

# Spell Slinger

Design Document  
4th Year Project

Student:	David Darigan
Student No:	C00263218
Supervisor:	Dr. Joseph Kehoe
Submission Date:	08/12/2023

# Table of Contents

<b>Table of Contents</b> .....	<b>1</b>
<b>Table of Figures</b> .....	<b>3</b>
<b>Introduction</b> .....	<b>4</b>
<b>System Architecture</b> .....	<b>5</b>
<b>Tools &amp; Technologies</b> .....	<b>6</b>
Kotlin.....	6
Jetpack Compose.....	7
Firebase.....	8
AR Core.....	9
Google Maps.....	10
<b>Design Patterns</b> .....	<b>11</b>
<b>Sequence Diagrams</b> .....	<b>12</b>
Register.....	12
Login.....	13
Logout.....	14
Delete Account.....	15
Train.....	16
Scan.....	17
Draw Spell.....	18
Equip Spell.....	19
Increase Stat.....	20
Battle Player.....	21
Battle Creature.....	22
View Leaderboard.....	23
<b>Class Diagram</b> .....	<b>24</b>
<b>Database Layout</b> .....	<b>25</b>
<b>Prototype Screens</b> .....	<b>28</b>
Register Screen.....	28
Login Screen.....	29
World Screen.....	30
Training Screen.....	31
Spell Book Screen.....	32
Leaderboard Screen.....	33
Settings Screen.....	34
Delete Account Screen.....	35
Spell Well Screen.....	36
Battle Screen.....	37
Battle Invite Screen.....	38

<b>Milestones.....</b>	<b>39</b>
<b>Attributions.....</b>	<b>40</b>
<b>Conclusion.....</b>	<b>41</b>

## Table of Figures

Figure 1-1 System Architecture Diagram	4
Figure 2-1 Sequence Diagram - Register	11
Figure 2-2 Sequence Diagram - Login	12
Figure 2-3 Sequence Diagram - Logout	13
Figure 2-4 Sequence Diagram - Delete Account	14
Figure 2-5 Sequence Diagram - Train	15
Figure 2-6 Sequence Diagram - Scan	16
Figure 2-7 Sequence Diagram - Draw	17
Figure 2-7 Sequence Diagram - Equip Spell	18
Figure 2-8 Sequence Diagram - Increase Stat	19
Figure 2-9 Sequence Diagram - Battle Player	20
Figure 2-10 Sequence Diagram - Battle Creature	21
Figure 2-11 Sequence Diagram - View Leaderboard	22
Figure 3-1 Class Diagram	23
Figure 4-1 Database Layout - User	24
Figure 4-2 Database Layout - Creature	25
Figure 4-3 Database Layout - Well	26
Figure 5 - Milestones	28

# Introduction

This design document's purpose is to detail features of the Spell Slinger application. This document will describe the architecture of the system, as well as the design patterns implemented alongside it. Sequence diagrams are included in order to illustrate the flow of events and data in use cases. Game objects will be shown in detail through a class diagram, and how their data is stored in the database layout. A list of prototype screen layouts will display the general idea of how the Spell Slinger application should look.

# System Architecture

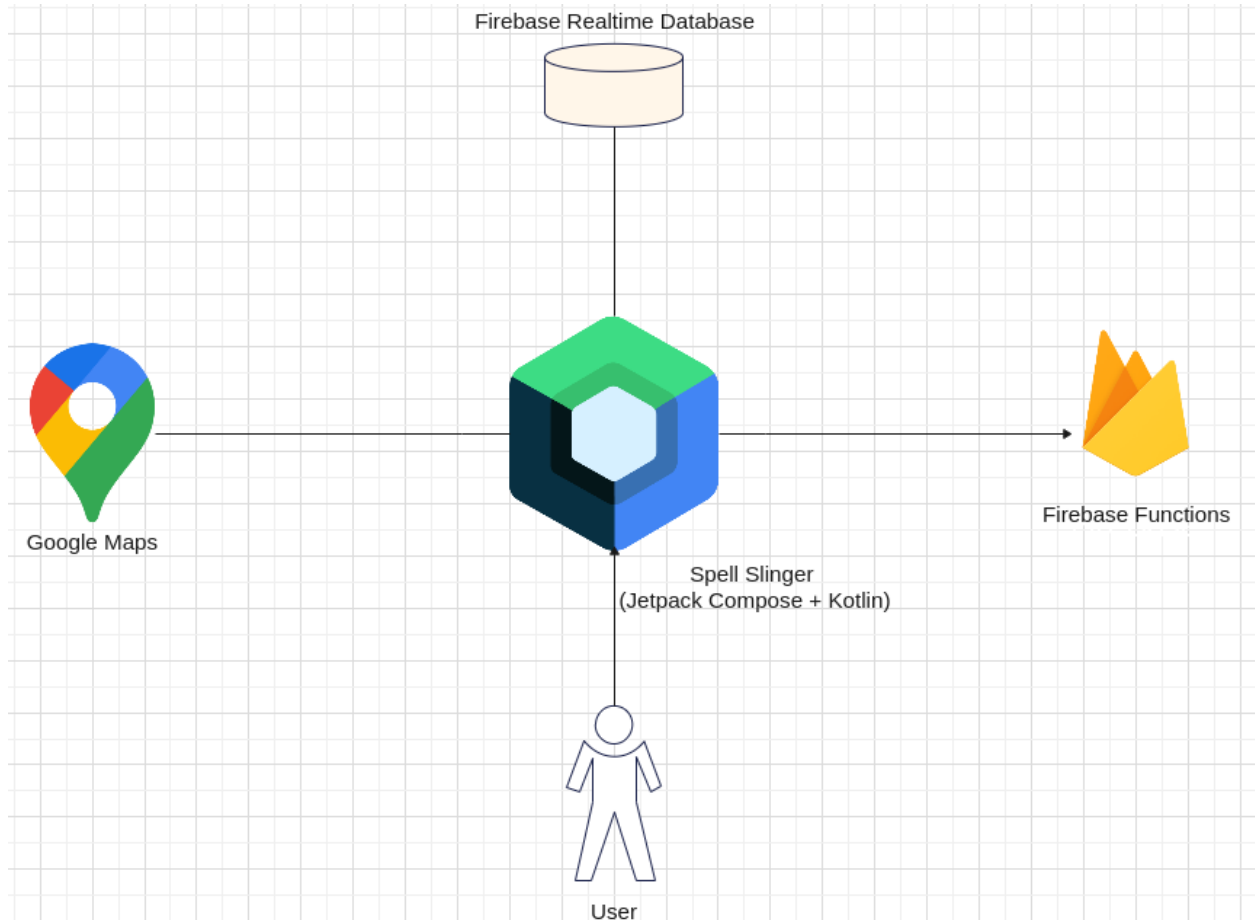


Figure 1-1 System Architecture Diagram

Spell Slinger will target Android and be written in Kotlin. The front-end will be developed in Jetpack Compose. Firebase realtime database will store account and game data. The serverless Firebase Functions will perform any complex code that the security rules for the Firebase real time database isn't suitable for. AR Core will be implemented in order to enrich player's experience, and Google Maps for Compose will be utilized in order to create Spell Slinger's World Map.

## Tools & Technologies

### Kotlin



“Kotlin is an open-source statically typed programming language that targets ..Android.. It's developed by [JetBrains](https://jetbrains.com).”

(<https://kotlinlang.org/docs/faq.html>)

#### Kotlin Syntax Example

```
// Variable declaration
// Types are statically inferred where possible
var id = 0;
// Immutable Values
val december = 12;

// Functions
fun greet(person: String) {
    // String interpolation
    println("Hello ${person}")
}

// Classes use primary constructors in their definition
class Person(val firstName: String, val surName: String, var birthYear: Int) {
    val fullName: String
        get() { return "$firstName $surName" }

    var isOver18: Boolean = false
        set(value) { field = value }

    init {
        // Init are functions that run after the constructor to validate date
        if(birthYear < 2005) { isOver18 = true }
    }
}

fun main() {
    var person = Person("David", "Darigan", 1991) // Instances are created by function invocation
    println("${person.fullName} is over 18: ${person.isOver18}")
}
```

## Jetpack Compose



Jetpack Compose is a declarative functional framework written in Kotlin developed by Google. It is growing in popularity and looks to be replacing the old Java Views of Android Development.

```
class MainActivity : ComponentActivity() {  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContent {  
            AppTheme {  
                var count by remember { mutableStateOf( value: 0) }  
                var increment = { count++; }  
                Surface(  
                    modifier = Modifier.fillMaxSize(),  
                    color = MaterialTheme.colorScheme.background  
                ) {  
                    Counter(count = count, incrementCount = { increment })  
                }  
            }  
        }  
    }  
}  
  
@Composable  
fun Counter(count: Int, incrementCount: () -> Unit, modifier: Modifier = Modifier) {  
    Button(  
        modifier = modifier.fillMaxSize(),  
        onClick = { incrementCount } { this: RowScope  
        Text( text: "Pressed $count times")  
    }  
}
```



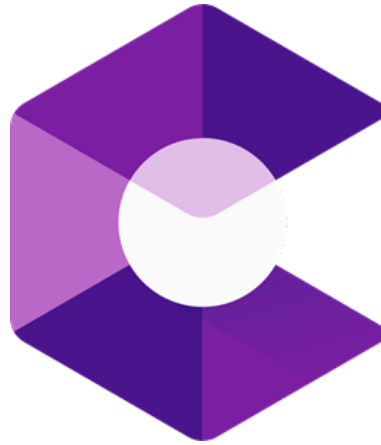
## Firebase



Firebase is a managed PaaS (Platform-as-a-Service) that offers numerous services. Spell Slinger takes advantage of Firebase Authentication in order to streamline creating accounts and logging users in. The realtime database is useful to store not only account data, but also multiplayer battle data temporarily.

Firebase Functions are distributed cloud functions with tight integration with the firebase platform, which allows for the server related capabilities in Spell Slinger.

