Software Requirements Specification

for

Edumon

Version 1.1 approved

Prepared by Group1:

Beh Chee Kwang Nicholas (U1824125F)
Chen Xueyao (U1922640G)
Chong Jing Hong (U1922300B)
Goh Hong Xiang, Bryan (U1920609E)
Huang Shaohang (U1921245D)
Lim Jun Wei (U1922756C)
Muhammad Al-Muhazerin (U1921191L)
Quah Dian Wei (U1920106C)
Shauna Tan Li-Ting (U1840754E)
Swa Ju Xiang (U1822040G)

14/11/2021

Table of Contents

Revision History

Name	Date	Reason for change	Version
Everyone	30/08/2021	Initial draft	0.1
Everyone	12/09/2021	Updated for submission	1.0
Everyone	14/11/2021	Final update	1.1

1. Introduction

1.1 Purpose

This document will provide detailed documentation of all the intended features of our application, Edumon. This document provides information on the target audience, features, interfaces and the relevant design considerations for the system. It also explains the system constraints, the different states of the system, as well as how it will interact with external inputs.

1.2 Document Conventions

These are the conventions used in this SRS:

- Font type: Arial
- Font size (main header): 18
- Font size (sub header): 14
- Font size (sub sub header): 12
- Font size (body): 12
- Bold letters represent headers/important information in the text body.

1.3 Intended Audience & Reading Suggestions

This document is intended for the Teaching, Learning and Pedagogy Division (TLPD), the professors and teaching assistants of the SCSE faculty from Nanyang Technological University, and the developers of this application.

Sequence for Reading:

1. Section 2: Overall Description

   This section will give an overall description and overview of the functions, requirements, and operating environment of the application.

2. Section 3: System Features / Functional Requirements

   This section will give a description of the main functions of the application.

3. Section 4: External Interface Requirements

   This section will explain the interfaces required for the application.

4. Section 5: Other Nonfunctional Requirements

   This section will give a description of the requirements that are not specific to the application.

5. Section 6: Other Requirements

   This section will give a description of the logical database requirement for the application.

1.4 Project Scope

Edumon plans to educate the end-users of this game on software engineering practices through fun and interactive manners. The eventual goal of this project is to revolutionize mundane learning processes by inducing game-like elements into the learning journey.

2. Overall Description

2.1 Product Perspective

The application is a self-contained game to gamify and socialise the teaching and learning of software engineering courses, in particular portions about the Software Development Life Cycle (SDLC). Students can learn through playing the game and compete with other fellow students, while Teachers can monitor and assess students' mastery of the course content through the use of data analysis.

2.2 Product Features

Students must be able to:

1. Choose a topic and play the corresponding quizzes for that topic
2. Complete and submit assignments that were issued by teachers
3. Create challenges that can be attempted by other students
4. View the leaderboard
5. Edit their profile

Teachers must be able to:

1. View and assess the progress of students
2. Edit their profile
3. Edit the gym questions
4. Set assignments for students to do
5. Generate a summary report to gauge students' progress

2.3 User Classes and Characteristics

Student
Students use the application for the purposes as mentioned in Section 2.2 Product Features.

Teacher
Teachers use the application for the purposes as mentioned in Section 2.2 Product Features.

Developer
Developers are involved in the creation, development, and future maintenance of the game.

2.4 Operating Environment

The Game Application is designed to be cross-platform and runs on computers which satisfy the minimum system requirements of:

	Windows	macOS	Linux
Operating System Version	Windows 7 (SP1+) and Windows 10	High Sierra 10.13+	Ubuntu 20.04, Ubuntu 18.04 and CentOS7
CPU	x86, x64 architecture with SSE2 instruction set support	Apple Silicon, x64 architecture with SSE2	x64 architecture with SSE2 instruction set support
Graphics API	DX10, DX11, DX12 capable	Metal capable Intel and AMD GPUs	OpenGL 3.2+, Vulkan capable

Include any other necessary requirements of other components.

2.5 Design & Implementation Constraints

Frontend implementation:

1. The game client is developed using the Unity game engine.
2. Unity uses C# as its main language for development.
3. Unity compiles and builds an executable file for each supported platform.
4. The version of Unity used is 2020.3.18f1.

Backend server implementation:

1. The backend server is powered by the Firebase platform.
2. Firebase provides services for database storage and user authentication.
3. The database structure shall be designed in a manner that is well-defined but leaves room for future modification without requiring significant overhaul.
4. The version of Firebase SDK for Unity used is 8.2.0.

2.6 User Documentation

The user documentation will consist of a user manual detailing the full functionality of the Game Application. It will contain guided instructions for both Students and Teachers with screenshots included to enable further ease of reference.

2.7 Assumptions and Dependencies

It is assumed that the end-user is equipped with a basic understanding of how to navigate and use desktop operating systems, through the use of commonly used input devices such as keyboards, mice, touchpads and touchscreens. The end-user must also have a stable internet connection.

3. System Features

3.1 Register

3.1.1 Description and Priority

The application allows users, both students and teachers, to register accounts using their email and password to enter the game.

Priority: 10/10

3.1.2 Stimulus/Response Sequences

Main Flow:

1. User selects the option to register an account.
2. System displays the ‘Create Account’ screen.
3. Users input their usernames, student IDs (for students), emails and passwords in the respective fields and indicate if the account is for students or teachers.
4. System displays the ‘Account Created’ screen.

Alternative Flows:

AF-S3: If user inputs invalid email or email that is already in use:

1. System displays an “Invalid email or email already in use” message.
2. Return to step 2 in the main flow.

AF-S3: If user inputs a username that is already in use:

1. System displays a “Username already in use” message.
2. Return to step 2 in the main flow.

AF-S3: If user inputs a password that is not at least 12 characters long, or does not have at least 1 uppercase letter, 1 lowercase letter, 1 number and 1 special character:

1. System displays “Passwords must be at least 12 characters long, and have at least 1 uppercase letter, 1 lowercase letter, 1 number and 1 special character” message.
2. Return to step 2 in the main flow.

AF-S3: If user passwords do not match:

1. System displays a “Passwords do not match” message.
2. Return to step 2 in the main flow.

3.1.3 Functional Requirements

3.1.3.1 The system shall allow a new student user to register for a new account.

3.1.3.1.1 The new student user must enter their student ID, school email, username and password.

3.1.3.1.1.1 The system shall validate that the student ID and email entered are valid.

3.1.3.1.1.2 The system shall validate that the email entered is a school email.

3.1.3.1.1.3 The system shall ensure that the student ID and email entered are not in the system.

3.1.3.1.1.4 The system shall ensure that the username entered has at least 1 character.

3.1.3.1.1.5 The system shall ensure that the username entered is not in the system.

3.1.3.1.1.6 The system shall ensure that the password entered is at least 12 characters long, and has at least 1 uppercase letter, 1 lowercase letter, 1 number and 1 special character.

3.1.3.1.2 The system shall ensure that the password field input is masked with a “*” character for each input character (e.g. “password” will be masked to “********”).

