NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Drawdiculous

Software Requirement Specifications (SRS)

Version 1.0

Prepared by Team Luck

Bian Hengwei	U1923732B
He Yinan	U1922693C
Jin Han	U1922733A
Lee Yu Jie Melvin	U1922533G
Renganathan Ramasamy	U1922494L
Sannabhadti Shipra Deepak	U1822459L

Table of Contents

1. Problem Statement

As people age, they are more likely to suffer from age-related diseases such as dementia. Currently, there are around 50 million people worldwide suffering from dementia. Due to the aging population, this number is expected to increase to 82 million by 2030. Dementia typically affects the elderly over 65 years old, however, research shows that dementia is now on the rise among younger people. In Singapore, the National Neuroscience Institute diagnosed four times as many people under 65 with dementia in 2017 than in 2011. Research shows that cognitive training can remarkably slow down the cognitive decline associated with aging.

2. Overview

2.1. Background

As technology develops, more and more elderly are owning and using mobile phones. Mobile games are the most accessible games now as it requires nothing other than a functioning mobile phone. There exist many games that target people with dementia, but these games often involve in-person interaction and are not playable if players do not meet physically. These games are less applicable now with the Covid-19 pandemic that discourages face-to-face interaction. There also exist many mobile games that are generally friendly to the elderly, however, the majority of these games target children, which makes the design less appealing to the elderly.

Considering the current situation of the aging population and the Covid-19 pandemic, a mobile game that is accessible, elderly-friendly, and stimulates thinking is much needed to alleviate the problem the world is facing now.

2.2. Overall Description

The mobile game will be developed using Android Studio for the frontend and Java for the backend. The mobile game targets the elderly by providing an elderly-friendly user interface and stimulates thinking while being interesting.

The mobile game will allow players to form groups and play draw-and-guess games. One user in the group will be provided with a prompt to draw and the other players will attempt to guess the word that the drawer is trying to draw.

3. Investigation & Analysis Methodology

3.1. System Investigation

Drawdiculous pushes any local changes, including account changes, game room status, drawing status, and guessing status to the server. The server listening to port 3002 will receive all updates from the user and handle the updates accordingly. Some crucial data, such as game room structures and drawing status, will be processed in an active object in the program. Upon user requests, information is pulled by the users and the UI is subsequently updated. Specifically, all data transmissions will be in a JSON format to help simplify the program.

3.2. Analysis Methodology

3.2.1. Feasibility study and requirements elicitation

Surveys on potential users about opinions on Draw and Guess type games will be conducted. Small-scale testing of the working protocol on these users will be done before the application is released. Regular meetings with sponsors and all team members will be held regularly at a fortnightly frequency. Small meetings with most team members present will be held while needed to discuss the project schedule, requirement details, and other questions.

3.2.2. System analysis and requirements specification

3.2.2.1 Perform an analysis of the problem using object-oriented techniques

An external view of the product including game rooms, g drawing games, activities such as guessing answers and checking leaderboard will be developed using Unified Modeling Language (UML). This System Requirement Specifications document is a part of the second round deliverable of the project. Our system will contain some features including:

- User authentication
- Create a game room
- Join an existing game room
- Draw on the canvas
- View others' drawing real-time
- Make guesses
- Check real-time leaderboard

3.2.2.2 Scope and Limitations

The analysis will include business analysis, functional and nonfunctional requirement analysis, architecture analysis, process analysis, and code analysis.

