hand, he will get an error response message. We will test it using postman.

Specifically, we will check these cases : normal case, wrong request which doesn't include necessary parameter or query string or body, request sent to URI, request sent to URI which doesn't have authority. We will make it as a domain group and test cases for each group. We can check the status code and response body through postman using these test cases.

When lots of clients access the system, the server sends a number of CRUD requests to the database, with small time latency. In this case, transactions occur in the database, which degrades the performance of the database. To minimize this happening, test the database performance and correct architecture where the high latency and 99% of latency occurs.

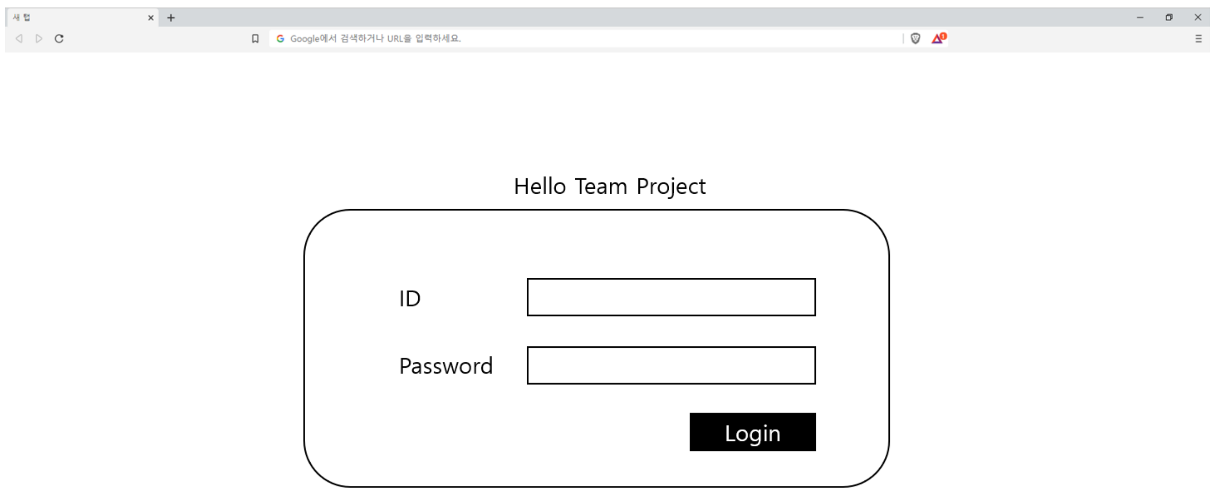
To test the connection performance of server and database, we use YCSB. Set the working proportion of read and write in the database and do CRUD tasks to check the latency and throughput. Set the domain group - when read, write proportion is same, when read is 100%, when write is 100% and read is 95% and write is 5%.

Do AB test to check the performance of backend server latency throughout REST API and maximum requests that backend server processes simultaneously. Using AB test, check the performance and stress of the backend, by API.

Evaluation standard of the stress test is based on the usage of CPU which is set by scale out policy by the Auto Scaling Groups of AWS. We gradually increase the simultaneous request by using AB tool and make ECS's CPU usage up to the amount we scaled out. If the amount of request does not reach the half of our expected simultaneous users, we optimize the corresponding REST API.

3. Software Unit Test

3.1. Login Page



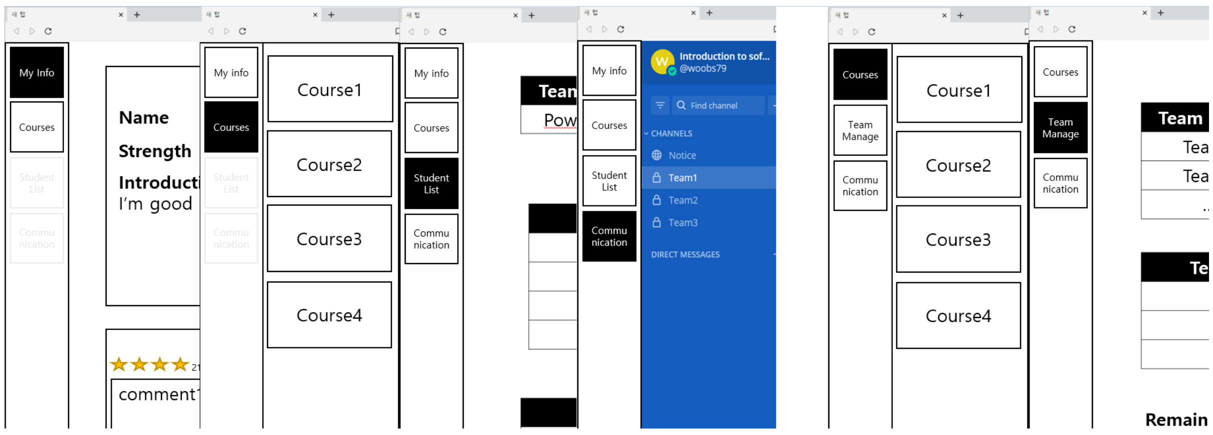
[Figure 1] Login Page

We should use a partition test, and there are eight domains.