## AR Core



ARCore is a SDK to implement augmented reality in Android applications. It provides APIs for motion tracking, light estimation and environmental understanding (which helps games figure out which elements can be used as planes in augmented reality)..

## Google Maps



Spell Slinger uses Google Maps to identify and create the users surrounding environment within the world map. The Google Maps API allows developers to place markers with custom icons (such as player, creature or spell well icons) at specific locations on the map. The detail of the map can be modified through the use of JSON styling in order to add, remove, or change elements (such as line thickness or fill color).

# Design Patterns

Jetpack Compose insists on a Model-View-ViewModel (MVVM) Design Pattern.

## Model

The Model in MVVM is a data class. It has no behavior, only pure data that is manipulated through the View Model.

## View Model

The View-Model in MVVM stores the model as a property and has a number of functions that contain complex logic for manipulating it, that's usually invoked by outside events from the View.

## View

The View in the presentation layer bridges the gap between the user and the view-model. The user interacts with the view, the view forwards that to the appropriate ViewModel, the ViewModel performs an operation on the Model, and then the View updates to display the new values of the Model.

# Sequence Diagrams

## Register

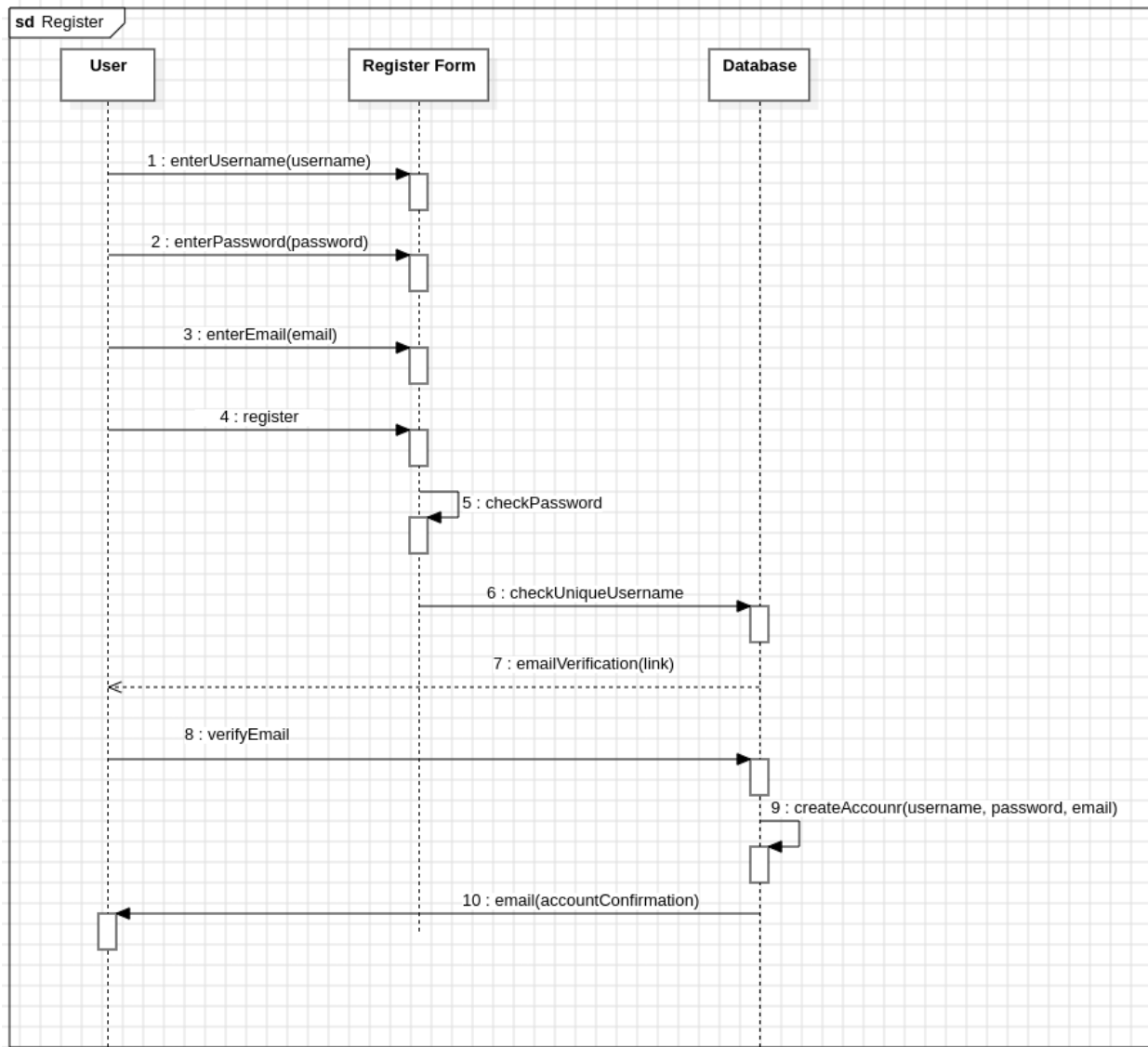


Figure 2-1 Sequence Diagram - Register

# Login

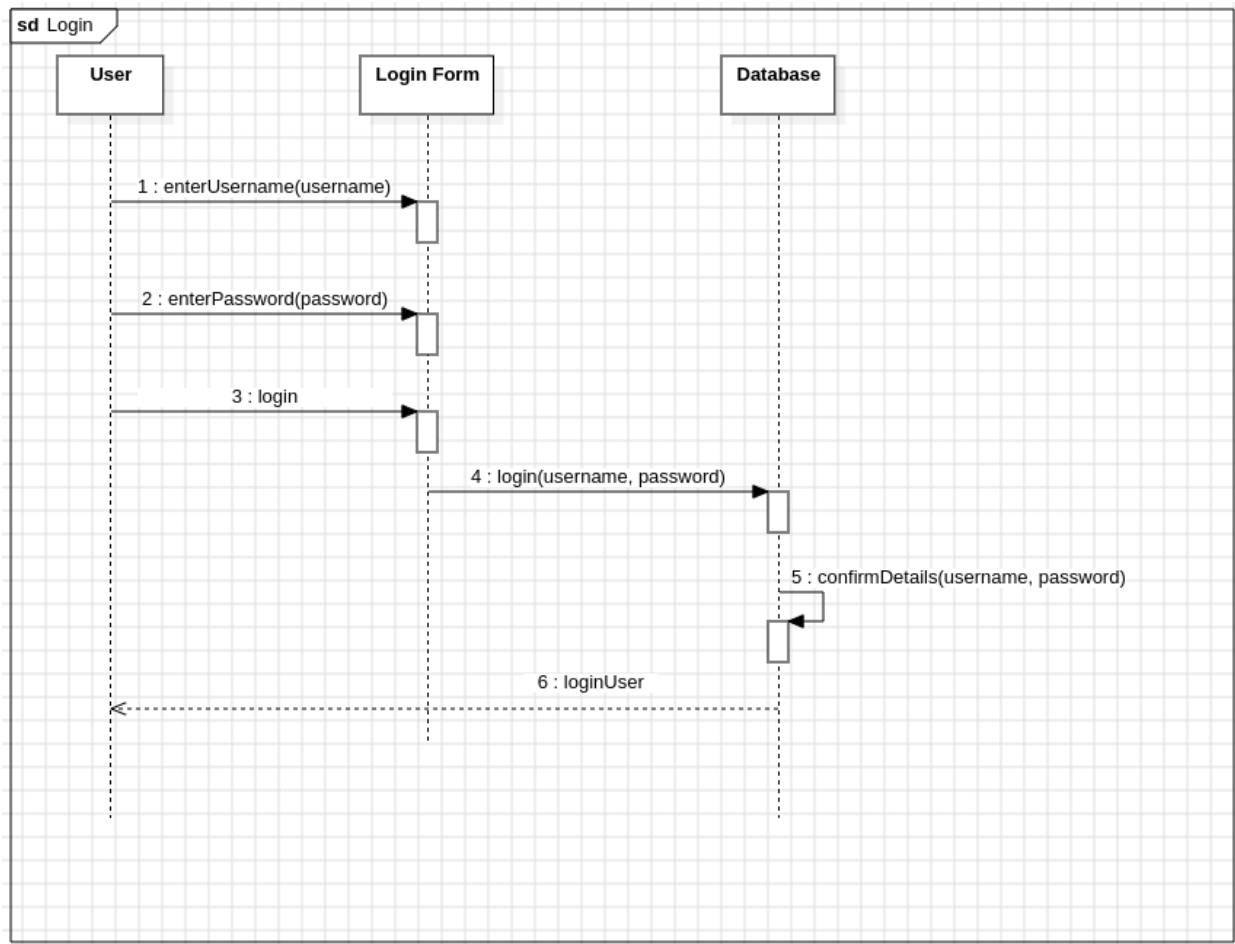


Figure 2-2 Sequence Diagram - Login

# Logout

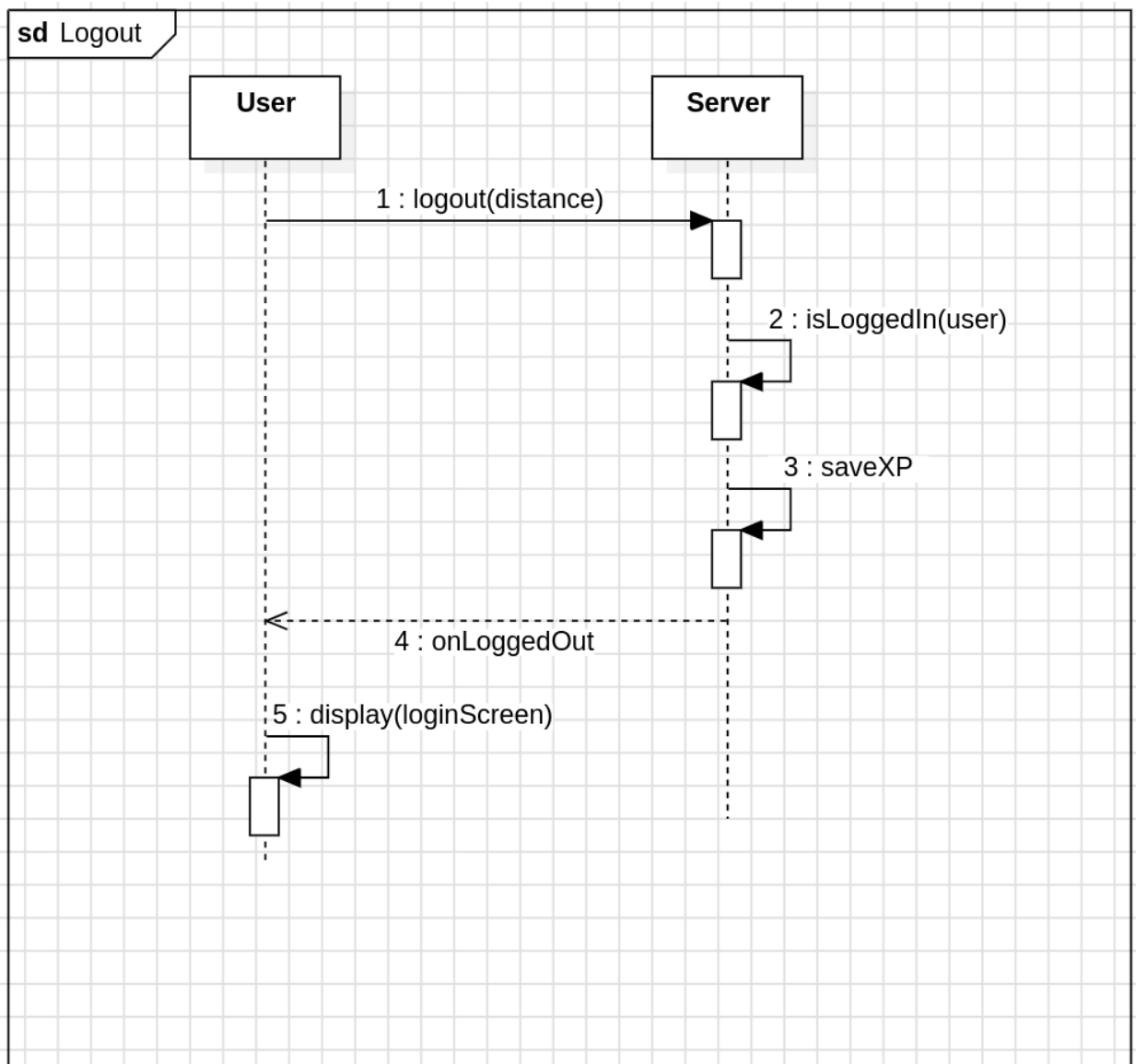


Figure 2-3 Sequence Diagram - Logout

## Delete Account

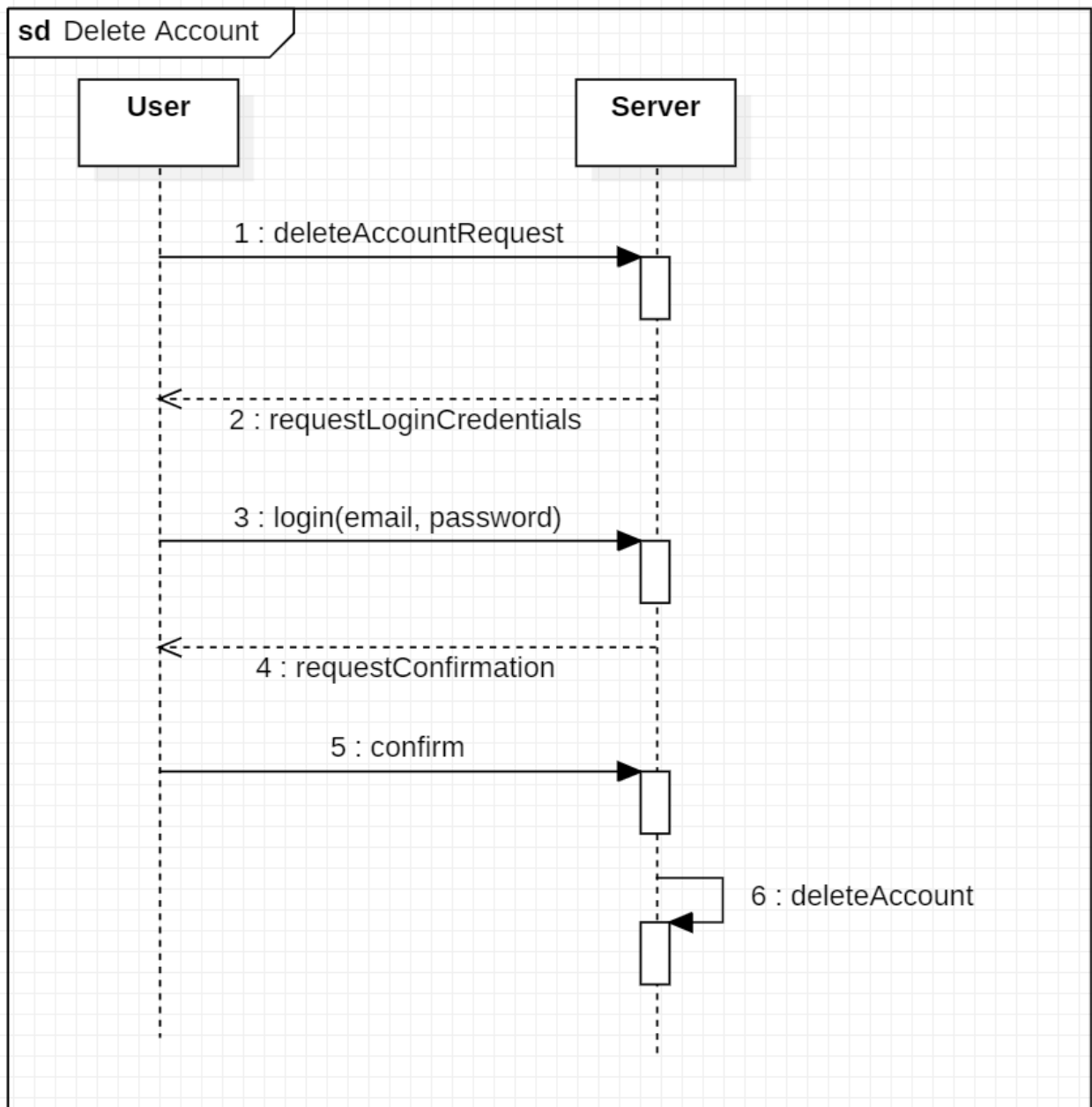


Figure 2-4 Sequence Diagram - Delete Account



# Train

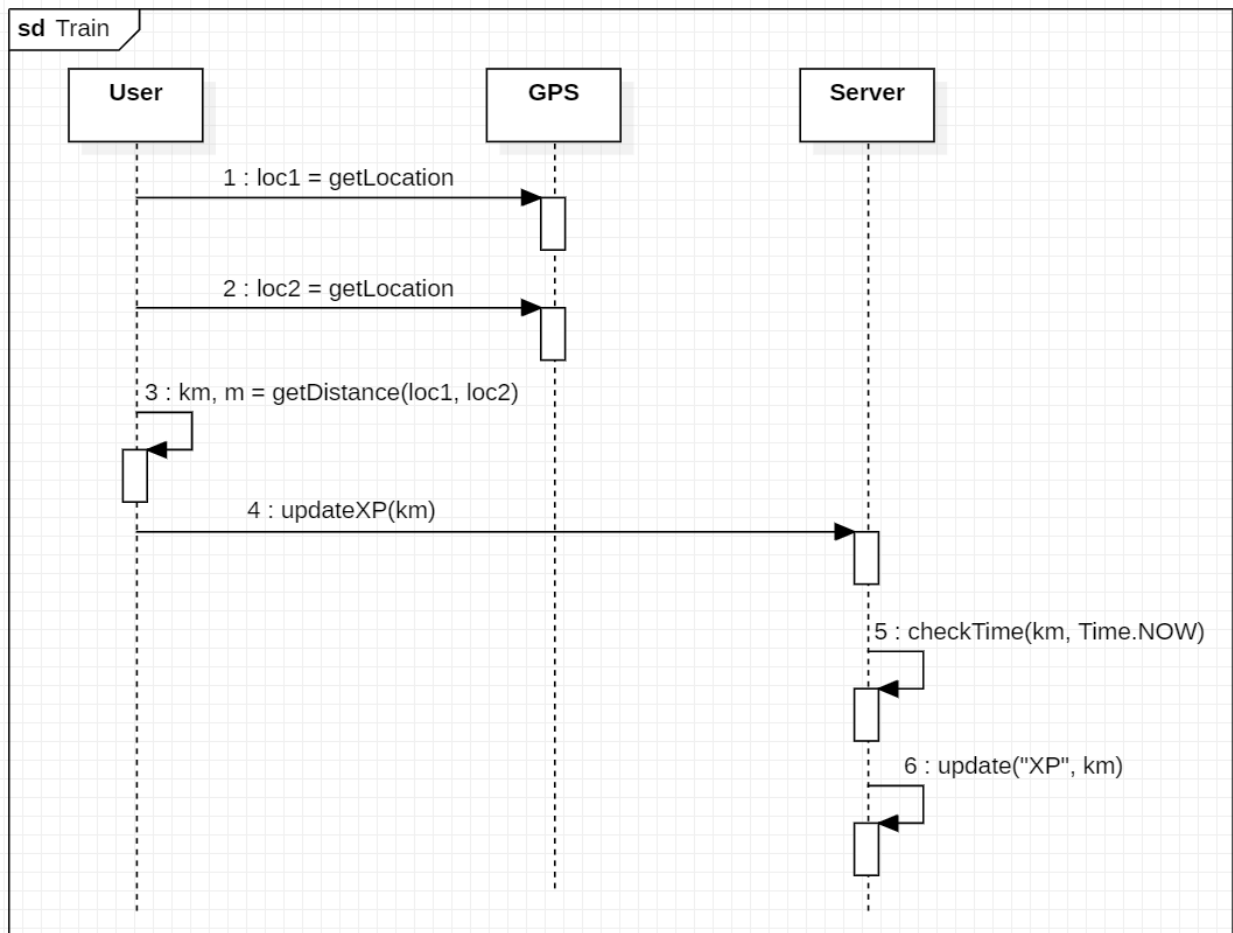


Figure 2-5 Sequence Diagram - Train

# Scan

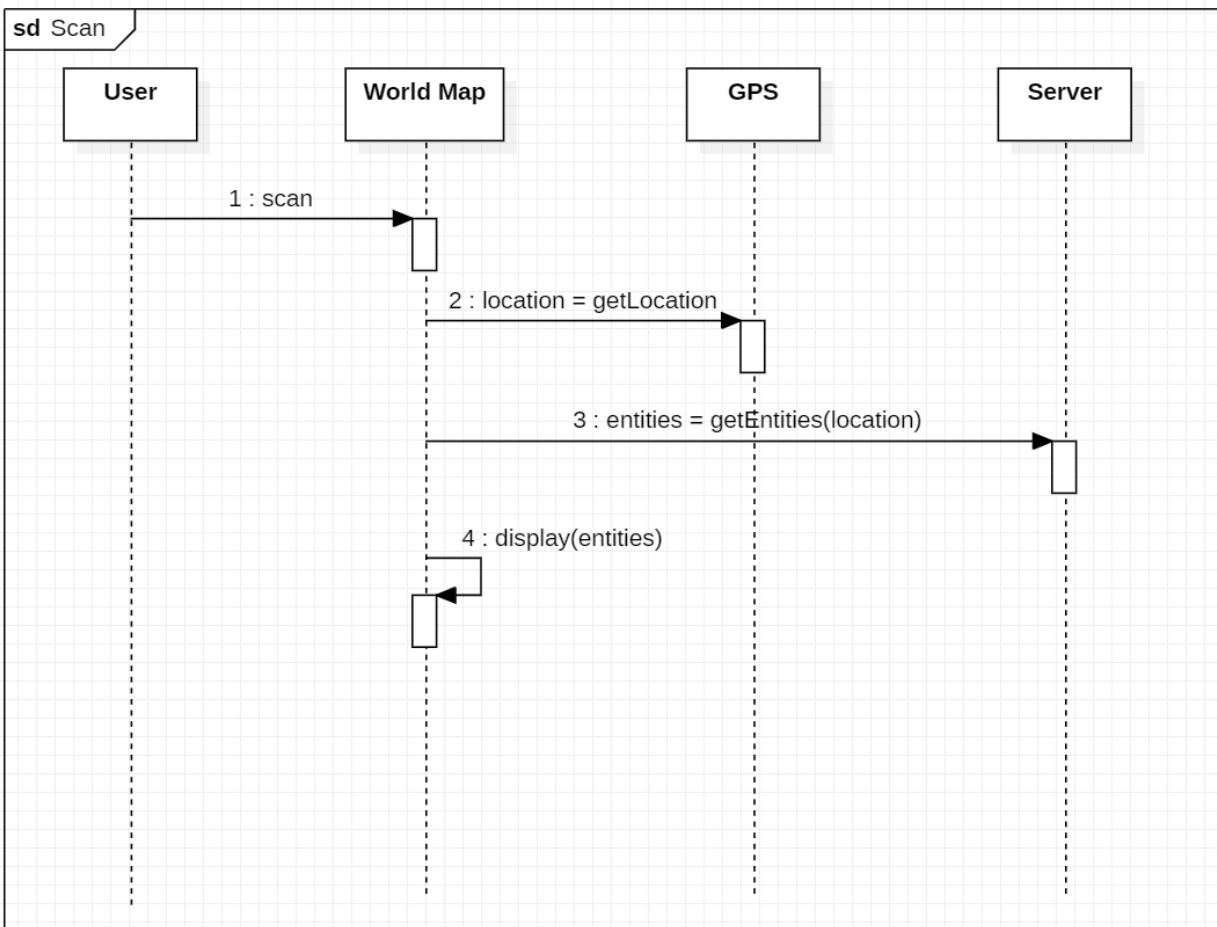


Figure 2-6 Sequence Diagram - Scan

