

**School of Computer Science and Engineering**

**CZ3003 Software Systems Analysis & Design**

**Lab Group: TR2**

**Team Name: Iced Milo**

**Application Name:**

**Team Members:**

**Benjamin Tan Jun Qi (U1922188H)**

**Joei Yan Zhu Yi (U1921976A)**

**Jonathan Chang Ji Ming (U1923331A)**

**Renganathan Ramasamy (U1922494L)**

**Tan Huiying Valerie (U1821571G)**

**Date submitted:**

**Revision History**

| **Name** | **Date** | **Reason for Changes** | **Version** |
| --- | --- | --- | --- |
| Iced Milo | 05/09/22 |  | 1.0 |

Table of contents

[**1.**](#_heading=h.1fob9te) **Introduction 1**

[1.1](#_heading=h.3znysh7) Purpose 1

[1.2](#_heading=h.2et92p0) Document Conventions 1

[1.3](#_heading=h.3dy6vkm) Intended Audience and Reading Suggestions 2

[1.4](#_heading=h.1t3h5sf) Product Scope 2

[1.5](#_heading=h.4d34og8) References 2

[**2.**](#_heading=h.17dp8vu) **Overall Description 3**

[2.1](#_heading=h.3rdcrjn) Product Perspective 3

[2.2](#_heading=h.26in1rg) Product Functions 3

[2.3](#_heading=h.35nkun2) User Classes and Characteristics 3

[2.4](#_heading=h.1ksv4uv) Operating Environment 4

[2.5](#_heading=h.44sinio) Design and Implementation Constraints 4

[2.6](#_heading=h.2jxsxqh) User Documentation. 4

[2.7](#_heading=h.z337ya) Assumptions and Dependencies 4

[**3.**](#_heading=h.3j2qqm3) **External Interface Requirements 4**

[3.1](#_heading=h.1y810tw) User Interfaces 5

[3.2](#_heading=h.4i7ojhp) Hardware Interfaces 9

[3.3](#_heading=h.2xcytpi) Software Interfaces 9

[3.4](#_heading=h.1ci93xb) Communications Interfaces 9

[**4.**](#_heading=h.3whwml4) **System Features 9**

[4.1](#_heading=h.2bn6wsx) Functional Requirement 10

[4.1.1](#_heading=h.qsh70q) Godot authentication 10

[4.1.2](#_heading=h.3as4poj) Creation of the World ,Section and Level 10

[4.1.3](#_heading=h.1pxezwc) Selection of World, Section 10

[4.1.4](#_heading=h.49x2ik5) Playing the Game 10

[4.1.5](#_heading=h.2p2csry) Saving Game data 10

[4.1.6](#_heading=h.147n2zr) View Leaderboard 11

[4.1.7](#_heading=h.3o7alnk) Data Analytics 11

[**5.**](#_heading=h.23ckvvd) **Other Non-functional Requirements 11**

[5.1](#_heading=h.ihv636) Flexibility 11

[5.2](#_heading=h.32hioqz) Performance Requirements 12

[**6.**](#_heading=h.1hmsyys) **Future Work 12**

# Introduction

## Purpose

This document serves to provide specification details of our application, .The purpose of this SRS is to describe the purpose and features of our game product and its expected behaviour along with any constraints faced while designing the system.

If you wish to look at how our project is implemented such as Entity Relationship Diagram , Component , Subsystem or use case model. **Please refer to our “Program document”**  instead

## Document Conventions

All requirements will be documented in detail in this document. Priority of requirements will be indicated, and the order of the requirements does not imply the priority level.

The following document convention is used in our Software Requirement Specification

* Admin  
   Refer to the Professor, lecturer or any account that is authorised to perform data analytics.
* Player  
  Refers to students or anyone playing the game without the authorization to data analytics feature.
* User  
  Refer to both Admin and Player.
* World   
  Refer to the course module.
* Section   
  Refer to the individual module chapter.
* Level   
  Refer to various Quiz question difficulties.
* HTTP  
  Refer to the Hypertext Transfer Protocol that is used by our Unity Client to make requests.
* Unity  
  Refers to our Game client engine, that is used to develop the game.
* React Application Programming Interface (API) end point   
  Refer to the API endpoint that helps support certain functions that our client could not support.
* JSON  
  Refer to the open standard file format used for data interchange.
* Game data  
  Refer to all information regarding the game example but not limited to “User score”.