1. no input. -> Give no input message
2. forbidden characters at ID, valid Passwords.
   -> Don't process it. Give "invalid ID or PW" message
3. valid ID and forbidden characters at Passwords.
   -> Don't process it. Give an "invalid ID or PW" message.
4. both invalid ID and PW using forbidden characters.
   -> Don't process it. Give an "invalid ID or PW" message.
5. wrong ID and valid PW
   -> Give "invalid ID or PW" message.
6. valid ID and wrong PW.
   -> Give "invalid ID or PW" message.
7. valid ID and valid PW.
   -> Give "Login success!" message.
8. very very long ID or PW.
   -> Don't process it. Give an "invalid ID or PW" message.

By using these domains, we can have a partitioning test.

3.2. Navigation bar



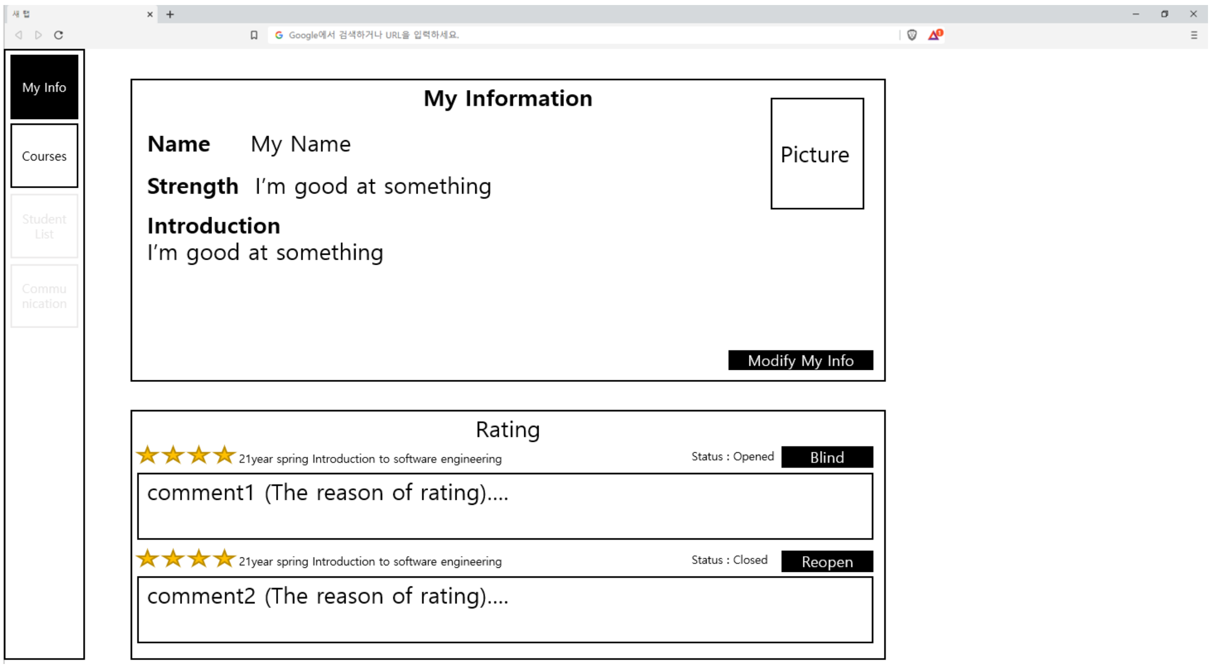
[Figure 2] Navigation bar

Navigation bar is used to go to the corresponding page.

We have to check the below checklist.

1. If the user is a student, My info, Courses, Student list, and Communication buttons are available. If the user is a professor, Courses, Team manage, and Communication buttons are available.
2. Student list and Communication buttons are only available when the user is in a certain course. Thus, when the user is in the My info page, which means currently the student does not belong to any course pages. Thus, Student list and Communication buttons have to be disabled.
3. When the user clicks My Info button, the user has to be navigated to the user's information page.
4. When the user clicks Courses, the course selecting bar shall appear.
5. When the user clicks the Student list button, the student list page of the current course shall be shown.
6. When the user clicks the Communication button, the communication page of the current course shall be shown.
7. When a professor logs on, the default selected course is the first course of the professor.
8. When the user clicks the team manage button, the team managing page of the current course shall be shown.

3.3. My Information Page



My Information

Name My Name

Strength I'm good at something

Introduction
I'm good at something

Rating

21year spring Introduction to software engineering Status : Opened

comment1 (The reason of rating)....