## Draw Spell

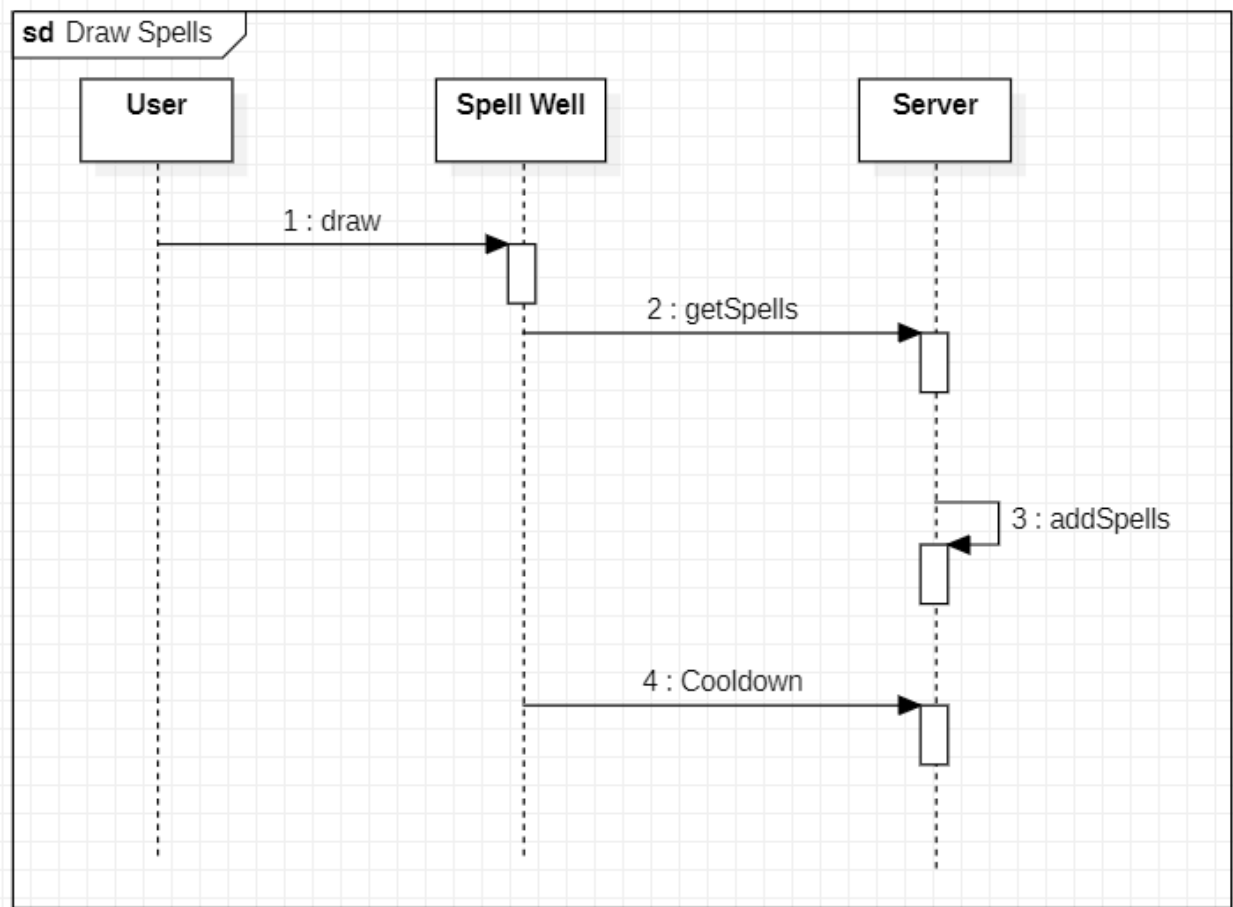


Figure 2-7 Sequence Diagram - Draw

# Equip Spell

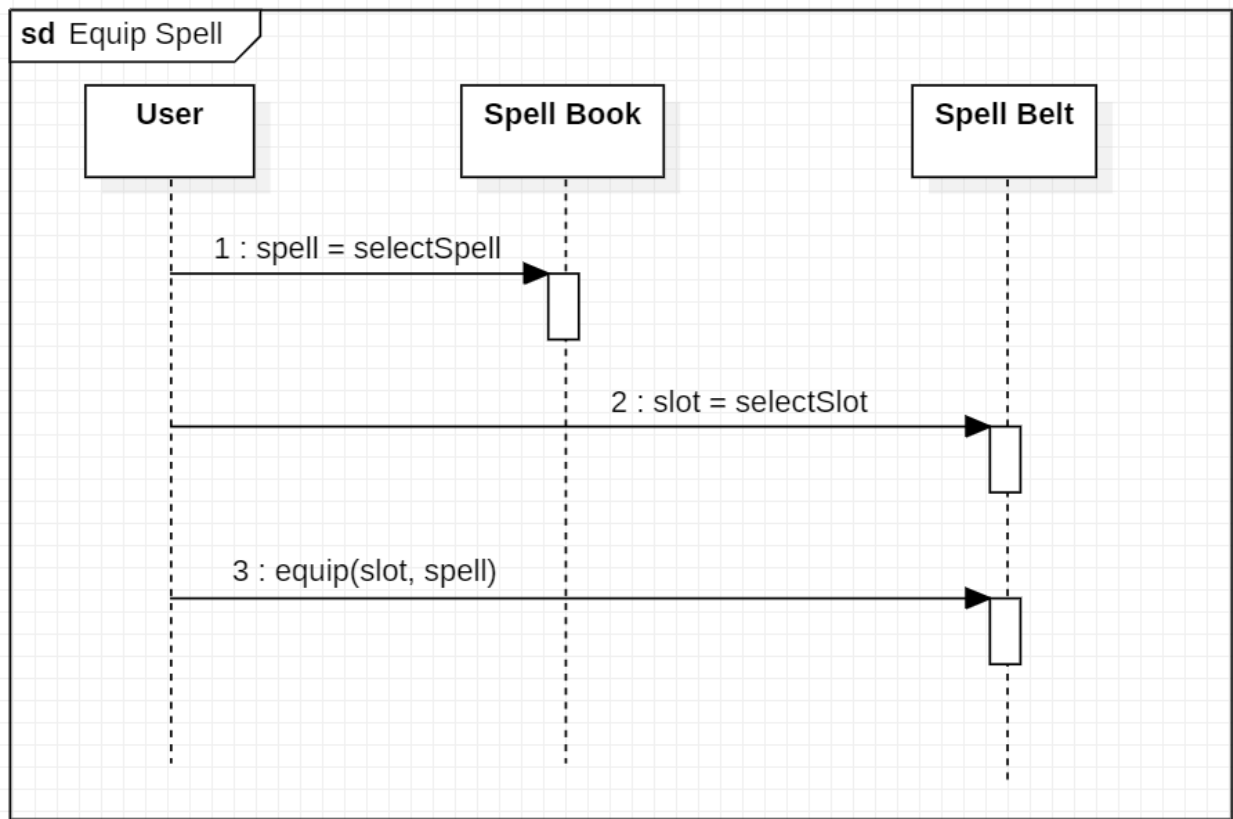


Figure 2-7 Sequence Diagram - Equip Spell

# Increase Stat

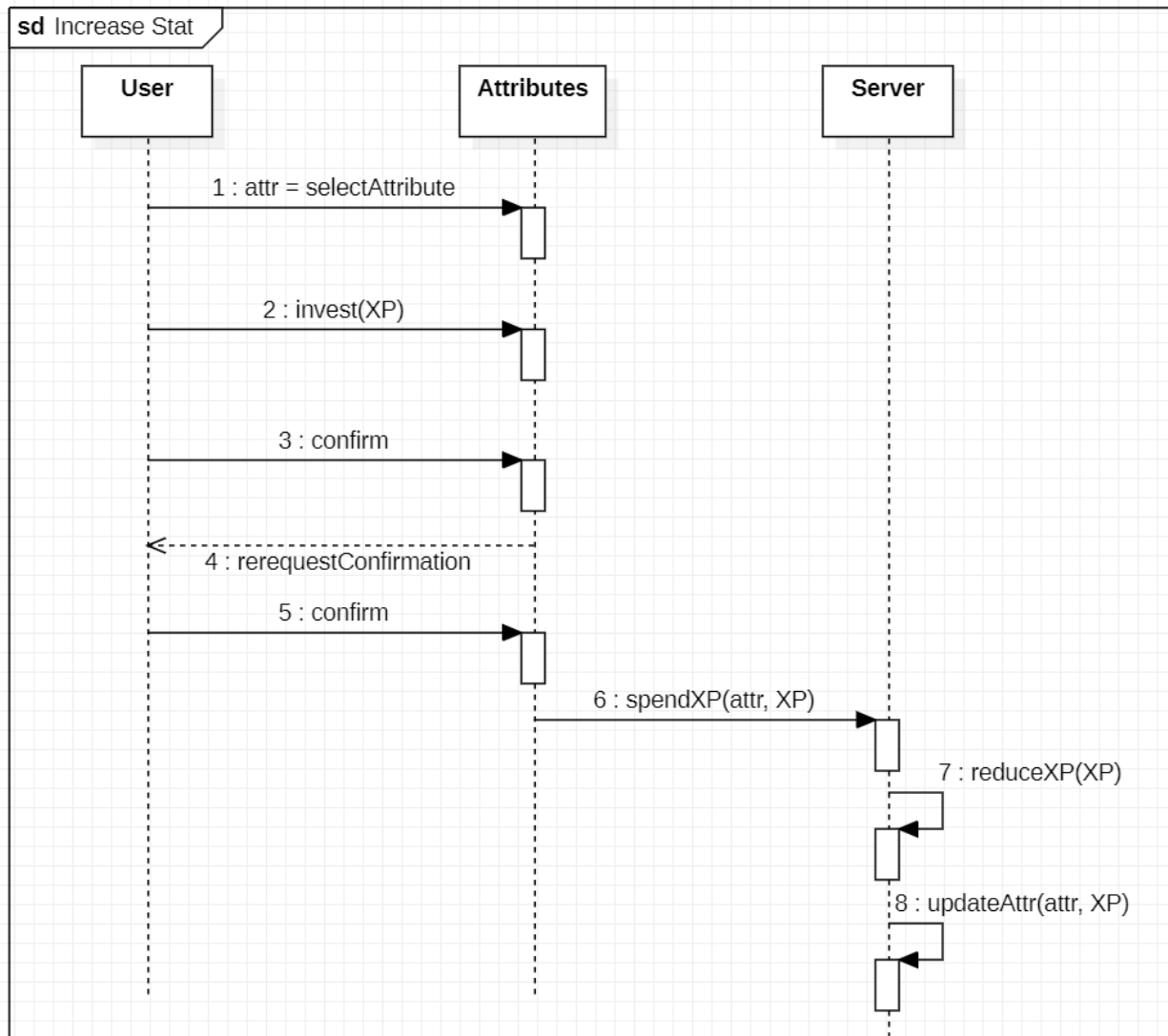


Figure 2-8 Sequence Diagram - Increase Stat

# Battle Player

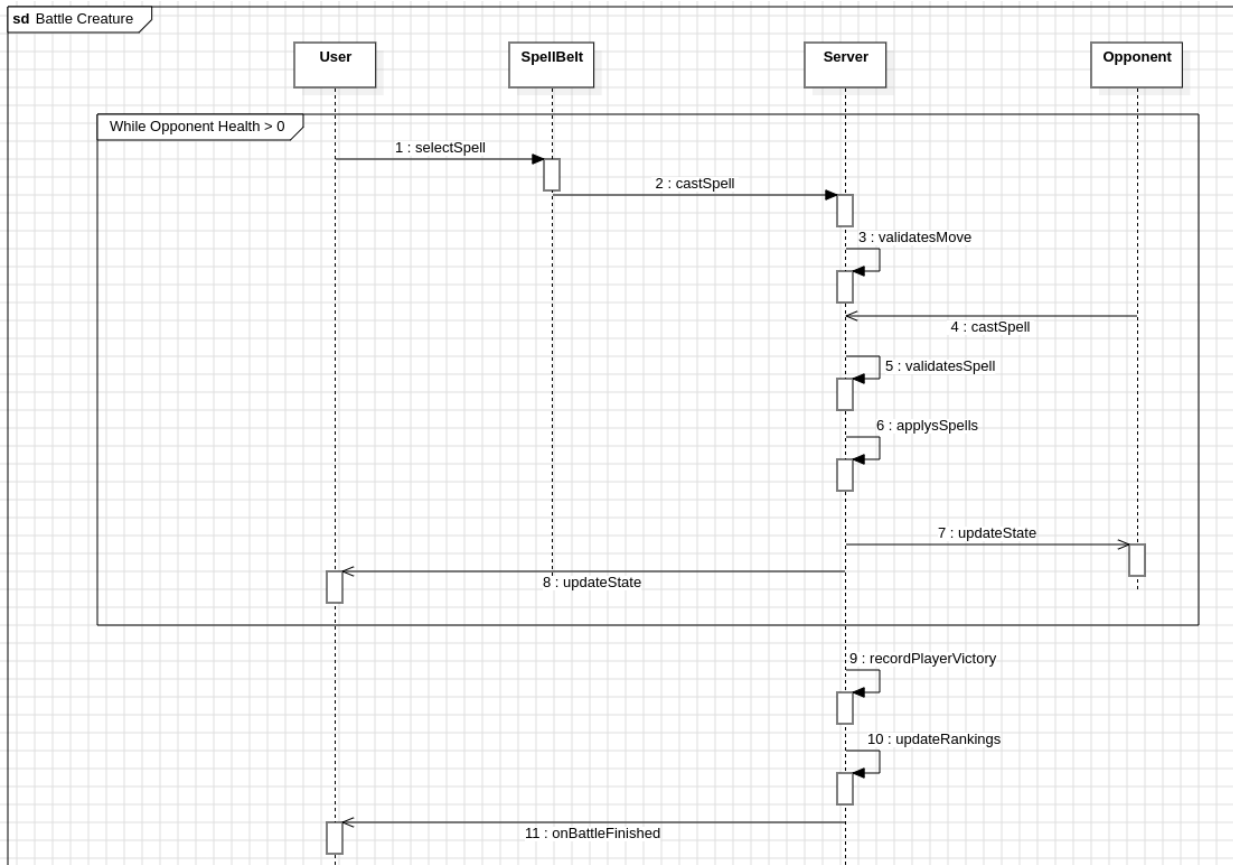


Figure 2-9 Sequence Diagram - Battle Player

# Battle Creature

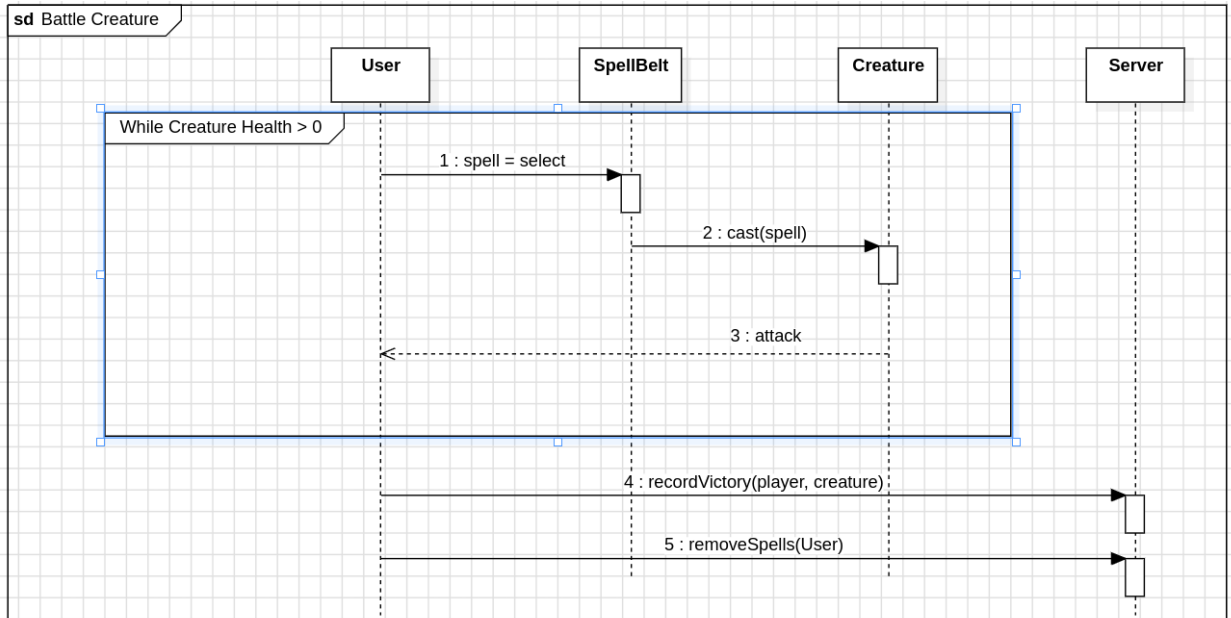


Figure 2-10 Sequence Diagram - Battle Creature

## View Leaderboard

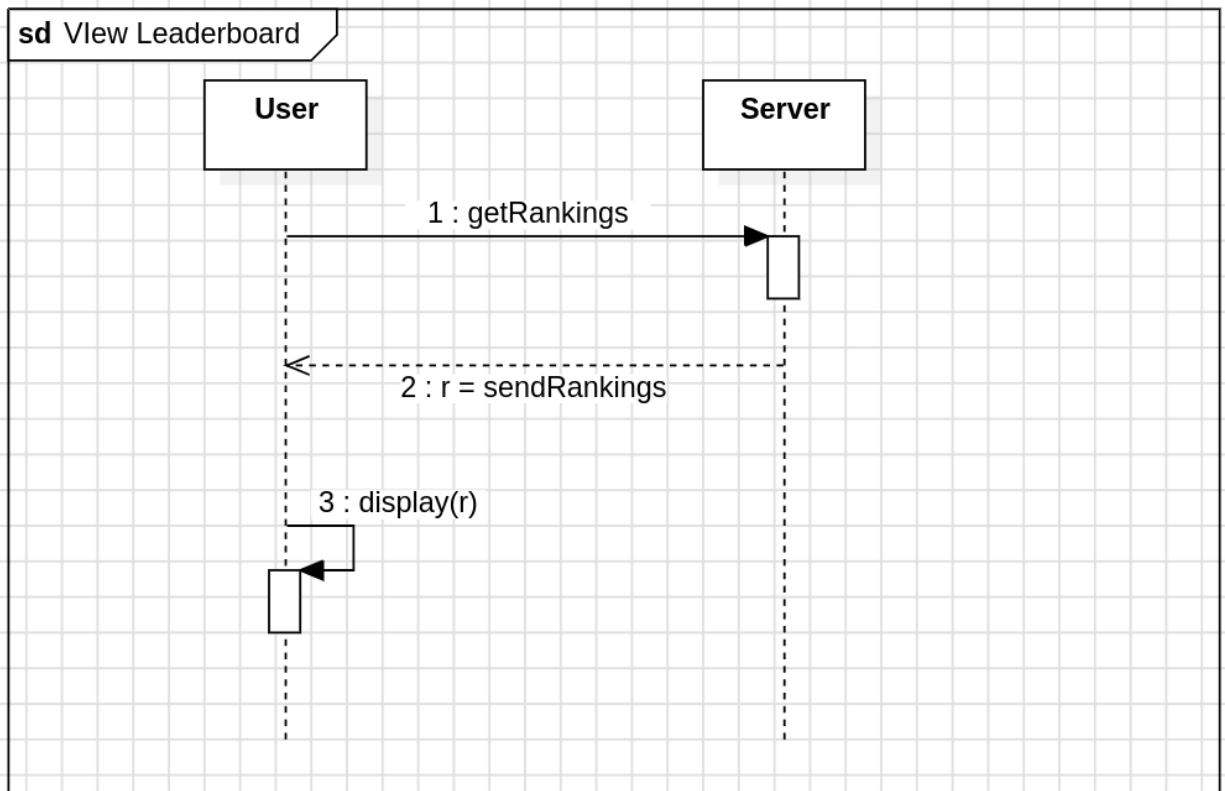


Figure 2-11 Sequence Diagram - View Leaderboard



# Class Diagram

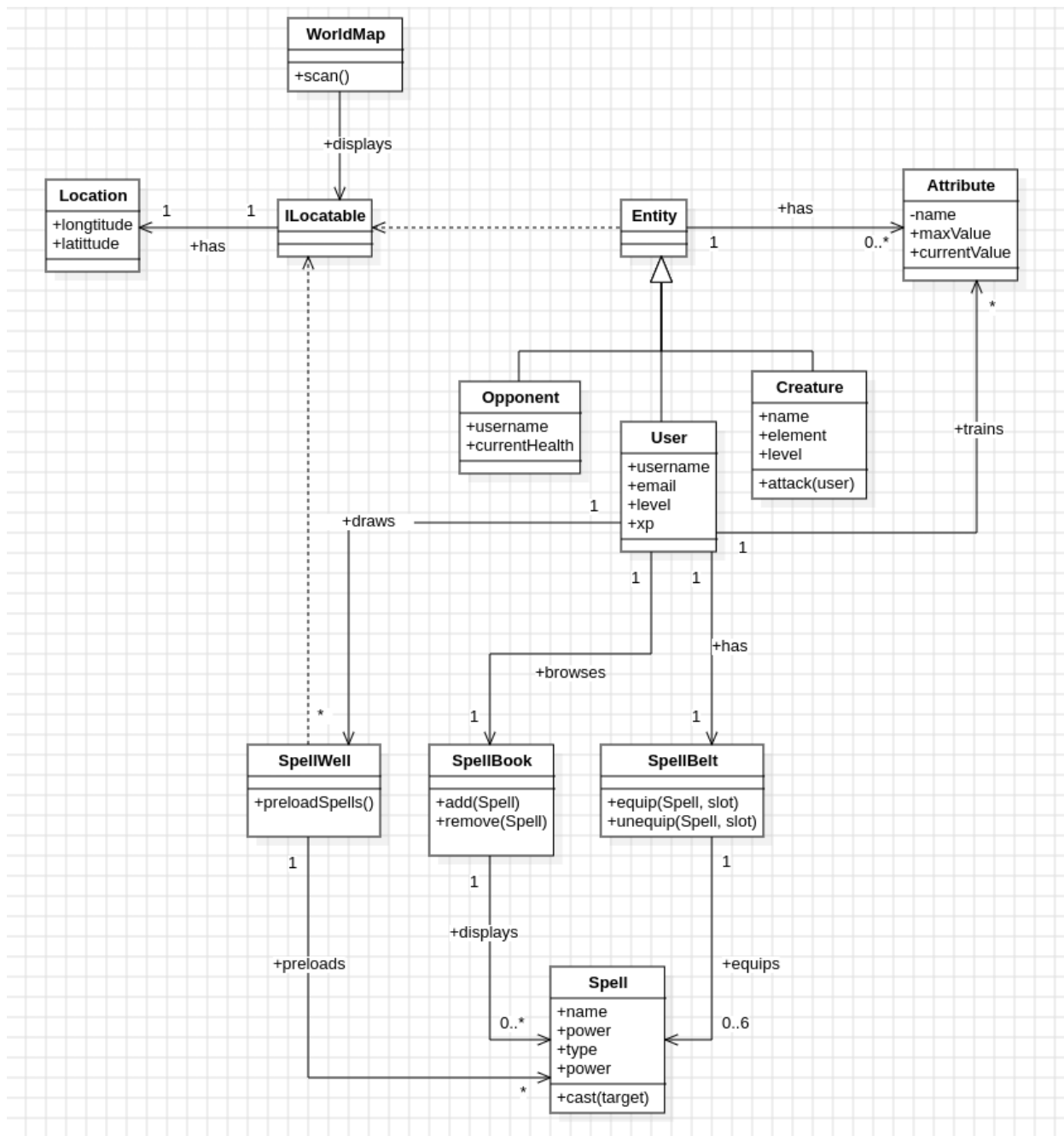


Figure 3-1 Class Diagram