## Intended Audience and Reading Suggestions

The intended audience of this document is for the Professor and Student of NTU, Developer of this system, tester, and documentation writer. This document will have information on our Project in a product perspective, its functional and non-functional requirements.

The reader can follow the document in the numbering system shown in the guide. beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.

For detail about System architecture and its subsystem implementation, refer to the Project documentation instead.

## Product Scope

This product aims to fulfil the following 5 points:

1. By gamification, it provides students with a fun and alternate way of learning. Students can learn Mathematics by playing games related to it and crossing different levels.
2. Since it’s fun, it strengthens the learning by helping students to stay focused and pay attention to the subject.
3. Allows students to review and compare their progress with their fellow schoolmates.

1. Aids professors in tracking the learning progress of the students and gives added support to students who need assistance..
2. Professors can gain from this by looking at how well their students performed and specifically where most of them struggle to understand certain concepts. With this data, professors can in turn develop a better way of teaching or provide more assistance for students in need.

## References

To help in our project development the following document is being reference.

* Unity Documentation   
  <https://docs.unity3d.com/Manual/index.html>
* React.js   
  <https://reactjs.org/docs/getting-started.html>
* Flask   
  https://flask.palletsprojects.com/en/1.1.x/
* IEEE Software Requirement Specification Template

# Overall Description

## Product Perspective

This product is a new, self-contained product designed to gamify and socialise teaching and learning of Mathematical concepts and topics. It aims at helping teachers better understand and meet the needs of the students.

## Product Functions

Our game supports authentication for both players and administrators.

Our product’s functionality can be categorised as follow(s):

Player

1. The game allows a player to answer questions on various levels set by the professors in an interactive manner.
2. The game allows players to view the leaderboard for the different worlds.
3. Worlds are divided into levels, which consists of multiple choice questions that the players must answer correctly to advance to the next level
4. The player would have to fight a “World Boss” as the final level of every world that will be a summary of the concepts learnt in that world

Admin

1. The game allows the administrator to analyse a player’s scoreboard historical records.
2. Administrators can add, remove or modify the content of the questions as they deem suitable.
3. Administrators are allowed to add or remove users.

## User Classes and Characteristics

The product will have two main user classes:

1. Players - Students that are playing the game without the authorization level of an Administrator.
2. Administrator – Professor or teaching staff that has access to the administrative functions of the game.

## Operating Environment

The minimum specifications needed to run the program include:

1. Mac 10+, Windows 7 or higher.
2. Internet connectivity.

## Design and Implementation Constraints

The product is designed with the following constraints:

1. Inexperience in game development
2. Game is only tested on the latest version of Windows 10.
3. Game development is limited to 8 weeks.
4. Lack of Mono/C# scripts supports deploying to WebGL in Godot.
5. Poor performance of the Light2D node in Godot.
6. Insufficient Godot documentations in Window PC social media sharing and connection to database
7. Lack of comprehensive tutorials/ resources on building a full game with Godot that meet all our functional requirement.
8. Provided with one centralized server.

## User Documentation.

A youtube demonstration video will be provided for new users to understand the functionality of the application.

## Assumptions and Dependencies

The following will be assumed for the development and usage of the system:

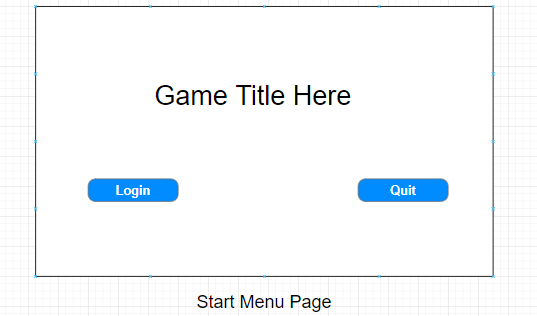
1. Users are running on a supported version of Windows 10.
2. Users of the system will have a stable internet connection of minimum 100mbps upload/download speed.
3. Users have a modern web browser pre-installed.
4. Users will know how to navigate through the game based on the user interface.
5. The system will not be able to use the students’ NTU login credentials for account creation and validation.
6. Users are well-versed in the common functions of web applications. (eg. sharing to social media)