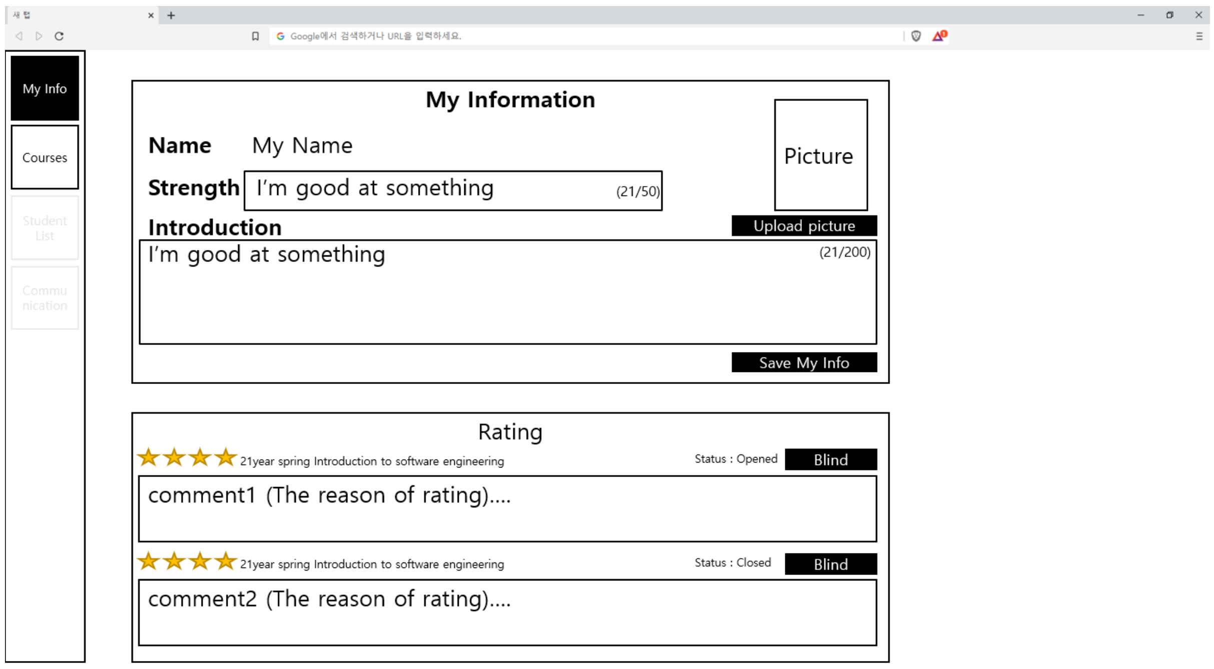
21year spring Introduction to software engineering Status : Closed

comment2 (The reason of rating)....

[Figure 3] My Information Page

My information page is the page that shows the current user's information. In this page we have to check the below checklist.

1. Check the page goes to the state described in figure 4 when the user clicks the "Modify My info" button.
2. Check the comment would be blind to others and the button would change to "Reopen" when the user clicks the "blind" button.
3. Check the comment would be open to others and the button would change to "Blind" when the user clicks the "Reopen" button.



My Information

Name My Name

Strength I'm good at something (21/50)

Introduction

I'm good at something (21/200)

Rating

21year spring Introduction to software engineering Status : Opened

comment1 (The reason of rating)....

21year spring Introduction to software engineering Status : Closed

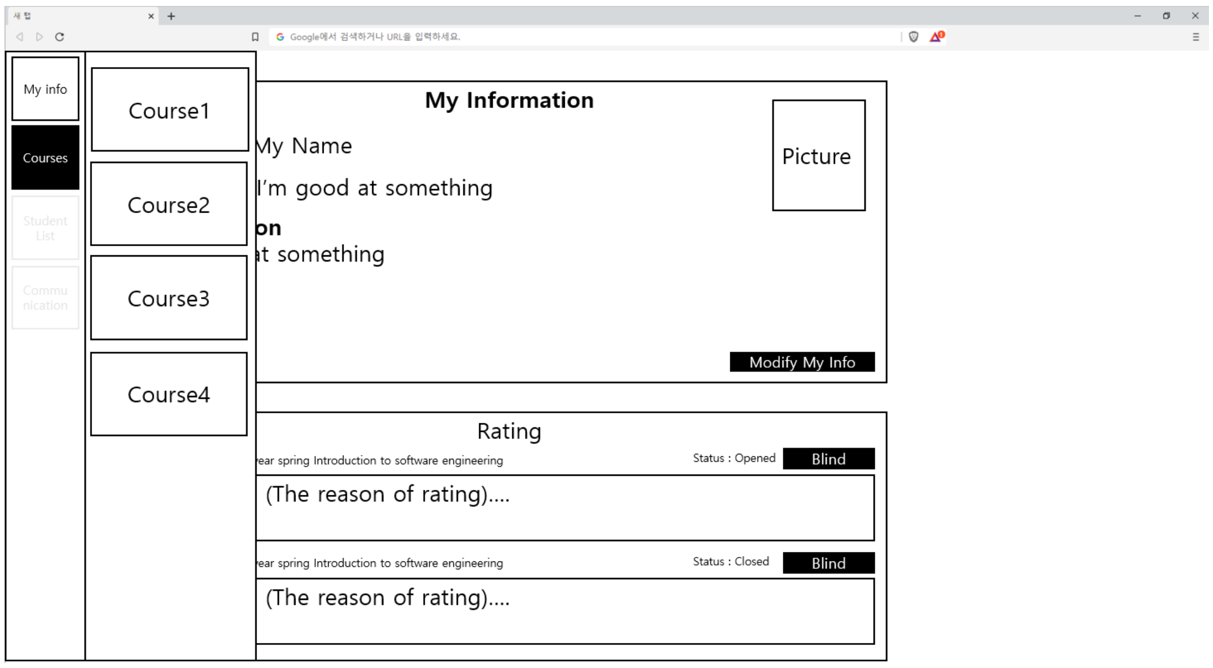
comment2 (The reason of rating)....

[Figure 4] Modify My Information

Figure 4 is the page when we clicked "Modify my Info". The user can change the profile on the page. In this page we have to check the below checklist.

1. Check the page saves the valid change of the text, including strength, introduction and pictures.
2. Check the page gives the message "The text is too long" when there is one or more text that is over the size of the maximum text.
3. Check the page gives the message "The picture size is too big" when the user uploaded too big file exceeds the maximum picture size.
4. Check the modified text or newly written text is well stored in our database "profile" table.