3.1.3.2 The system shall allow a new teacher user to register for a new account.

3.1.3.2.1 The new teacher user must enter their teacher ID, school email and password.

3.1.3.2.1.1 The system shall validate that the teacher ID and email entered are valid.

3.1.3.2.1.2 The system shall validate that the email entered is a school email.

3.1.3.2.1.3 The system shall ensure that the teacher ID and email entered are not in the system.

3.1.3.2.1.4 The system shall ensure that the password entered is at least 12 characters long, and has at least 1 uppercase letter, 1 lowercase letter, 1 number and 1 special character.

3.1.3.2.2 The system shall ensure that the password field input is masked with a “*” character for each input character (e.g. “password” will be masked to “********”).

3.1.3.3 The system shall register the new student or teacher user in the database.

3.2 Login

3.2.1 Description and Priority

The application allows users (students and teachers) to login to their existing account.

Priority: 10/10

3.2.2 Stimulus/Response Sequences

Main Flow:

1. Users select the login option.
2. System displays the ‘Login’ screen.
3. Users input their valid email addresses and password.
4. System displays the corresponding home pages after verifying account credentials.

Alternative Flows:

AF-S3: If the user inputs an invalid email address:

1. System displays an “Invalid email address” message.
2. Return to step 2 in the main flow.

AF-S3: If user inputs the wrong password:

1. System displays a “Wrong password” message.
2. Return to step 2 in the main flow.

3.2.3 Functional Requirements

3.2.3.1 The user must be logged in to use the system.

3.2.3.2 The system must allow the user to login with their pre-existing account.

3.2.3.2.1 The system must validate that both the email address and password fields are not empty.

3.2.3.2.2 The system must validate that the email address entered is a valid email address and is linked to a pre-existing account.

3.2.3.2.3 The system must validate that the email address and password are correct (they match each other in the database).

3.2.3.2.4 The system must prompt an error if the user entered the wrong email address or password.

3.3 Edit Student Profile

3.3.1 Description and Priority

The “Edit Profile Page” function allows students to change their usernames, email, password and student ID.

Priority: 6/10

3.3.2 Stimulus/Response Sequences

Main Flow:

1. Students (“Users”) log in to their admin/user accounts respectively with valid email addresses and passwords.
2. System verifies accounts credentials and displays the home page.
3. Users select the “Profile Page” option.
4. Users select the “Edit Profile” option.
5. Users make amendments to their current username, email, password and student ID.
6. Users select the “Edit” option.
7. The system saves new profile information and displays “"Successfully change username, studentId, email, and password"

Alternate Flows:

AF-S5: Passwords do not match:

1. System displays a “Passwords do not match” message.
2. Return to step 5 of main flow

AF-S5: Password does not contain uppercase, lowercase, numbers and special characters:

1. System displays a “Password does not contain uppercase, lowercase, numbers and special characters” message.
2. Return to step 5 of main flow

AF-S5: Password is less than 12 characters long:

3. System displays a “Password is less than 12 characters” message.
4. Return to step 5 of main flow

3.3.3 Functional Requirements

3.3.3.1 The system shall display a “Profile Page” option in the homepage of both Teacher and Student accounts that redirects users to their personal profile pages.

3.3.3.2 The system shall display an “Edit Profile” option that allows users to edit their username and password.

- 3.3.3.2.1 The system shall display an “Edit Username” option that allows users to edit their username.
  - 3.3.3.2.1.1 The system shall verify the user input for the new username.
    - 3.3.3.2.1.1.1 The system shall display an error message if the “Username” field is left blank: “Please fill in your new username”.
    - 3.3.3.2.1.1.2 The system shall display an error message if the new username is not unique: “Username already in use”.
    - 3.3.3.2.1.1.3 The system shall display an error message if no amendments have been made to the user’s username: “New username is identical to your old username”.
  - 3.3.3.2.1.2 Users must be redirected back to their profile pages once their usernames are successfully changed.
- 3.3.3.2.2 The system shall display a “Change Password” option that allows users to edit their password.
  - 3.3.3.2.2.1 The system shall display input fields for “Username”, “Email”, “Password” and “Re-enter Password”.
    - 3.3.3.2.2.1.1 The system shall verify the user input for the new username.
      - 3.3.3.2.2.1.1.1 The system shall display an error message if the “Username” field is left blank: “Please fill in your new username”.

3.3.3.2.2.1.1.2 The system shall display an error message if the new username is not unique: “Username already in use”.

3.3.3.2.2.1.1.3 The system shall display an error message if no amendments have been made to the user’s username: “New username is identical to your old username”.

3.3.3.2.2.1.2 The system shall verify the user input for the new password.

3.3.3.2.2.1.2.1 The system shall verify that the “Password” field is not left blank. If it is left blank, the system shall display an error message: “Please fill in your new password”.

3.3.3.2.2.1.2.2 The system shall ensure that the password field input is masked with a “*” character for each input character (e.g. “password” will be masked to “********”).

3.3.3.2.2.1.2.3 The system shall ensure that the password entered is at least 12 characters long, and has at least 1 uppercase letter, 1 lowercase letter, 1 number and 1 special character. If the entered password is not of this format, the system shall display an error message.

3.3.3.2.2.1.3 The system shall verify the user input for the “Re-Enter Password”.

3.3.3.2.2.1.3.1 The system shall verify that the “Re-Enter Password” field is not left blank. If it is left blank, the system shall display an error message: “Please re-type in your password”.

3.3.3.2.2.1.3.2 The system shall verify that the input is identical to the “Password” input. If it is different, the system shall display an error message “Password does not match”.

3.3.3.2.2.1.4 Users must be redirected back to their profile pages once their usernames and/or passwords are successfully changed.

3.4 Edit Teacher Profile

3.4.1 Description and Priority

The “Edit Profile Page” function allows teachers to change their usernames, email, password.

Priority: 6/10

3.4.2 Stimulus/Response Sequences

<u>Main Flow:</u>

1. Teachers log in to their admin/user accounts respectively with valid email addresses and passwords.
2. System verifies accounts credentials and displays the home page.
3. Users select the “Profile Page” option.
4. Users select the “Edit Profile” option.
5. Users make amendments to their current username, email, password and student ID.
6. Users select the “Edit” option.
7. The system saves new profile information and displays “"Successfully change username, studentId, email, and password"

<u>Alternate Flows:</u>

AF-S5: Passwords do not match:

1. System displays a “Passwords do not match” message.
2. Return to step 5 of main flow

AF-S5: Password does not contain uppercase, lowercase, numbers and special characters:

1. System displays a “Password does not contain uppercase, lowercase, numbers and special characters” message.
2. Return to step 5 of main flow

AF-S5: Password is less than 12 characters long:

1. System displays a “Password is less than 12 characters” message.
2. Return to step 5 of main flow

3.4.3 Functional Requirements

3.4.3.1 The system shall display a “Profile Page” option in the homepage of both Teacher and Student accounts that redirects users to their personal profile pages.

3.4.3.2 The system shall display an “Edit Profile” option that allows users to edit their username and password.

3.4.3.2.1 The system shall display an “Edit Username” option that allows users to edit their username.

3.4.3.2.1.1 The system shall verify the user input for the new username.

3.4.3.2.1.1.1 The system shall display an error message if the “Username” field is left blank: “Please fill in your new username”.