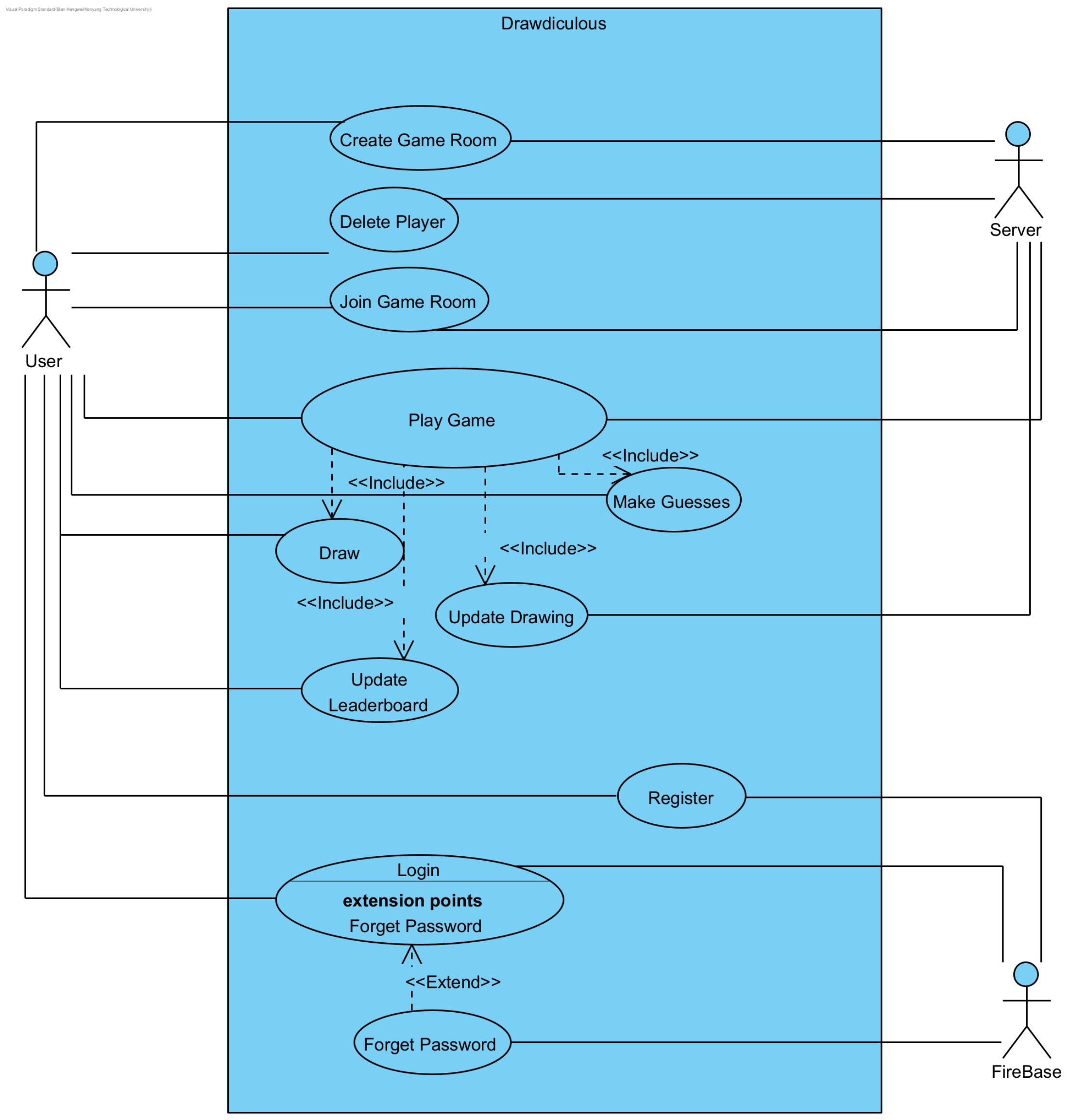
- Business analysis: business rules, business partners, sponsorship, budget management
- Requirement analysis: functional and non-functional requirements, sponsors' requirements
- Architecture analysis: app layered architecture, server implementation
- Process analysis: data flow analysis, client-server connection analysis
- Code analysis: code style, code length

There is an unavoidable time limitation on our project. Though analysis is done with care, the time limitation decides that we are not able to extend more.

3.2.3. Object-oriented design using UML

An android application designed based on Object-Oriented Design principles and MVC patterns will be developed. The design will be detailedly documented using UML diagrams.

3.2.3.1 Use Case



3.2.4. Prototyping

We will be using the Object-Oriented Rapid Prototyping (OORP) method to implement a basic working prototype of the product. The prototype will be used for demonstration purposes by our sponsors. The prototype will include a basic UI of the application and simplified communication protocols between the client and the server.

4. Constraints

4.1. Scalability

According to our current design, too many connections to the server would overload the server and result in either connection error or very slow response speed. Our product is not scalable due to server constraints. In order to make the product more scalable, a distributed game server system would be needed instead of a single server. The server program needs to be improved as well.