3.4. Select a Course

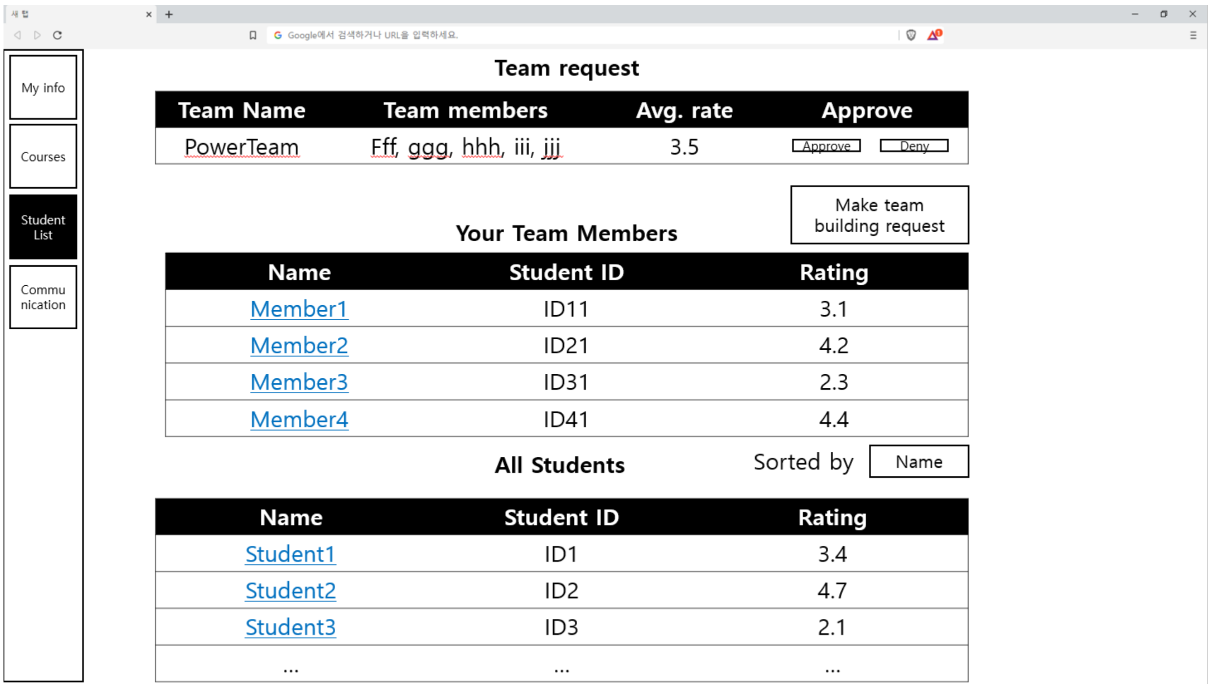


[Figure 5] Select a Course

This is the page when we clicked the "courses" button, and the navigation bar is shown. We can choose one of the courses that we are taking. In this page we have to check the below checklist.

1. Check that all courses that the user attends are shown properly.
2. Check that the page goes to the "Student Information List Page" of the selected course, if a student clicks the particular course.
3. Check that the page goes to the "Team Management Page" of the selected course, if a professor clicks the particular course.

3.5. Student Information List Page



Team request

Team Name	Team members	Avg. rate	Approve
PowerTeam	Fff, ggg, hhh, iii, jjj	3.5	Approve Deny

Your Team Members

Name	Student ID	Rating
Member1	ID11	3.1
Member2	ID21	4.2
Member3	ID31	2.3
Member4	ID41	4.4

All Students

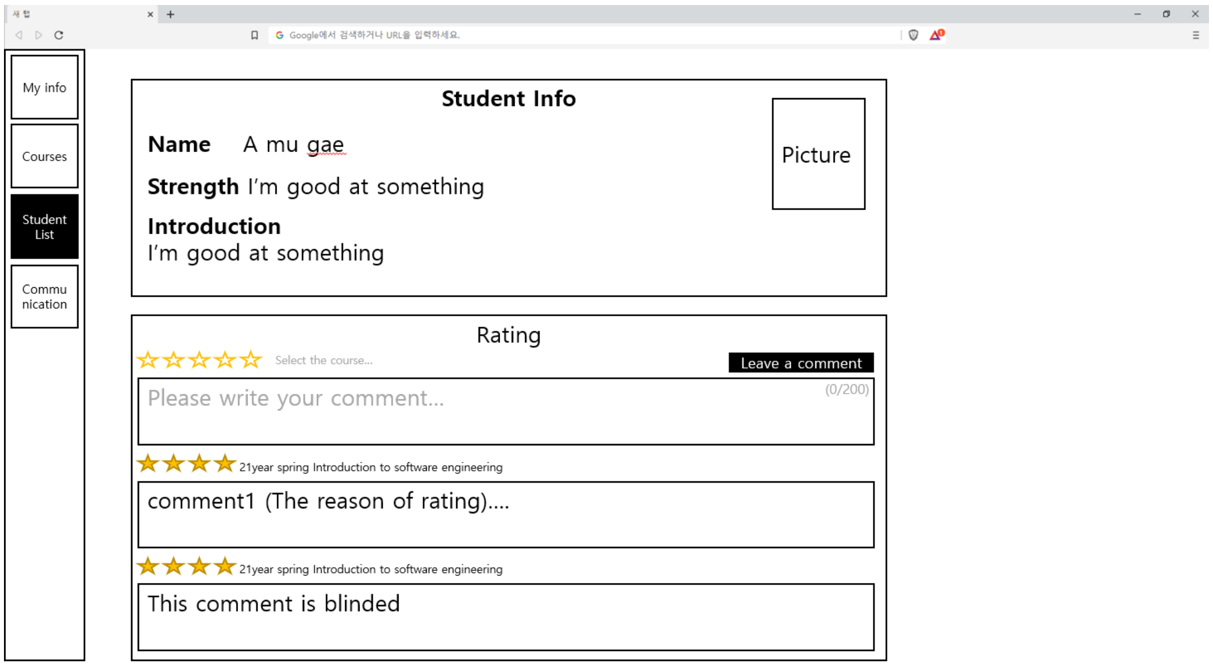
Name	Student ID	Rating
Student1	ID1	3.4
Student2	ID2	4.7
Student3	ID3	2.1
…	…	…