3.4.3.2.1.1.2 The system shall display an error message if the new username is not unique: “Username already in use”.

3.4.3.2.1.1.3 The system shall display an error message if no amendments have been made to the user’s username: “New username is identical to your old username”.

3.4.3.2.1.2 Users must be redirected back to their profile pages once their usernames are successfully changed.

3.4.3.2.2 The system shall display a “Change Password” option that allows users to edit their password.

3.4.3.2.2.1 The system shall display input fields for “Username”, “Email”, “Password” and “Re-enter Password”.

3.4.3.2.2.1.1 The system shall verify the user input for the new username.

3.4.3.2.2.1.1.1 The system shall display an error message if the “Username” field is left blank: “Please fill in your new username”.

3.4.3.2.2.1.1.2 The system shall display an error message if the new username is not unique: “Username already in use”.

3.4.3.2.2.1.1.3 The system shall display an error message if no amendments have been made to the user’s username: “New username is identical to your old username”.

3.4.3.2.2.1.2 The system shall verify the user input for the new password.

3.4.3.2.2.1.2.1 The system shall verify that the “Password” field is not left blank. If it is left blank, the system shall display an error message: “Please fill in your new password”.

3.4.3.2.2.1.2.2 The system shall ensure that the password field input is masked with a “*” character for each input character (e.g. “password” will be masked to “********”).

3.4.3.2.2.1.2.3 The system shall ensure that the password entered is at least 12 characters long, and has at least 1 uppercase letter, 1 lowercase letter, 1 number and 1 special character. If the entered password is not of this format, the system shall display an error message.

3.4.3.2.2.1.3 The system shall verify the user input for the “Re-Enter Password”.

3.4.3.2.2.1.3.1 The system shall verify that the “Re-Enter Password” field is not left blank. If it is left blank, the system shall display an error message: “Please re-type in your password”.

3.4.3.2.2.1.3.2 The system shall verify that the input is identical to the “Password” input. If it is different, the system shall display an error message “Password does not match”.

3.4.3.2.2.1.4 Users must be redirected back to their profile pages once their usernames and/or passwords are successfully changed.

3.5 Create Assignment

3.5.1 Description and Priority

The ‘Set Assignment’ function allows teachers (admins) to set an assignment for their students. As assignments are a good indicator of how well students are following the teachers’ lessons, this function has moderately high priority.

Priority: 8/10

3.5.2 Stimulus/Response Sequences

Main Flow:

1. Teachers login to their admin accounts with valid email addresses and passwords.
2. System verifies accounts credentials and displays the main page.
3. Teachers select the option to set assignments.
4. System displays the ‘Set Assignment’ page.
5. Teachers input the number of questions and click the ‘Create’ option.
6. System display ‘Create New Question’ Page.
7. Teachers input the following:
    a. Question
    b. Options 1 to 4
    c. Answer

    and selects the ‘Next Question’ option.
8. System displays the inputs and asks for confirmation.
9. Teachers select the Create option.
10. System displays Set Assignment page.
11. Teacher enters the deadline date and time.
12. Teacher selects submit option
13. System saves the assignment in the database and sends the assignment to student accounts.

Alternative Flows:

AF-S6: Teacher leaves question, option 1, option 2 and/or answer fields empty:

1. System displays a “Question, Option1, Option2 and Answer cannot be empty” message.
2. Return to step 6 in the main flow.

AF-S6: Teacher enters any number that < 1 or > number of options for answer input:
1. System displays an “Invalid Answer” message.
2. Return to step 6 in the main flow.

AF-S5: Teacher enters leaves number of questions field empty:
1. System displays an “Number of questions cannot be empty” message.
2. Return to step 5 in the main flow.

AF-S5: Teacher enters 0 for number of questions field:
1. System displays an “Invalid number of questions” message.
2. Return to step 5 in the main flow

AF-S11: Teacher leaves deadline data and/or time field empty:
1. System displays an “Deadline cannot be empty.” message.
2. Return to step 11 in the main flow

AF-S11: Teacher enters non-existent date and/or time:
1. System displays an “Unable to parse deadline to DateTime.” message.
2. Return to step 11 in the main flow

AF-S11: Teacher enters a date and/or time that has passed:
1. System displays an “Invalid Deadline.” message.
2. Return to step 11 in the main flow

3.5.3 Functional Requirements

3.5.3.1 The system must allow all teacher accounts to create an assignment.

3.5.3.2 The assignment must have a deadline, in the format DD/MM/YYYY Hour:Minute.

3.5.3.2.1 The system must close the assignment when the deadline is reached.

3.5.3.2.1.1 If the student attempts to open the assignment after the deadline has passed, an error message “The deadline has passed. You may not attempt the assignment.” should be displayed.

3.5.3.2.1.2 If the student is in the middle of completing the assignment, the students’ current answers should be automatically saved and submitted for grading.

3.5.3.2.1.3 If the student is in the middle of completing the assignment, after the students’ current answers are automatically submitted, the student must be redirected back to the Home Page.

3.5.3.3 The teacher must be able to optionally set a starting date for the assignment, in the format DD/MM/YYYY Hours:Minutes.

3.5.3.4 The teacher must be able to set Multiple Choice Questions for the assignment.

3.5.3.4.1.1 MCQs can have a variable number of options.

3.5.3.4.1.2 MCQs must be single-input only questions (e.g. Choose only 1 option out of 4 options)

3.5.3.4.2 The assignment must be automatically graded, with each correct input giving exactly 1 mark.

3.5.3.5 The system shall only allow students to submit the assignment when all questions have been answered.

3.5.3.5.1 Students cannot leave the assignment mid-way through.

3.6 Create Challenge

3.6.1 Description and Priority

The application allows student users (players) to create their own multiple-choice quizzes to challenge their peers, if they are in the Challenger's Gym. Students can only challenge one player at a time and each challenge can only contain 3 questions.

Priority: 5/10

3.6.2 Stimulus/Response Sequences

<u>Main Flow:</u>

1. Students log in to their player account with valid email addresses and passwords.
2. System verifies accounts credentials and displays the main page.
3. Students control their characters to enter the Challenger's Gym.
4. System displays the Challenger's Gym.
5. Students interact with the Non-Playable Character (NPC) and select the option to create a challenge.
6. System displays the 'Create Challenge' page.
7. Students input the number of questions and click the 'Create' option.
8. System display 'Create New Question' Page.
9. Students input the following:
    a. Question
    b. Options 1 to 4
    c. Answer

    and selects the 'Next Question' option.
10. Students input the opponent's email and select the 'Submit' option.
11. System saves the challenge in the database and sends the challenge to the specified student (if applicable) or all students.