# Database Layout

## Users

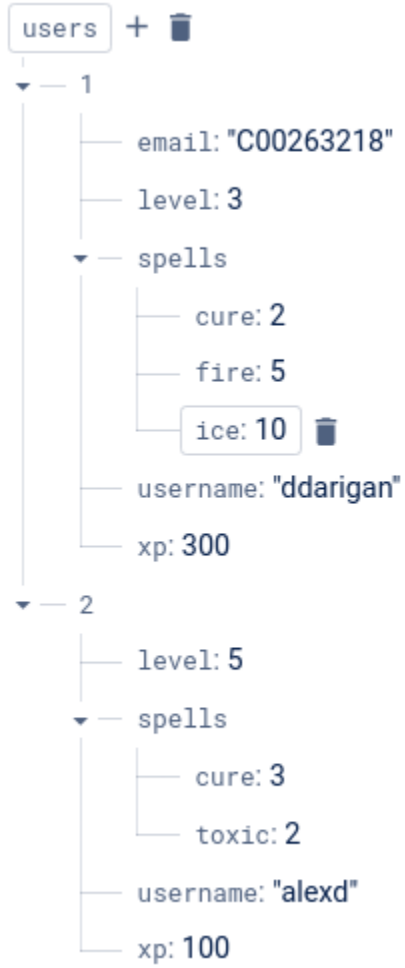


Figure 4-1 Database Layout - User

### Creature



Figure 4-2 Database Layout - Creature

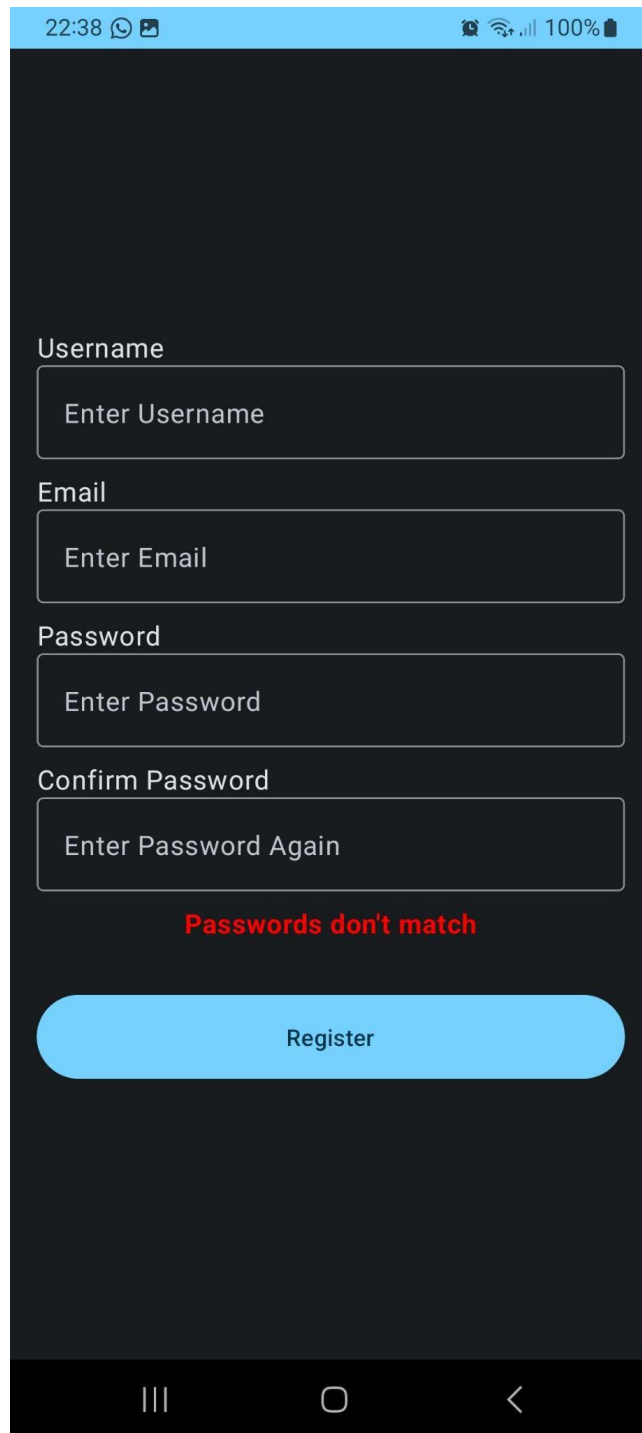
### Spell Well



Figure 4-3 Database Layout - Well

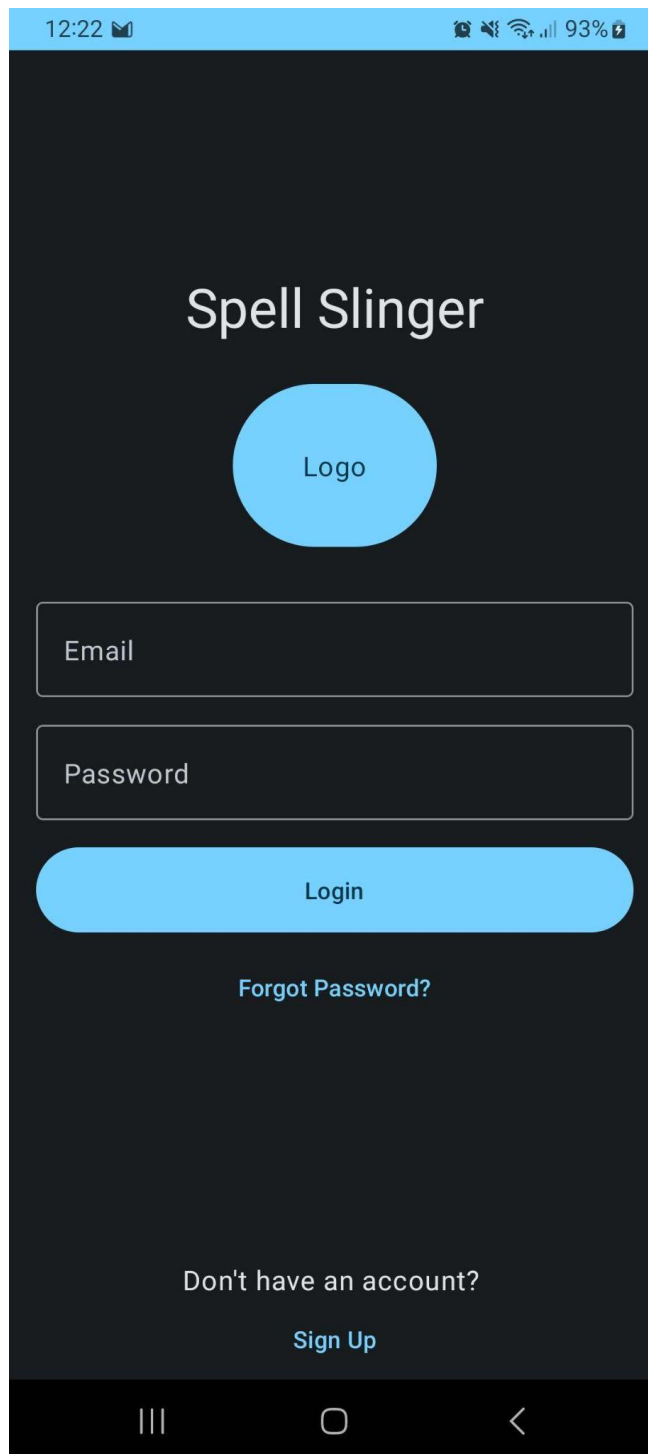
# Prototype Screens

## Register Screen

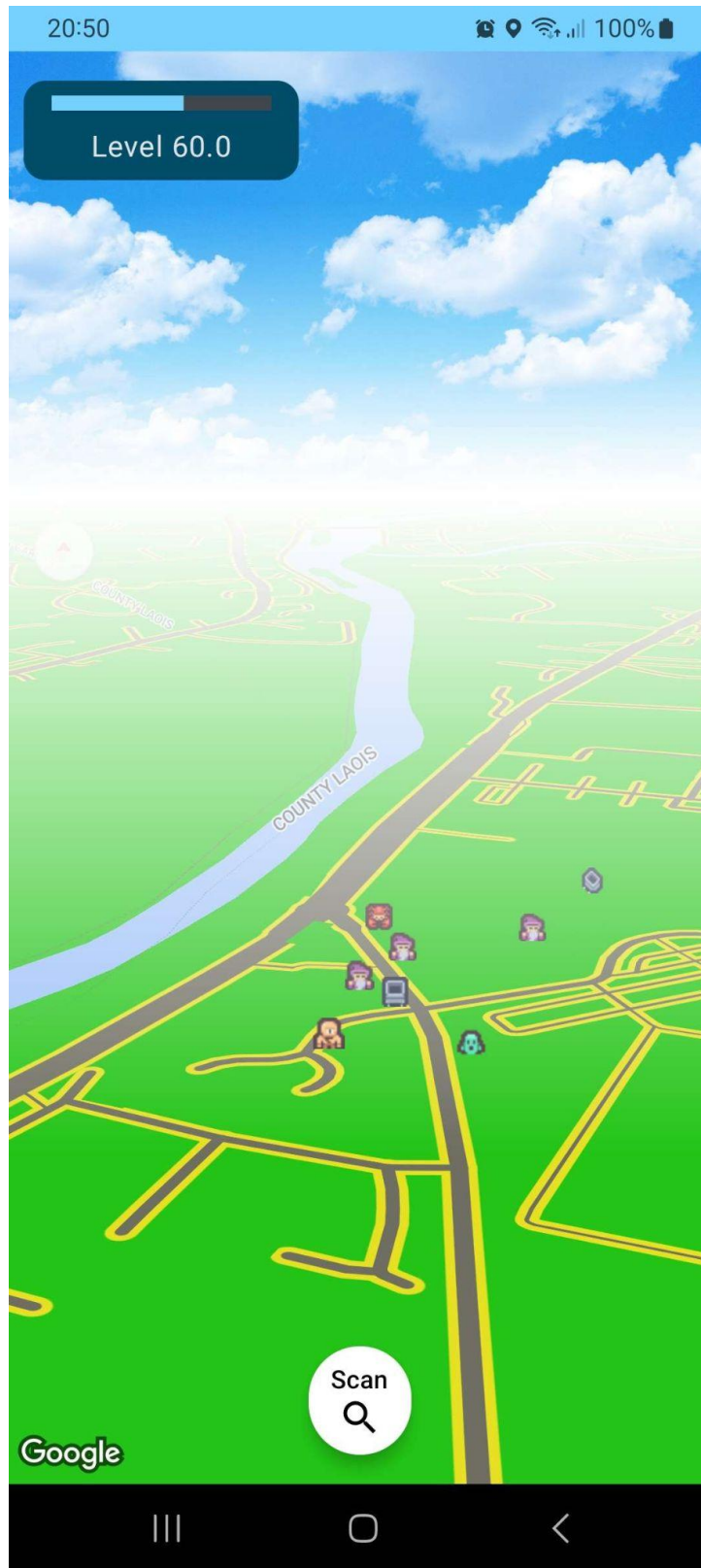


The image shows a mobile application prototype for a registration screen. The screen has a dark grey background. At the top, there is a light blue status bar with the time 22:38, a lock icon, a Wi-Fi icon, a signal strength icon, and a battery icon showing 100%. Below the status bar, the title "Register Screen" is centered. The form consists of four input fields, each with a label above it: "Username" (with placeholder "Enter Username"), "Email" (with placeholder "Enter Email"), "Password" (with placeholder "Enter Password"), and "Confirm Password" (with placeholder "Enter Password Again"). Below the "Confirm Password" field, there is a red error message: "Passwords don't match". At the bottom of the form, there is a large, rounded, light blue button with the text "Register". The bottom of the screen shows the Android navigation bar with three icons: a home button (three vertical lines), a back button (a circle), and a forward button (a triangle).