[Figure 6] Student Information List Page

This is the page where we can see the classmates that are taking the same class. In addition, we can deny or accept the team request. And there is a user's team member information. In this page we have to check the below checklist.

1. Check that the information of all students who attend the course is loaded properly in "All students" table.
2. Check that the team canceling requests are sent to all candidates properly when one of the candidates clicks the "deny" button.
3. Check that the notification is sent to every candidate of the team when every candidate clicks the "accept" button.
4. Check the popup page described in figure 8 appears when the user clicks the "Make team building request."
5. Check the sorting criteria can be changed when the user clicks "Name".
   (The criteria is changed to student rating. The button is changed to "Rating".)
6. Check the sorting criteria can be changed when the user clicks "Rating".
   (The criteria is changed to student name. The button is changed to "Name".)

3.6. Student Information Page



Student Info

Name A mu gae

Strength I'm good at something

Introduction
I'm good at something

Rating

☆☆☆☆☆ Select the course...

Please write your comment... (0/200)

★★★★ 21year spring Introduction to software engineering

comment1 (The reason of rating)....

★★★★ 21year spring Introduction to software engineering

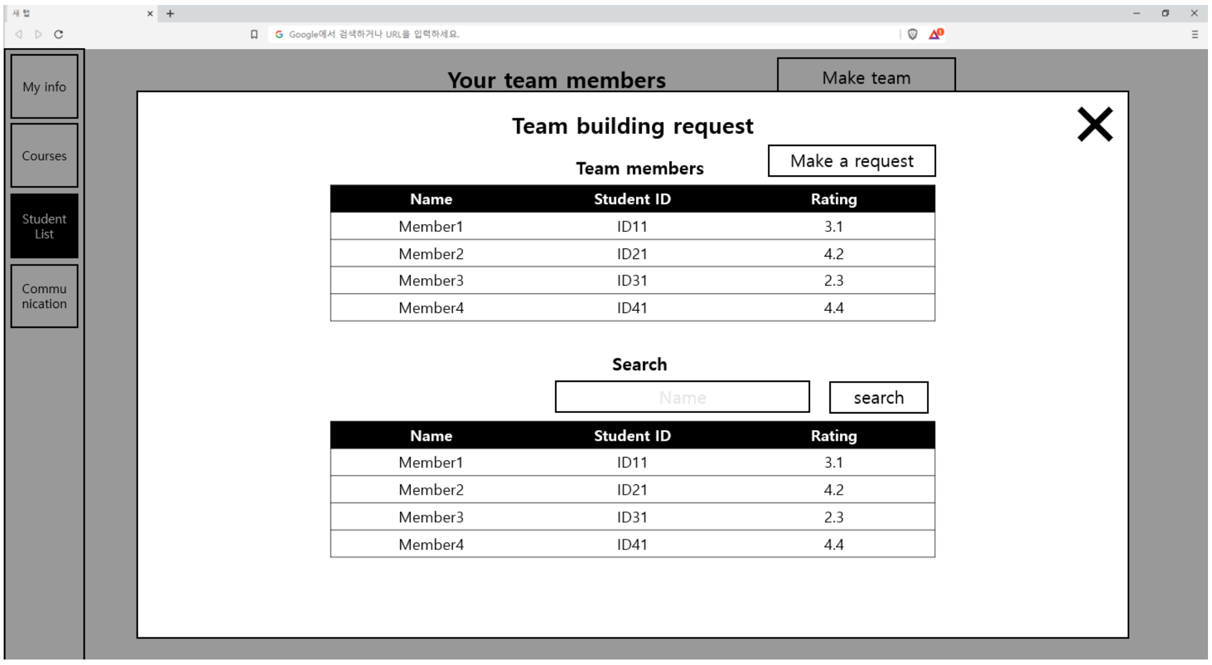
This comment is blinded

[Figure 7] Student Information Page

This is the other student's page (Not the student himself), where we can see his/her comments. In this page we have to check the below checklist.

1. Check the "Leave a comment button" is shown, when the student is the team member of the user.
2. Check the "Text is too long" message is shown when the user writes too long text in our comment, exceeding the maximum text size.
3. Check the "Enter the text" message is shown when the user enters no text in comment.