<u>Alternative Flows:</u>

AF-S1: If the student inputs an invalid email address:

1. System displays an "Invalid email address" message.
2. Return to step 1 in the main flow.

AF-S1: If student inputs the wrong password:

1. System displays a “Wrong password” message.
2. Return to step 1 in the main flow.

AF-S7: If the student leaves the number of questions field empty:

1. System displays an “Number of questions cannot be empty” message.
2. Return to step 7 in the main flow.

AF-S7: If student inputs any value less than ‘1’ for the number of questions:

1. System displays an “Invalid number of questions” message.
2. Return to step 7 in the main flow.

AF-S7: If student inputs any number other than ‘3’ for the number of questions:

1. System displays an “Only 3 Questions are required for challenge” message.
2. Return to step 7 in the main flow.

AF-S10: If the student leaves the opponent email field empty:

1. System displays an “Opponent Email cannot be empty” message.
2. Return to step 10 in the main flow.

3.6.3 Functional Requirements

3.6.3.1 The system shall allow any student account to create challenges in the Challenger’s Gym.

3.6.3.2 The challenges shall have a 1 week deadline from the day it is created.

3.6.3.2.1 The system shall close the challenge automatically after 1 week.

3.6.3.3 The system shall allow the student to challenge one specific player using their email.

3.6.3.3.1 Students shall come up with 3 Multiple Choice Questions (along with the answers).

3.7 Attempt Assignment

3.7.1 Description and Priority

The application allows student users (players) to attempt the assignments that their teachers have given them.

Priority: 8/10

3.7.2 Stimulus/Response Sequences

Main Flow:

1. Students login to their player accounts with valid email addresses and passwords.
2. System verifies their credentials and redirects the student to the Home Page.
3. The student selects the “Attempt Assignment” button to show a list of assignments.
4. The student inputs an assignment id.
5. The student completes the assignment and their answers are submitted for grading.
6. The system displays the student’s grades immediately if the assignment has been set to be automatically graded by the teacher.

3.7.3 Functional Requirements

3.7.3.1 Students must access their assignments through the Home Page.

3.7.3.2 Students shall only be able to attempt assignments before the deadline.

3.7.3.2.1 If the deadline has been passed, the student must see an error message, “The deadline has passed. You may not attempt the assignment.”

3.7.3.2.2 If the deadline passes while the student is attempting the assignment, the students’ current answers should be automatically saved and submitted for grading.

3.7.3.3 Students shall only be able to submit their assignments when all questions have been answered.

3.7.3.4 Students must be able to view their results immediately after the assignment has ended.

3.8 Attempt Challenge

3.8.1 Description and Priority

The application allows student users (players) to attempt the challenges set by their peers, if they are in the Challenger’s Gym.

Priority: 5/10

3.8.2 Stimulus/Response Sequences

Main Flow:

1. Students login to their player accounts with valid email addresses and passwords.
2. System verifies accounts credentials and displays the main page.
3. Students control their characters to enter the Challenger’s Gym.
4. System displays the Challenger’s Gym page.
5. Students interact with the NPC and select the option to attempt challenges.
6. System displays the ‘Challenges’ page.
7. Students answer the quiz and submit their answers.
8. System asks for confirmation of answers.
9. Students select the confirm option.
10. System displays result.

3.8.3 Functional Requirements

3.8.3.1 Students must be in the Challenger’s Gym to take part in any challenge.

3.8.3.2 Students shall only be allowed to take part in the challenge within 1 week from its creation date.

3.8.3.3 All challenges available to the student shall be listed for the student once they enter the Challenger’s Gym.

3.8.3.3.1 The student shall be able to choose to attempt any of the available challenges.

3.9 Summary Report

3.9.1 Description and Priority

The application allows teachers (admins) to view individual student's summary reports, which are based on the student's mastery of the course. Teachers can use these reports to adjust the teaching contents and key points during classroom teaching.

Priority: 9/10

3.9.2 Stimulus/Response Sequences

Main Flow:

1. Teachers login to their admin accounts with valid email addresses and passwords.
2. System verifies accounts credentials and displays the main page.
3. Teachers select the option to view students' summary reports.
4. System displays the 'View Summary Report' page.
5. Teachers enter the student's ID into the search bar and select the 'Get Report' option.
6. System displays a summary report.

Alternative Flows:

AF-S5: Teacher enters invalid student ID:

1. Nothing is displayed.
2. Return to step 5 in the main flow.

AF-S5: Teacher enters student ID with no scores:

1. System displays 'No Records'
2. Return to step 6 in the main flow

3.9.3 Functional Requirements

3.9.3.1 Teachers shall be able to view any student's summary report.

3.9.3.1.1 The summary report shall display the student's username and email.

3.9.3.1.1.1 The summary report shall display the student's complete progress for each world.

3.9.3.1.1.1.1 The summary report shall display which questions the student answered correctly.

3.9.3.1.1.1.2 The summary report shall display which questions the student answered incorrectly.

3.10 Visit World

3.10.1 Description and Priority

The application allows student users (players) to visit different worlds in the game. The application consists of a series of worlds to be explored, each representing a phase in the software development lifecycle.

Priority: 10/10

3.10.2 Stimulus/Response Sequences

Main Flow:

1. Students login to their player accounts with valid email addresses and passwords.
2. System verifies accounts credentials and displays the main page.
3. Students control their characters using the arrow or WASD keys to move horizontally or vertically to visit different worlds.
4. System displays the world their characters move into.

3.10.3 Functional Requirements

3.10.3.1 The system shall allow students to choose to spawn in any of the 6 worlds which they have unlocked.

3.10.3.2 The system shall allow students to explore the world using the arrow or WASD keys to move.

3.10.3.3 The system shall allow students to interact with objects in the world by pressing the ‘E’ key.

3.10.3.4 The system shall allow students to return to the home page by pressing the escape key.

3.11 View Leaderboard

3.11.1 Description and Priority

The application allows student users (players) to view their standings on the leaderboard. This leaderboard is based on the points they have scored from answering the MCQ questions. The students can only see their peers' standings.

Priority: 6/10

3.11.2 Stimulus/Response Sequences

Main Flow:

1. Students login to their player accounts with valid email addresses and passwords.
2. System verifies accounts credentials and displays the main page.
3. Students select the option to view the leaderboard.
4. System displays the 'View Leaderboard' page.

3.11.3 Functional Requirements

3.11.3.1 The system shall display the leaderboard for all students based on the individual total points of the selected gym.

3.11.3.2 The system shall display the leaderboard for all students based on the individual points of completed assignments.

3.11.3.3 The system shall display the leaderboard for all students based on the individual points of completed challenges.

3.11.3.4 The system shall display the entire list of students in descending order by their points.

3.11.3.4.1 Each page of the leaderboard shall have 10 students listed.

3.11.3.4.2 For each student, the system shall display their email and total points.

3.12 Gym Battle

3.12.1 Description and Priority

The application allows students to engage in gym battles in each world. These gym battles will serve as quizzes to test the students' knowledge of the software development lifecycle. The points accrued from gym battles will contribute to their standings on the leaderboard.

Priority: 10/10

3.12.2 Stimulus/Response Sequences

<u>Main Flow:</u>

1. Students login to their player accounts with valid email addresses and passwords.
2. System verifies accounts credentials and displays the main page.
3. Students control their character to enter a gym.
4. System displays the gym page.
5. Students interact with the NPCs.
6. System displays the gym battle page.
7. Students answer all questions.
8. System calculates students' marks and displays them.
9. System unlocks the next map for students.