4.2. Platform

We are developing the application using Android Studio and the product would only be available on the Android platform. An Android device is required to run the application. To make the application more portable, we need to rewrite the program in Swift / C#, etc.

4.3. Project Schedule

We have a three-month timeframe to implement the Drawdiculous application. This project should begin from the Lab 1 session of CZ3002 and finish by Lab 5.

5. Operational Requirements

5.1. System User Support

We will be providing a detailed user manual for users. The user manual will include a detailed description of how to create and log in to an account, how to manage a room, how to play games, etc. There will also be a list of Frequently Asked Questions (FAQ) to address and provide answers for common questions or concerns.
In case the user is unable to find the solution to their problem in the FAQ, the "Help" section will also include a telephone and email address where users can call or email for further support. Users will have 24x7 access to email and telephone assistance for any technical questions that may arise, such as trouble resetting their password, application errors, and so on.

5.2. Application Services and Technical Support

Programmers and application developers will have access to source code to address bugs or system enhancements as deemed necessary. Any bugs that are discovered will be recorded and a ticket will be created for the bug fix.
Network Administrator and DBA support are also required to maintain a 24x7 system uptime.

5.3. System Hardware and Data

A computer operations center will be created. It will handle system hardware tasks such as carrying out scheduled back-ups to prevent data loss as well as system patches when a bug is discovered and fixed. Every month, hardware maintenance and scheduled system maintenance must be carried out to ensure quality remains up to standard. A failover must also be set up in anticipation of system shut-downs or servicings.

5.4. Audit Trail

System audit trails are an inherent part of Drawdiculous. Among others, all transaction records will capture what action was taken, when (time-stamp) the transaction occurred and who made the transaction. The audit trail must also be verified on a monthly basis.

6. Functional Requirements

Drawdiculous is a real-time, multiplayer mobile game that caters to the social and cognitive needs of the elderly.

6.1. Registration

6.1.1 New user must be able to create a new account using the following information:

6.1.1.1 Username

6.1.1.2 Email address

6.1.1.3 Password

6.1.2 System must check that all the fields are valid

6.2. Login

6.2.1 User must enter their email address and password

6.2.2 System must check that the credentials are valid

6.2.3 System must log the user into their account and redirect them to the main page

6.3. Forget Password

6.3.1 User must enter their email address

6.3.2 Firebase must check the validity of the email

6.3.3 Firebase must send a link to change the password to the email

6.3.4 User must enter a new password

6.4. Play Game

6.4.1 System must assign the first player as drawer

6.4.1.1 System must give drawer a prompt

6.4.1.2 System must send the drawing in realtime to the guessers

6.4.2 System must assign all other players as guessers

6.4.2.1 User must be able to guess the drawer's picture and input their answer

6.4.3 System must be able to assign points to each player

6.4.3.1 For drawers, the system must be able to assign points based on how many guessers got the correct answer

6.4.3.2 For guessers, the system must be able to assign points based on how fast they entered the correct answer

6.4.4 System must assign next player in line as drawer, and others as guessers when each round ends

6.4.4.1 System must assign in a sequential manner until all players have been drawer once

6.4.4.2 System must end the game and declare the winner when all players have finished being the drawer once

6.4.5 System must be able to rank each player according to their cumulative points

6.4.5.1 User must be able to view this ranking by clicking on “View Leaderboard”

6.4.5.2 System must be able to declare the player with the most points as the winner

6.5. Join Game Room

6.5.1 User must choose “Join Room” or “Join Random Room”

6.5.1.1 To “Join Room”, users must enter the id of the room they wish to join

6.5.1.2 To join a private room, the user must enter the password as well

6.5.1.3 System must validate the room id (and password, if applicable)

6.5.2 System must redirect users into their chosen game room

6.6. Create Game Room

6.6.1 User must choose “Create Room”

6.6.2 System must generate unique room id

6.6.2.1 User must enter a password to create a private room.

6.6.2.2 User must keep the password field empty to create a public room

6.6.3 System must redirect users into their created game room

6.7. Remove Player

6.7.1 User must click on the “X” button beside the player

6.7.1.1 User must be the host of the game room

6.7.2 System must remove the player from the game room

6.7.3 System must direct removed player to the main page

7. Input Requirements

7.1. Room Type

Players should be able to create a room mode (i.e. public or private). The system will generate a room ID upon creation.