3.7. Team Building Request Page



Team members

Name	Student ID	Rating
Member1	ID11	3.1
Member2	ID21	4.2
Member3	ID31	2.3
Member4	ID41	4.4

Name	Student ID	Rating
Member1	ID11	3.1
Member2	ID21	4.2
Member3	ID31	2.3
Member4	ID41	4.4

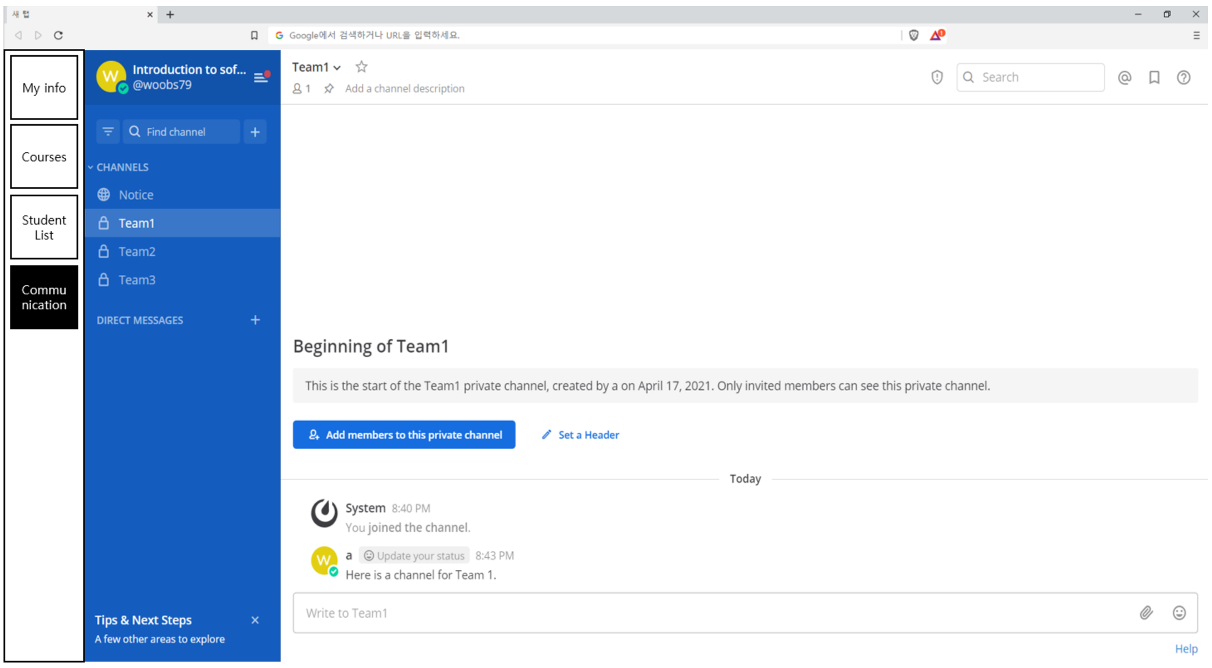
[Figure 8] Team Building Request Page

Team building request page is a page where a student makes a team building request. In this page, we have to check the below checklist.

1. When the user tries to search a student, check that the system ignores the input that is none, very very long, and includes unaccepted characters. Partitioning tests are needed.
    a. Check when input length is zero.
    b. Check when input length is limit length
    c. Check when input length is the half of limit length
    d. Check when input length exceeds the limit length
2. When the user clicks the search button, check that the system shows the student list on the below table. Names are matched if the user's input is a substring of the name. Also check whether the all searched students are available, which means they are not members of any team.
3. Check the program adds the member to the above table, when the user clicks the student name at the below table.

4. When the user tries to click the make a request button, check the team member is valid. And check the request is sent to the candidates properly.
5. When the user clicks the x button, the popup window shall be closed properly.

3.8. Communication Page



Beginning of Team1

This is the start of the Team1 private channel, created by a on April 17, 2021. Only invited members can see this private channel.

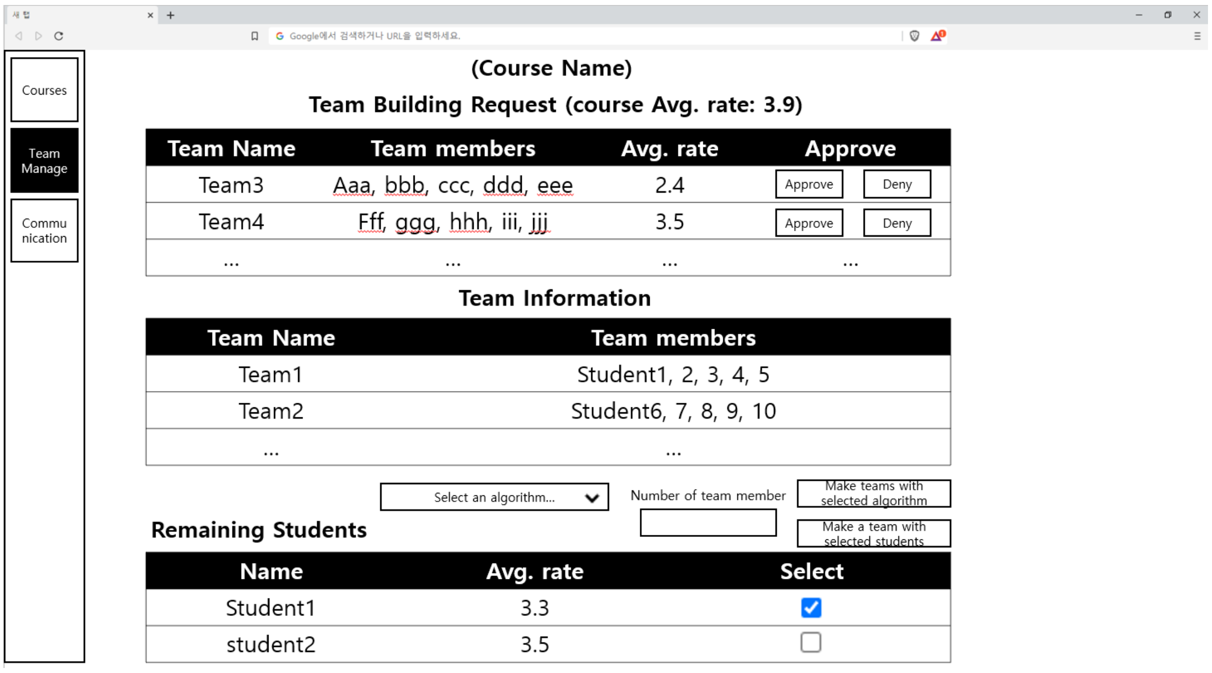
System 8:40 PM
You joined the channel.