<u>Alternative Flows:</u>

AF-S9: Student fails the gym battle:

1. System displays a 'Gym Battle Failed' message.
2. Return to step 4.

3.12.3 Functional Requirements

3.12.3.1 The system shall allow students to enter the gym of the world they are in.

3.12.3.2 The system shall allow students to challenge the gym leader.

3.12.3.3 The system shall be able to automatically and immediately mark the student's answers, and give the total score.

3.12.3.4 The system shall unlock the next map for students if they pass based on their total score.

3.13 Edit Gym Questions

3.13.1 Description and Priority

The application allows teachers to edit the questions of the gym battles. This is to ensure that students are constantly challenged and will not only receive easy questions while their knowledge increases.

Priority: 10/10

3.13.2 Stimulus/Response Sequences

Main Flow:

1. Teachers login to their admin accounts with valid email addresses and passwords.
2. System verifies accounts credentials and displays the main page.
3. Teachers select the option to edit gym questions.
4. System display “Edit Gym Questions” page.
5. Teachers select the question set to edit.
6. System displays the questions and answers for the gym battles in that world.
7. Teachers edit the question and answers and select the submit option.
8. System asks for confirmation.
9. Teachers select the option to confirm.
10. System saves the questions and answers in the database.

Alternative Flows:

AF-S7: Teacher leaves question or answer field blank:

1. System displays a “Missing fields” message.
2. Return to step 7 in the main flow.

AF-S7: Teacher provides an answer that is not an option:

1. System displays an “Invalid Answer” message.
2. Return to step 7.

3.13.3 Functional Requirements

3.13.3.1 The system shall allow teachers to edit the gym questions.

3.13.3.2 The system shall update the questions in the gym only when the teacher submits the new questions.

3.13.3.3 The system shall allow students to continue challenging the gyms while teachers are editing the questions.

3.13.3.4 The system shall state the date on which the questions for the gym were last updated.

3.14. Random Knowledge Encounter

3.14.1. Description and Priority

Players have a 10% chance of encountering random knowledge when walking in tall grass. These encounters will help them in their gym battles.

Priority: 7/10

3.14.2. Stimulus and Response Sequences

<u>Main Flow:</u>

1. Players walk into tall grass.
2. System displays random knowledge.
3. Players press ‘E’ key to close the text

3.14.3. Functional Requirements

3.14.3.1. The system shall display random knowledge with 10% probability.

3.14.3.2. The system shall display knowledge relevant to the world the player is in. (i.e. In the analysis map, the system displays information about the analysis phase).

3.14.3.3. The system shall have a list of more than 1 knowledge to display for each map.

3.14.3.4. The system shall close the knowledge text once the players press the ‘E’ key.

4. External Interface Requirements

4.1 User Interfaces

The system GUI provides menus, toolbars, buttons and panes, allowing for easy control of the system by a keyboard and a mouse for users.

4.2 Hardware Interfaces

The system supports all personal computing systems running either Microsoft Windows or Apple MacOS. Other additional hardware interfaces may include a keyboard and mouse.

4.3 Software Interfaces

The system runs on Unity version 2020.3.18f1.
The system allows teachers to link their accounts with their Twitter and Facebook accounts so that they can give students assignments through social media.
The system will use Google Firebase as its database.

4.4 Communication Interfaces

The system will use common communication resources. This includes, but is not limited to, HTTP protocol for communication with the web browser and the web server. TCP/IP network protocol with HTTP protocol. This is done for compatibility and stability purposes.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The Performance Requirements comprises a set of requirements which states how the application should perform under certain set of conditions.

Behaviour

- The application should not take more than 5 seconds to start up
- The application should be responsive to different screen size
- Should the application crash, the application should not take more than 5 seconds to reboot

Functions

- Credential identification should not take more than 2 seconds (Login)
- The application button must be responsive and the game character should move responsively and immediately according to the button pressed
- The application should not take more than 1 second to transit between scenes
- The application should not take more than 2 seconds to calculate the score for that certain player

5.2 Safety Requirements

Safety Requirements are safeguard measures put in place to reduce or eliminate possible loss, damage, or harm that could result from the use of the product.

1. Past player data like username, password, scoreboard etc. will be kept for future games / attempts
2. Database will be able to restore before game data in any case of application crashes
3. Database will be backed up every 3 months to ensure possible restoration in any case of irreversible failure

5.3 Security Requirements

Security Requirements are measures regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product.

- Password should be 12 characters minimum, inclusive of at least one capital letter, non-capital letter, digits and symbols
- Passwords will be masked
- Data will be encapsulated to prevent theft of data

5.4 Software Quality Attributes

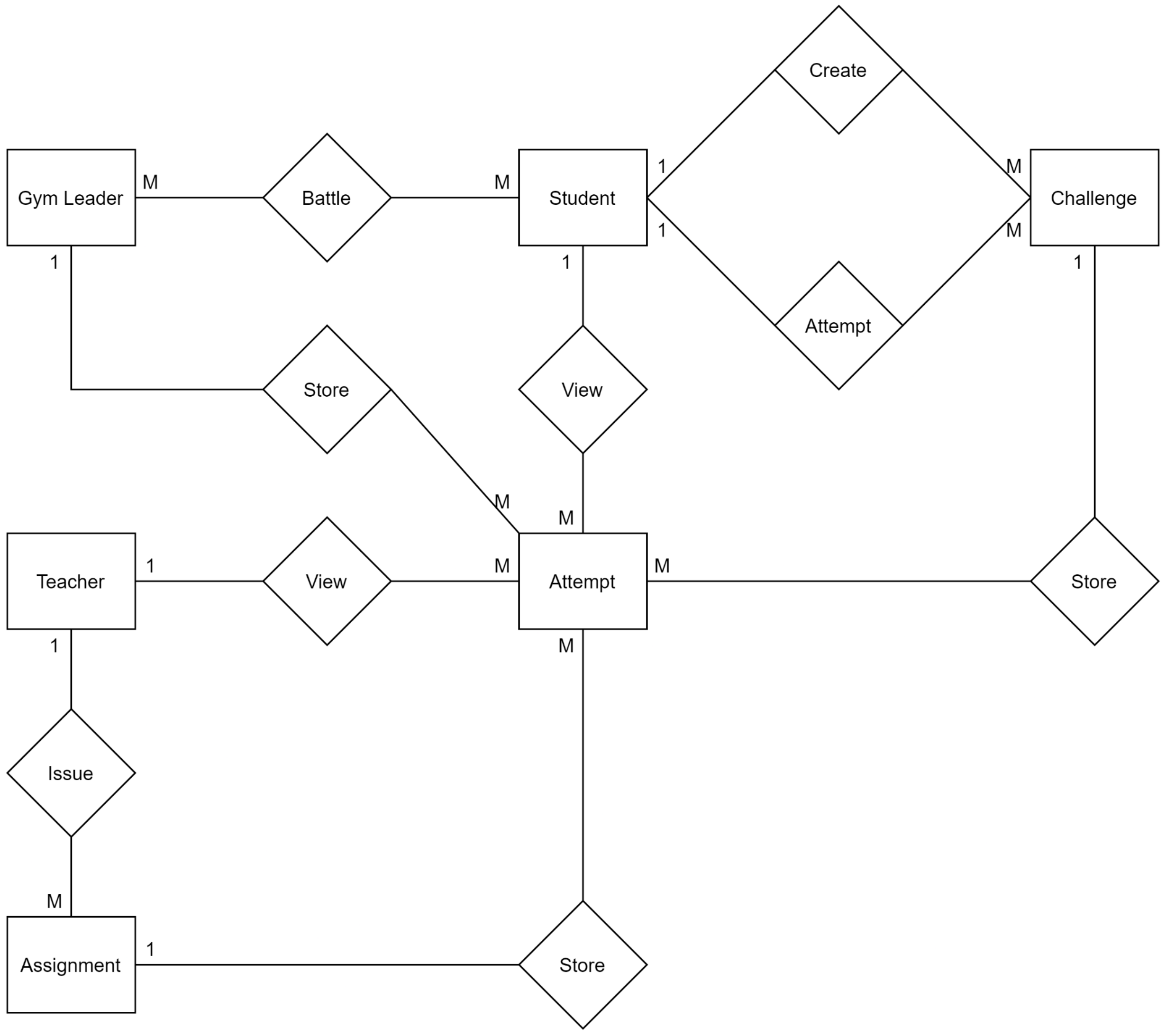
Specifications of any additional quality characteristics for the product that will be important to either the customers or the developers. We will consider only one of the quality attributes: Usability.