7.2. Room ID

Players who choose to join an existing room should know the ID of the room and password in the corresponding ID to play with friends. For those private rooms, only guests of the creator should be allowed to enter the room.

7.3. Game Input

During the games, players will play 2 different roles. For drawers, the players will need to draw the given words on the canvas. The system should be able to recognize, save and transmit that information. The input bar should be disabled.
During the guessing phrase, all non-drawers should be able to guess the drawing through the input bar. However, the drawing interface should be disabled at all times for non-drawers.

8. Process Requirements

8.1. Data Confidentiality

To protect personal data or information, all data transmitted to and fro the database should be encrypted.

8.2. Data Integrity

Data error from the user's end to the data-based end must be gracefully handled. Timeout transactions such as during login should be rolled back to prevent cheating (such as when playing). During processing, the data should not be changed to another value which may alter the result of the information.

8.3. Data Availability

The server must be available on a 24x7 basis. The performance should be there to prevent any in-game lagging when playing.

9. Output Requirements

9.1. Drawing

All drawings that the drawer draws on his screen should be displayed in real-time on the other player's screen. The canvas size and drawing should be displayed on the different players' phones according to the player's phone screen size.

9.2. Leaderboard Display

The game must be able to extract all the points data to display the points table of the players in the leaderboard section of the game after each game ends.

9.3. Correct guess output

After each correct guess is made, the system must check with the actual word in the database and display the correct word on the screen (if the correct word is guessed already). Letter case should be ignored when validating the correctness of a player's guess.

10. Hardware Requirements

10.1. Client Devices

Android Mobile Phones which can support at least Android 10

10.2. Developer Devices

Mac or Windows laptops running Android Studio

10.3. System for Team to communicate

Computer systems with Internet access to connect through a cloud server such as Google Drive.

11. Software Requirements

11.1. Client Operating Systems

1. Android OS 10

11.2. Client Applications

1. Android OS 10

11.3. Network System

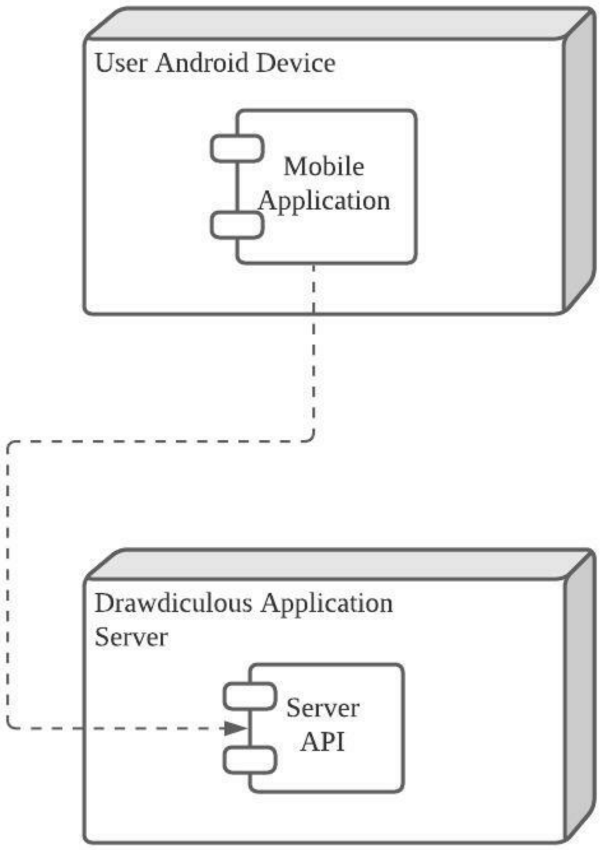
1. Network software and protocols for system communication: TCP/IP, HTTP, HTTPS, SSH, etc

11.4. Licenses

Valid licenses are required to run software from third-party vendors:

1. To use application development tools
2. To use application server and database software in development, testing, and production

12. Deployment Requirements


User Android Device
Mobile Application
Drawdiculous Application Server
Server API