a Update your status 8:43 PM
Here is a channel for Team 1.

[Figure 9] Communication Page

The communication page is a page that students can communicate with each other. The page uses the Mattermost platform and the Mattermost is a COTS system. Thus, all we have to check is just checking whether the page is loaded well when the user clicks the Communication button on the left navigation bar.

3.9. Team Management Page


새 탭
Google에서 검색하거나 URL을 입력하세요.
Courses
Team Manage
Commu nication
Select an algorithm...
Number of team member
Make teams with selected algorithm
Make a team with selected students

(Course Name)

Team Building Request (course Avg. rate: 3.9)

Team Name	Team members	Avg. rate	Approve
Team3	Aaa, bbb, ccc, ddd, eee	2.4	Approve Deny
Team4	Fff, ggg, hhh, iii, jjj	3.5	Approve Deny
…	…	…	…

Team Information

Team Name	Team members
Team1	Student1, 2, 3, 4, 5
Team2	Student6, 7, 8, 9, 10
…	…

Remaining Students

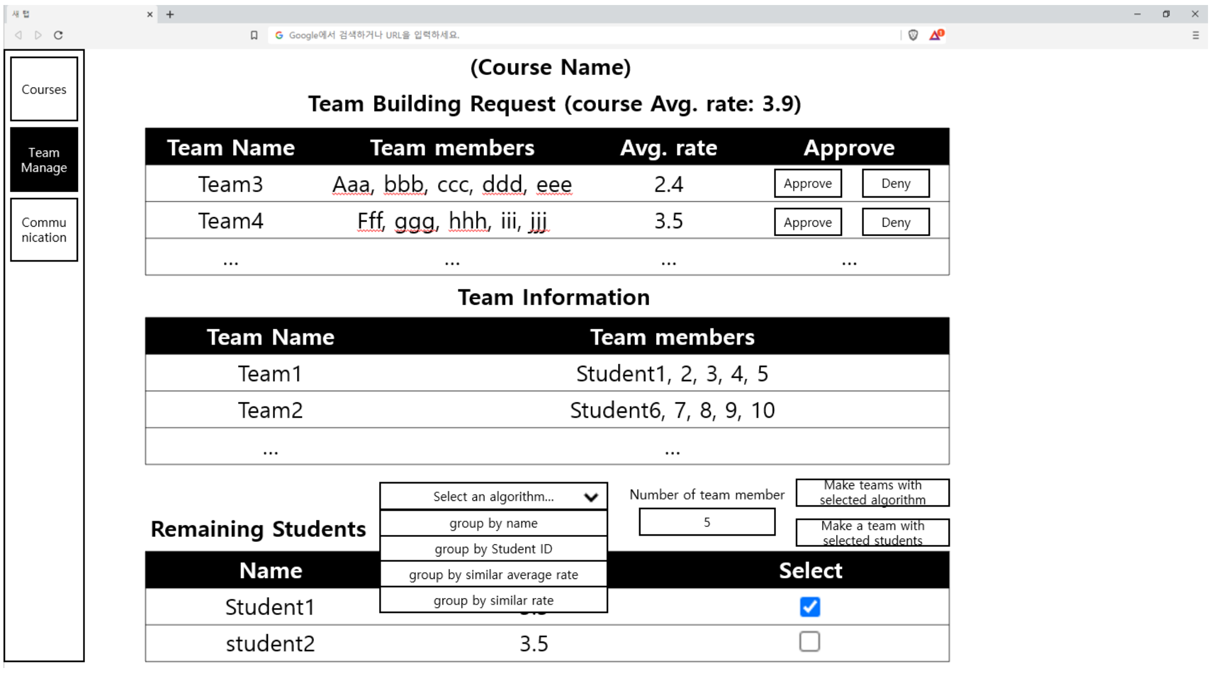
Name	Avg. rate	Select
Student1	3.3	✓
student2	3.5	☐

[Figure 10] Team Management Page

Team management page is the page managing the teams in the corresponding course. There are three tables in the team management page. In this page we have to check the below checklist.