Usability: Game manuals will be scripted for first-time users. But in general, the interface will be design friendly and users should know of the functions behind the buttons at one glance. (E.g. arrow up / key w to move upwards, arrow down / key s to move downwards)

6. Other Requirements

6.1 Logical Database Requirement

6.1.1 ER Diagram



6.1.2 Types of information used by various functions

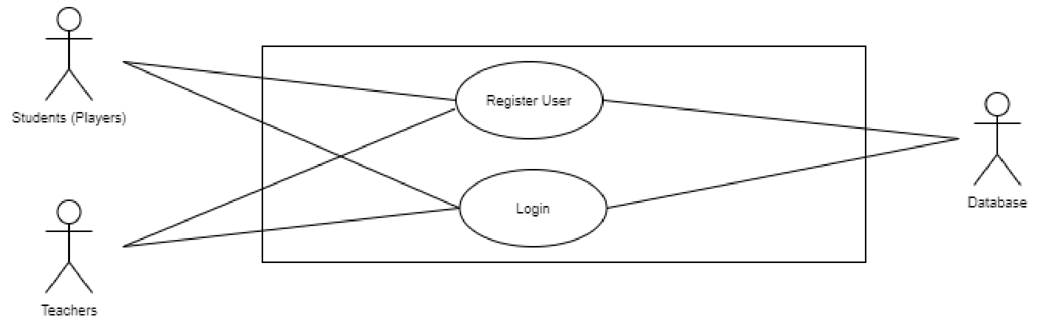
Function	Information Used
Login for student and teacher	(Student/Teacher Entity) Email and Password
Register for student	(Student Entity) StudentId, Username, Email, Password, CharacterSprite, EnrolledClass
Register for teacher	(Teacher Entity) TeacherId, Email, Password, Class
View Report Summary	(Student Entity) StudentId, Enrolled Class (Playing History) Type, ChallengeDifficulty, ChallengeScore, Assignment, AssignmentScore, SectionName, SectionScore
Create/Attempt Assignment	(Assignment Entity) AssignmentId, Question, Option1, Option2, Option3, Option4, CorrectAnswer, Score
Create/Attempt Challenge	(Challenge Entity) ChallengeId, ChallengeDifficulty, Question, Option1, Option2, Option3, Option4, CorrectAnswer, Score
Visit World	(World Entity) WorldId
Gym Battle/Edit Gym Questions	(Gym Leader Entity) GymLeaderId, GymLeaderName, Question, Option1, Option2, Option3, Option4, CorrectAnswer, Score
View Leaderboard	(Student) Username (Playing History) Type, SectionName, Score, DateRecorded

Appendix A: Glossary

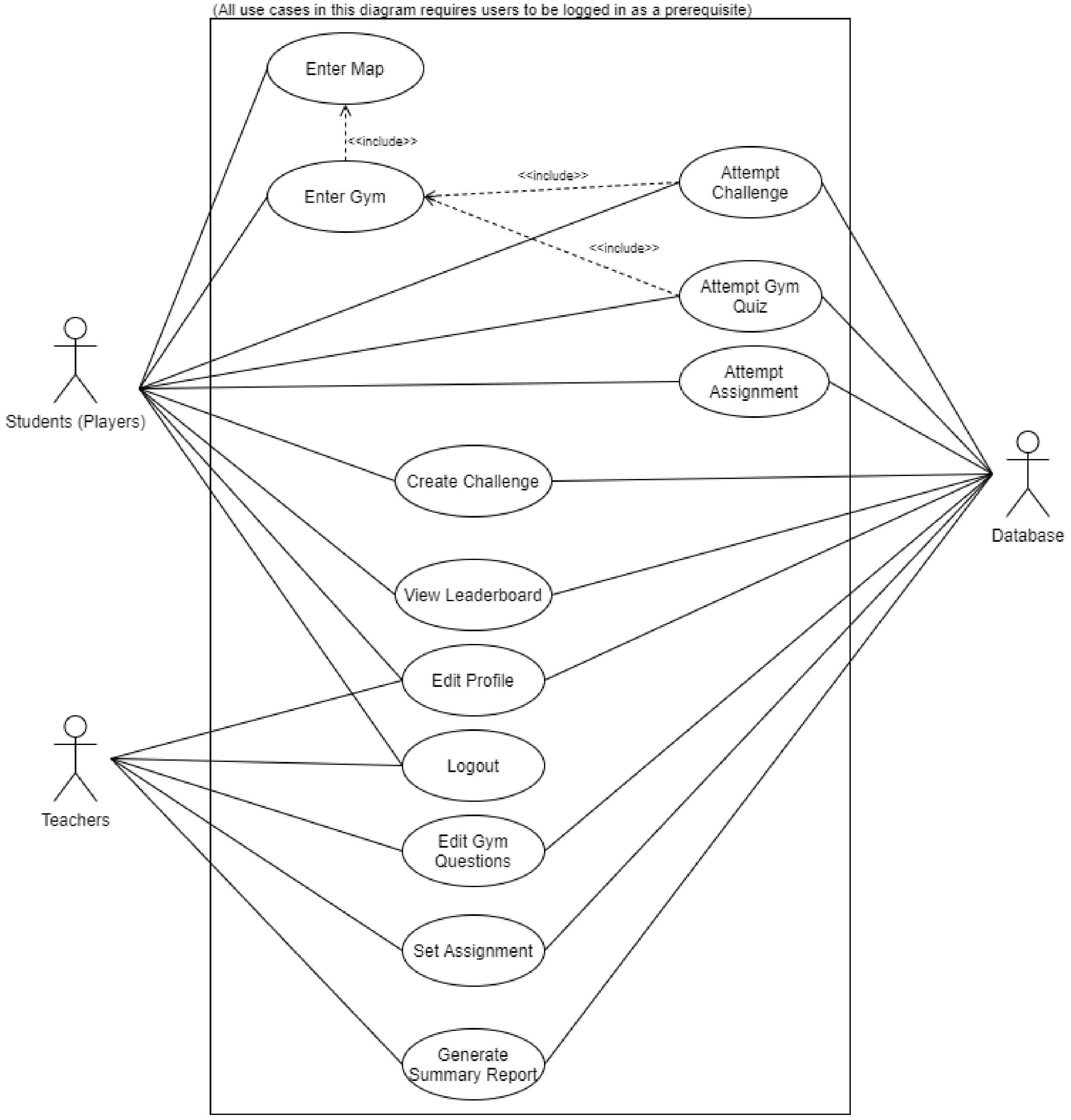
Student	The player that is playing this game.
Teacher	Teaching Assistant/Professor that is teaching a class, and is the admin in the game.
Class	A class consists of a teacher teaching the class and students enrolled in a particular class.
World	Different gyms containing different stages of the Software Development Life Cycle. (Requirements, Design, Implementation, etc) Each world contains a gym leader which students have to beat before progressing to the next world.
Gym Leader	An NPC that students challenge. The gym leader has 10 questions from multiple topics of that particular SDLC stage.
Challenges	A student-made series of questions about a particular world. Other students can attempt these challenges.
Assignment	A quiz made by a teacher.
Leaderboard	A list that shows students’ progress in the game sorted by rank. (Better score and faster progress = higher rank).