# External Interface Requirements

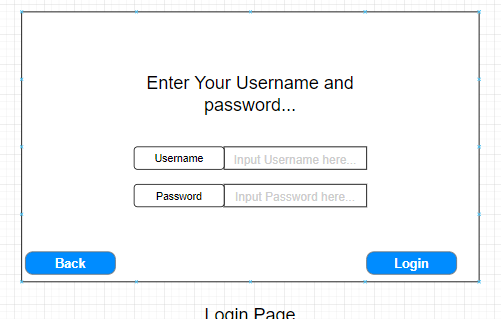
This section defines and describes the interface requirements for the system. This includes requirements for user interfaces, hardware interfaces, software interfaces and communication interfaces.

## User Interfaces

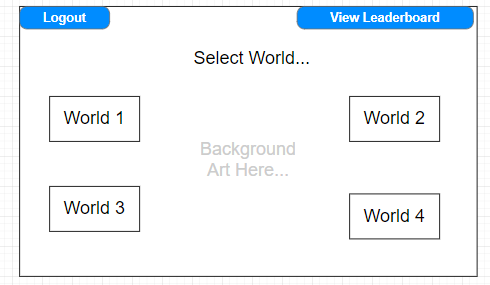
All user interactions are done through a Graphical User Interface (GUI). The following illustrations depict the relevant user interfaces and describe the main logic behind each view.



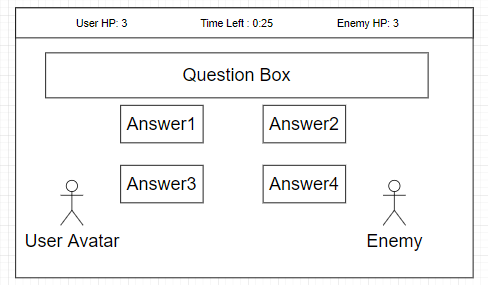
When the user starts the game, the start menu scene as shown above will be displayed.



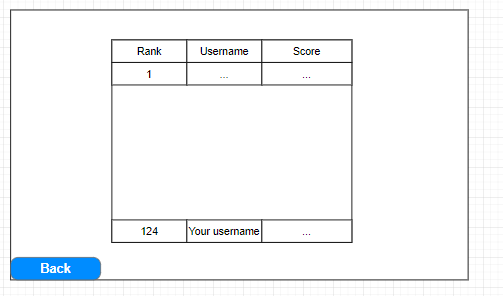
When the user clicks on the login button on the start menu scene, the login scene will be displayed as shown above. Users must input their username and password to autheticate. Depending on whether the entered username and password correspond to an admin on a student, the system will display either the world select scene (for students) or the admin view scene (for admins).



The World Select Scene is shown as above. Clicking on the top-right button will display the score leaderboard. Clicking on the top-left button will log-out the user and return to the start menu scene. Clicking on the world buttons will display the gameplay scene.



The gameplay scene is shown above. The student will answer the questions in a quiz format in a limited time.



Above shown is the leaderboard display. The users will be ranked according to their descending score, with the current student’s rank and score being shown at the bottom of the leaderboard.

## Hardware Interfaces

Users will require the following hardware to interact with the software:

1. Mouse
   1. Main form of navigation for the application
2. Keyboard
   1. To input data for user registration, logging in and gameplay
3. Network – Wired or Wireless
4. Operating System – MacOS, Unix, Windows 10
5. 1GB of RAM or higher
   1. To run the application with less than 0.1seconds of delay.

## Software Interfaces

The following is a list of softwares that will be used for the development of the application

1. Unity 3D 2017
2. NodeJS
3. Firebase Database
4. Facebook and Twitter API

## Communications Interfaces

1. The Application will communicate with the database through the use of Hypertext Transfer Protocol (HTTP) by calling a Service (containing REST-API endpoints).  
   1. API endpoints accept the response in a JSON format with the parameters as key-value pairs or a String format for single arguments.
   2. API endpoints will always return the response in JSON format.
2. The Application will communicate with social media platforms through the use of the respective platform’s API, adhering to the platform’s API documentation.
3. The use of Database Connectivity APIs for the Service to perform CRUD operations on the database.