1. Check that the Avg.rate above is well printed as the average rate of the current existing teams.
2. Check that the team is made and displayed at the "Team Information" table when the professor clicks the approve button. Check the notification is sent to the students that are part of the team.
3. Check that the team is declined when the professor clicks the deny button. Check the notification is sent to the students that are part of the team.
4. Check that every team that currently exists is shown at the "Team Information" table.
5. Check that the manual team matching is done when the professor checks the list of students and clicks the "Make a team with selected students" button.
6. Check that the system gives a "Please check the students" message when none of the students are checked and the professor clicks the "Make a team with selected students" button.

7. Check that the system gives a “Please check the number of team members” if the professor clicks the button with inappropriate value. Partitioning tests are needed.
    a. check the value is not integer.
    b. check the value is zero.
    c. check the value is three. (value between zero and six)
    d. check the value is six (maximum team size).
8. Check that the system gives a “Please select an algorithm” message if the professor clicks “Make teams with the selected algorithm” without selecting an algorithm.
9. Check that the system makes an automatic team with remaining students by selected algorithm, when the professor clicks the “Make teams with selected algorithm” with appropriate value and after he/she selected algorithm.



(Course Name)

Team Building Request (course Avg. rate: 3.9)

Team Name	Team members	Avg. rate	Approve	
Team3	Aaa, bbb, ccc, ddd, eee	2.4	Approve	Deny
Team4	Fff, ggg, hhh, iii, jjj	3.5	Approve	Deny
…	…	…	…	

Team Information

Team Name	Team members
Team1	Student1, 2, 3, 4, 5
Team2	Student6, 7, 8, 9, 10
…	…

Remaining Students

Name		Select
Student1	[illegible]	✓
student2	3.5	☐

[Figure 11] Select team making algorithm

4. Software Interface Test

4.1. Log-In

The Log-In class deals with user account information, which is obtained by a school database. After student login, the server can give the user profile page. Therefore, there are some interactions between the user interface and the database through the backend. We need to check:

1. Check if the ID and password from the school database is well received by the backend server.
2. Check when the Log-In process successes, whether the backend server gives the right information about the user profile page received from our database.

4.2. Mattermost Channel Generating System

The Mattermost Channel Generating system has an interaction between backend and Mattermost server. If the team member is assigned by the professor or made by themselves, information about the team member, professor and TA should be given to the Mattermost server. With the given information, Mattermost server will make a channel for them. We need to check:

1. Ideal case:

   Input: All independent student, professor and TA information is given to the mattermost server

   Output: Sends the Mattermost channel with all student, professor TA's in the channel.

2. Check if the information we want is well sent to the backend from the Database when the professor clicks on the approve button or when the professor clicks to match team members.
3. Check if all the information is received and sent well between backend and Mattermost server.

4.3. Getting Information From The University

Since our software uses the university account, we need to bring the course list, student's basic information and professor's basic information. Therefore, we need some interaction with the university account and our database. We need to check:

1. Check if the course list which is open this semester is well stored in our "course" table
2. Check if the basic information about the student which are student name, student number, email, account ID and password and other basic information is well stored in our database.

4.4. Team Matching System

Team matching system is a system where a professor matches the team member for the leftover students who did not make their team. Team matching system has an interaction between user interface and backend. From the information which are how to group students and group with how many students given by the user interface, the backend system should automatically make groups. We need to check:

1. Check if all the students shown at the user interface attend that course by checking the database.
2. Check the grouping algorithm is grouping well with the selected algorithm. Therefore, we need to check the group by name, student ID, average rate of group, similar rate of group is well performed.
3. Check if the result of the automatic team generation satisfies the condition which is the number of members in one group and to make sure that there are no leftover students.

4.5. Comment Blind System

Comment blind system is a system that blinds the comment if the user requested to hide the comment. Comment blind system has an interaction between the user interface and database through the backend. When the user requests to blind the comment, we change the hide/open flag at the database. Backend checks the flag and changes the opened comment to blind comment. We need to check:

1. Check if the hide/open flag is well changed when the user clicks on blind button or reopen button.
2. Check the comment is well hidden when the hide/open flag is set to hide and check the comment is well shown when the hide/open flag is set to open.

4.6. Team Building Request System

Team request system is a system that makes the team request and manages the acceptance of the team. The main point of this interface of this system is checking the notification is well conveyed to the appropriate user. We need to check:

1. Check if the notification is conveyed to team candidates when the professor declines the team request. Check the team request is deleted in the database.

2. Check if the notification is conveyed to team candidates when the professor accepts the team request. Also, check whether the team information is well stored in the database.

3. Check if the notification is conveyed to team candidates when a student makes the team requests. The students of the team would accept or decline the request. Check the team request is saved in the database.

4. Check if the notification is conveyed to team candidates when one student declines the team requests. Check the team request is deleted in the database.

5. Check the information is conveyed to the professor when all students accept the team request. Check the notification is conveyed to the team candidates.

5. Supporting Information

5.1. Document History

[Table 2] Document History

Date	Version	Description	Writer
2021/05/22	0.1	Addition of 1, 2, 3	Byeongsu Woo Jaehyun Ju
2021/05.24	0.2	Addition of 4.1~4.5	HeeSoo Jung
2021/05/25	0.3	Revision of 2, Addition of 4.6	Seongwook Lim Min Jang
2021/05/25	1.0	Distribution	