Appendix B: Analysis Model

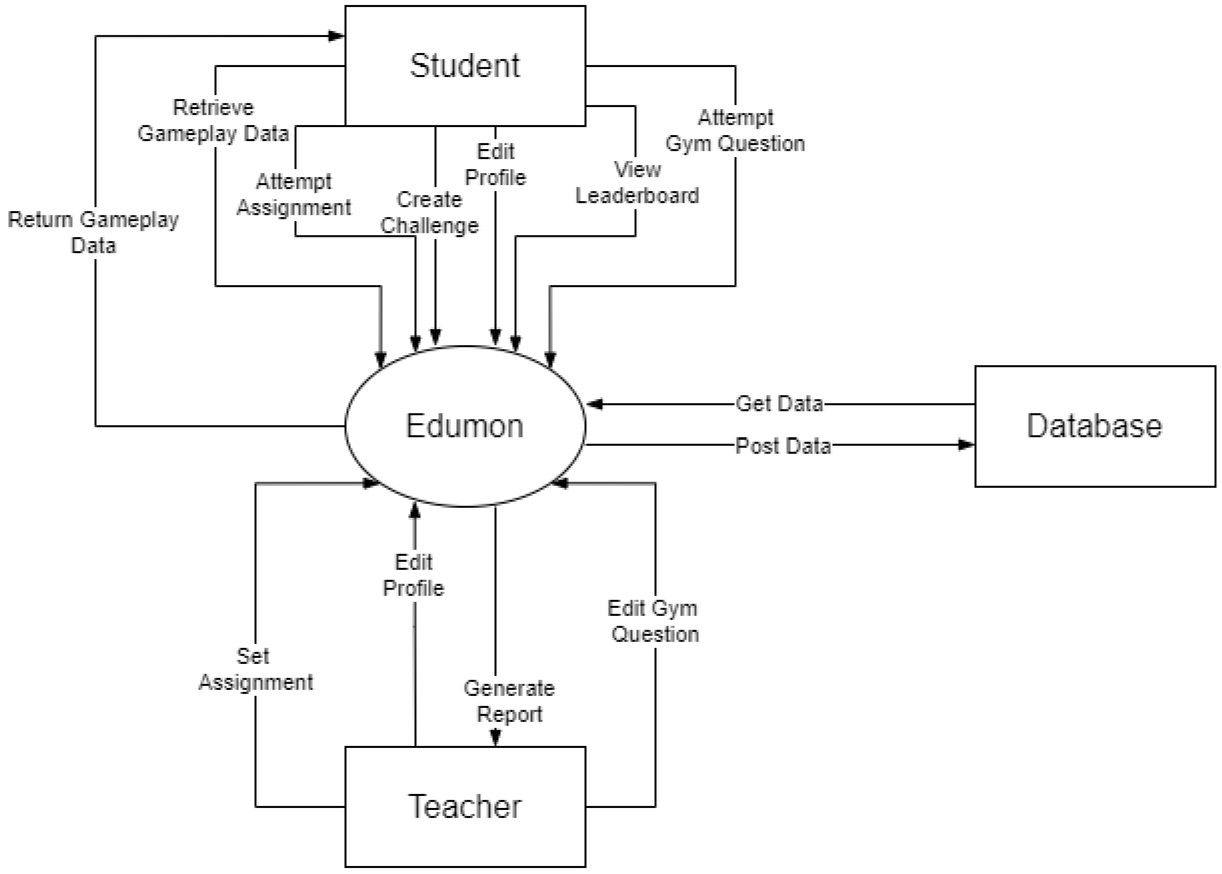
Use Case Diagram



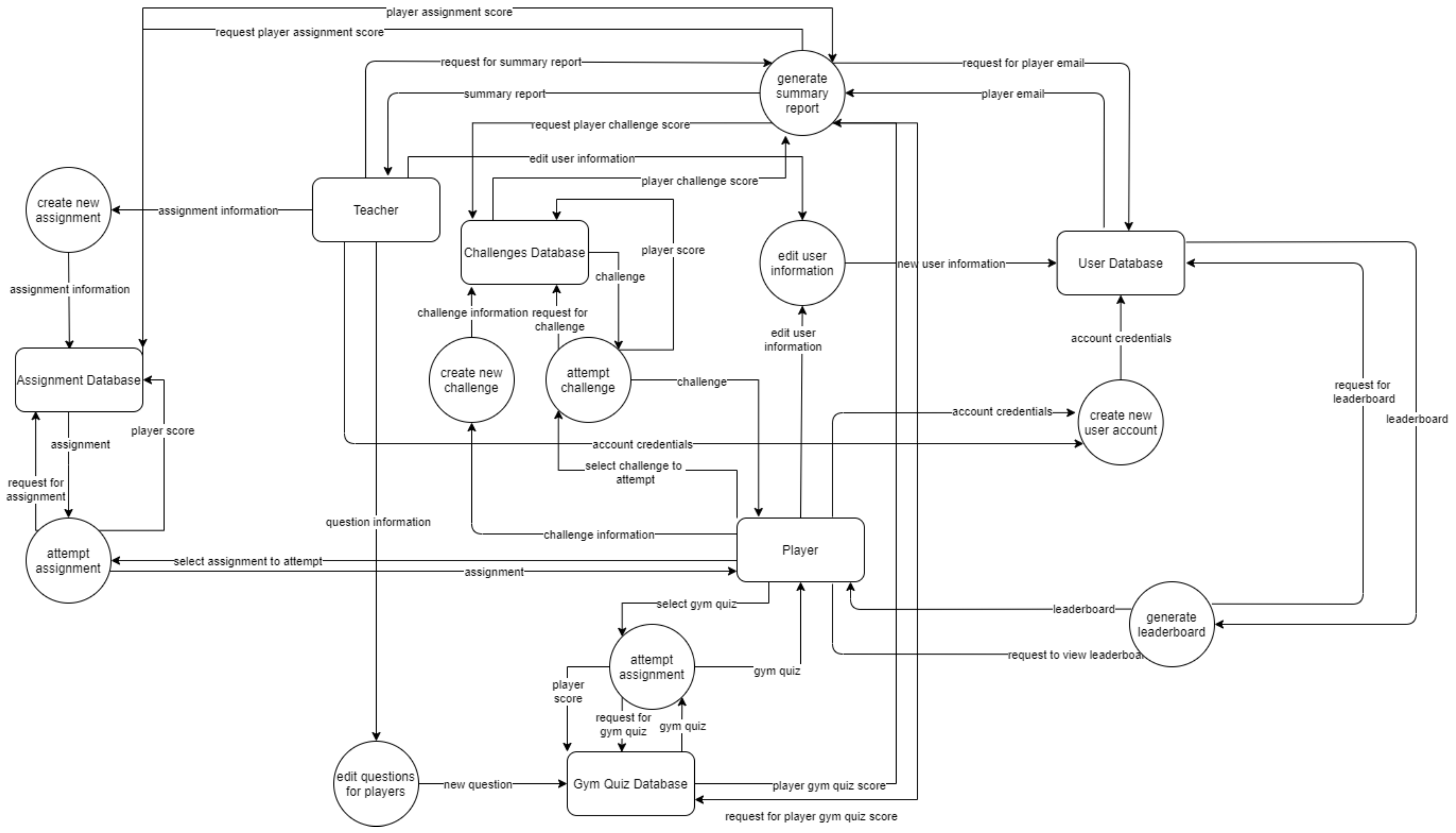
(All use cases in this diagram requires users to be logged in as a prerequisite)