# System Features

Our system features are as follows

1. Players can increase their Mathematical prowess through our game.
2. Users can share their leaderboard position on social media.
3. Users can design and introduce their own levels into the game.

## Functional Requirement

### User authentication

1. Upon starting the software, the user shall be able to input two fields, henceforth referred to as the username and password, and shall be able to click on a button to authenticate the user using the username and password.  
   1. Upon authentication, the system shall query an external interface on whether the username and password are correct, as in, a matching key-value pair of username and password are found in the corresponding database.
   2. If the username or password is either blank or incorrect, authentication is considered unsuccessful, and the program will display a prompt with the text “Please enter valid username and password”.
   3. If the authentication is successful, the system must check whether the user is an admin or a student.
      1. If the user is a student, the software program will display the World Selection selection scene.
      2. If the user is an admin, the software program will display the Administrative scene.

### Adding/Removing Users

1. Admins must be able to add users and their credentials to the game.
   1. When the admin requests to add a user, the admin must input the new user’s username and password.
      1. If the username or password is blank, the system will refuse the user add request and display ‘Please fill in the username/password.’
      2. If the username input is already being used by another user, the system will refuse the user add request and display ‘Please choose another username’
      3. Else, the system will allow the request and add the user credentials to the database.
2. Admins must be able to remove users and the credentials from the game database.
   1. When the admin requests to remove a user, the system will display a list of usernames.
      1. The admin must select and confirm the corresponding username of the user to be removed. The system will then remove the user and their credentials from the user database.
      2. **Creation/Deletion of the Question**
3. Users must be able to create questions with varying difficulties.  
   1. The fields to be entered are “Question Number, “Description”, “Four Options” and “Explanation”.
   2. Upon successful creation, there would be an option provided to share the students’ results to social media.
      1. Section to be shared to “Facebook” via Facebook API.
      2. Section to be shared to “Twitter” via Twitter API.
4. Users must be able to delete questions.
   1. When a user requests to delete a question, the system will display a list of questions that can be removed at the time.
      1. If the user is a student, the system will only display their own created questions in their worlds.
      2. If the user is an admin, the system will display all questions, including student-created questions and admin-created questions.
      3. The user must select and confirm the corresponding question to be removed. The question will then be removed from the database.

### Selection of World and Section

1. Players are able to select the world that they wish to play in with the use of their mouse.  
   1. Upon selection, players would then be able to press on the “Play button” to commence with the game.

### Playing the Game

1. During gameplay, the system will display a question to the user, along with several answers.
   1. Each question will only have one ‘correct’ answer displayed, along with several ‘incorrect’ answers. The user must be able to click on one of these answers to answer the question.
      1. On selecting a correct answer, the corresponding question will be considered to have been answered correctly.
      2. On selecting an incorrect answer, the corresponding question will be considered to have been answered incorrectly.
   2. For each question, a timer will be displayed. Once the timer reaches zero, the corresponding question would be considered to have been answered incorrectly.
   3. For the whole gameplay session, the system will display the number of tries the user has left, which must be set at the beginning of the gameplay session.
   4. Once a question has been answered:
      1. If the question is answered, and no more questions are left, end the game.
      2. If a question has been answered and more questions are left:
         1. If the question has been answered correctly, move on to the next question.
         2. If the question has been answered incorrectly, reduce the number of tries the user has left. If the number of tries reaches zero, end the game. Else, move on to the next question.
2. When the game has ended, the system will update and upload the user’s score to the database.

### View Leaderboard

1. Players will be able to view the leaderboard for that section.
2. Users would be able to share their leaderboard positions to social media platforms (i.e. Facebook and Twitter).

# Other Non-functional Requirements

## Performance

Our game would need to meet the following performance requirements.

1. The system must support a simultaneous gameplay of a minimum of 2 players.
2. The response time for displaying a question must be less than 1 min.
3. Players’ answers must be submitted and a result must be generated within 2 mins.
4. The system must support simultaneous uploading of results to social media.
5. In the case where the system is rebooted, the application should launch within 30 seconds.
6. The system must not crash when an administrator edits a question while a player is using the application.

# Future Work

Due to certain constraints, some non-critical functions were left out intentionally.

1. Picking of Character
2. World Creation
3. Student Analytics such as their own History
4. Creation of account.