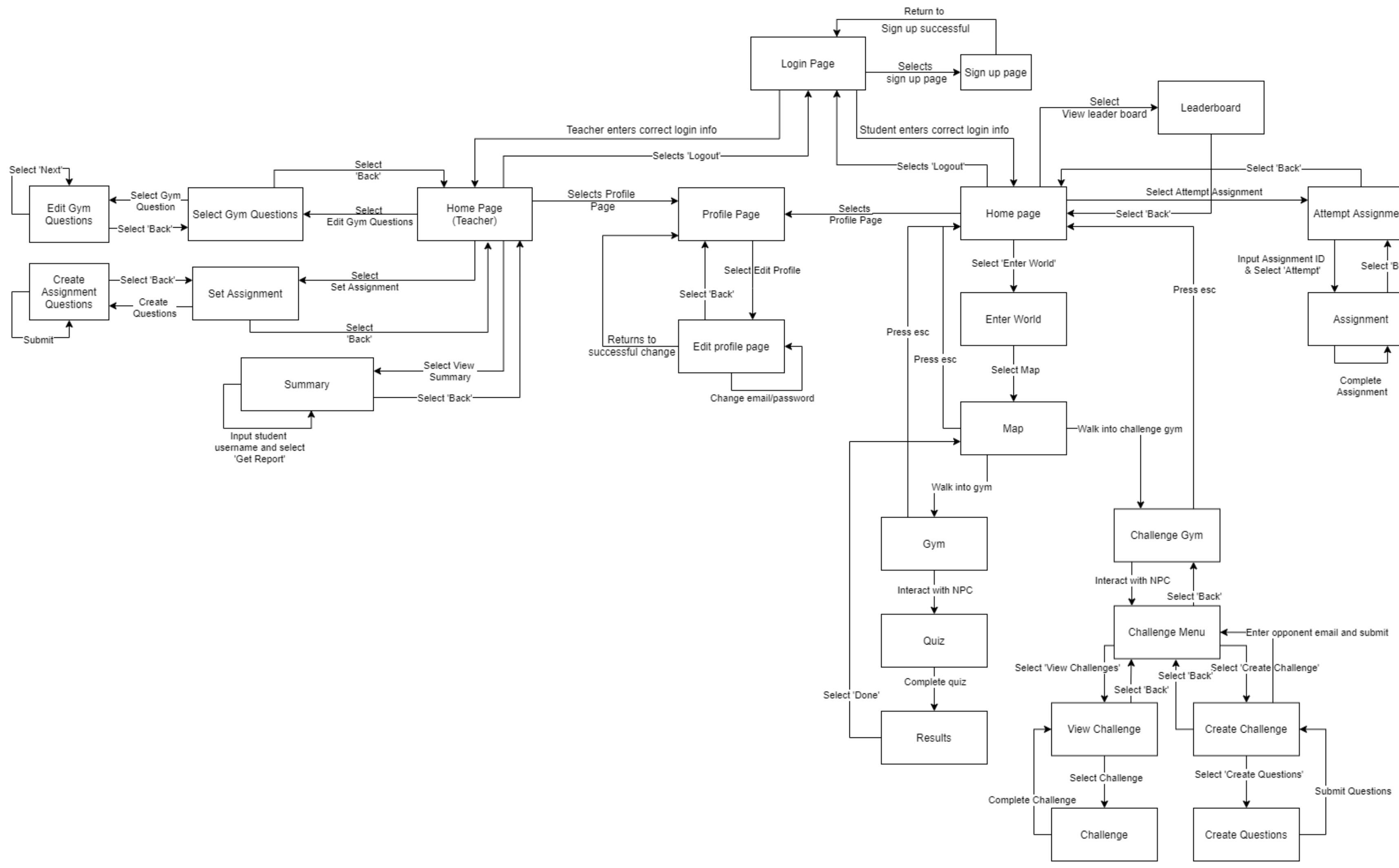
Context Diagram



Data Flow Diagram



Dialog Map Diagram


Return to
Sign up successful
Login Page
Selects
sign up page
Sign up page
Select
View leader board
Leaderboard
Teacher enters correct login info
Student enters correct login info
Selects 'Logout'
Selects 'Logout'
Select 'Back'
Select 'Next'
Edit Gym Questions
Select Gym Question
Select 'Back'
Select Gym Questions
Select 'Back'
Select Edit Gym Questions
Home Page (Teacher)
Selects Profile Page
Profile Page
Selects Profile Page
Home page
Select Attempt Assignment
Select 'Back'
Attempt Assignment
Create Assignment Questions
Select 'Back'
Create Questions
Set Assignment
Select Set Assignment
Submit
Select 'Back'
Select Edit Profile
Select 'Back'
Returns to successful change
Edit profile page
Change email/password
Select 'Enter World'
Enter World
Press esc
Press esc
Press esc
Input Assignment ID & Select 'Attempt'
Select 'Back'
Assignment
Complete Assignment
Select View Summary
Summary
Select 'Back'
Input student username and select 'Get Report'
Select Map
Map
Walk into challenge gym
Walk into gym
Gym
Interact with NPC
Quiz
Complete quiz
Select 'Done'
Results
Challenge Gym
Interact with NPC
Select 'Back'
Challenge Menu
Enter opponent email and submit
Select 'View Challenges'
Select 'Back'
Select 'Back'
Select 'Create Challenge'
View Challenge
Create Challenge
Select Challenge
Complete Challenge
Challenge
Select 'Create Questions'
Submit Questions
Create Questions