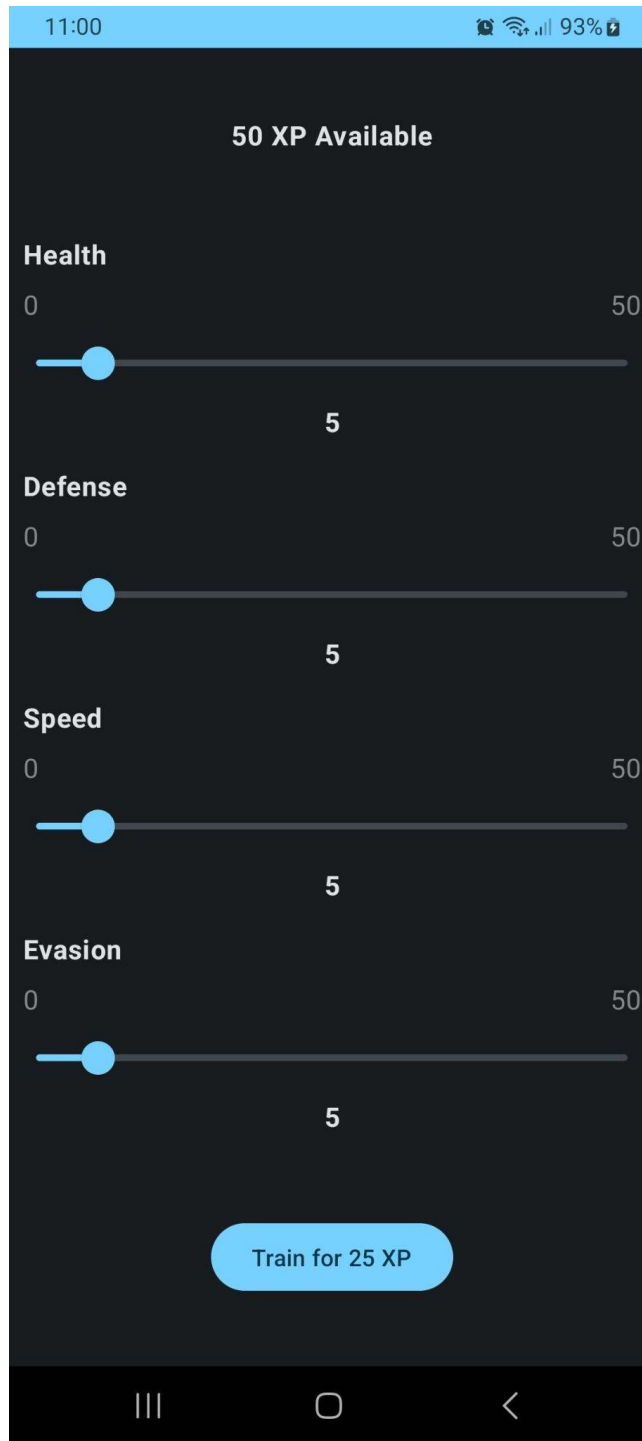
## Login Screen



# World Screen

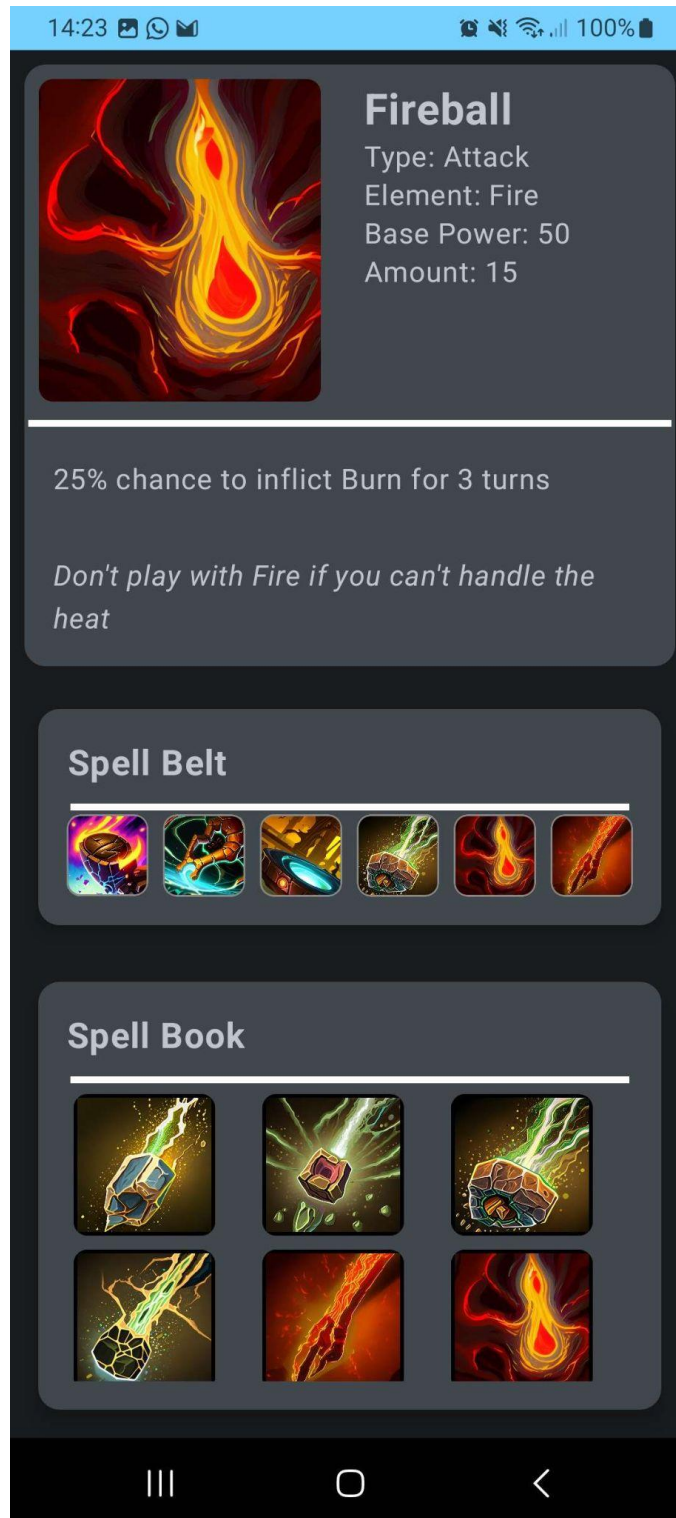


## Training Screen

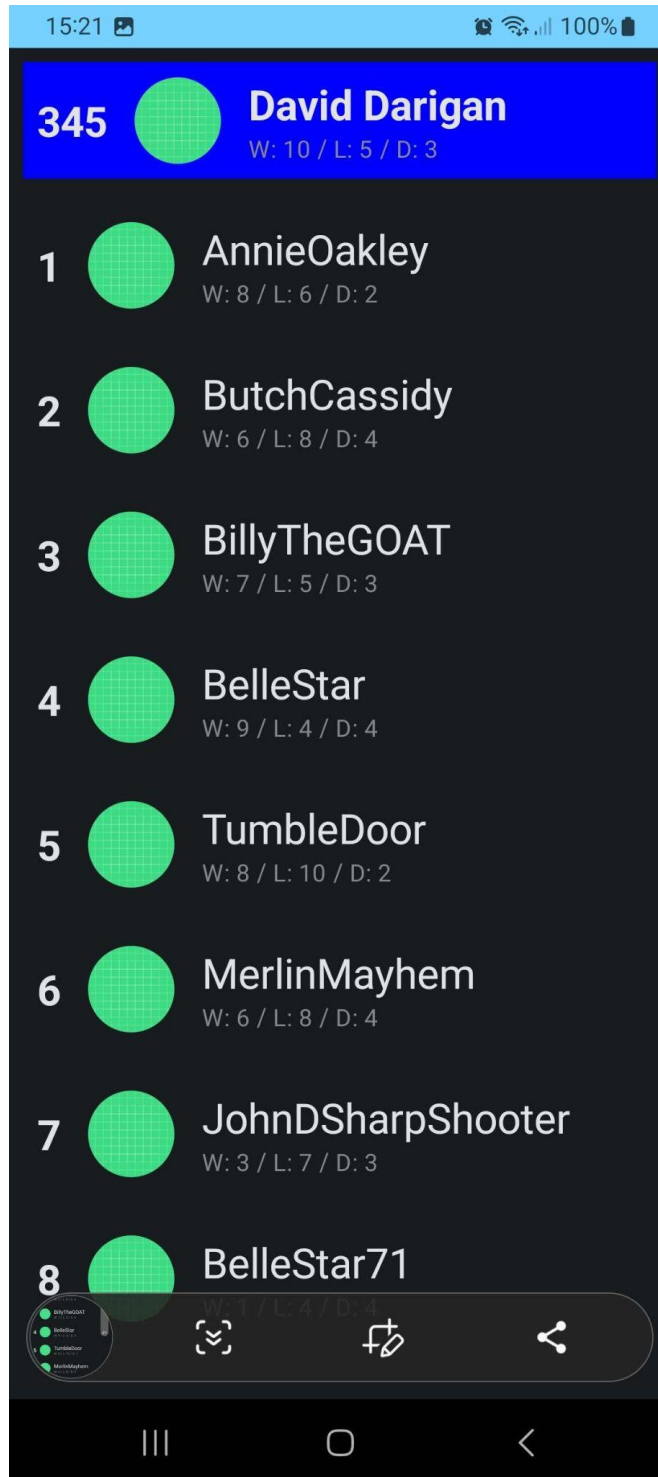




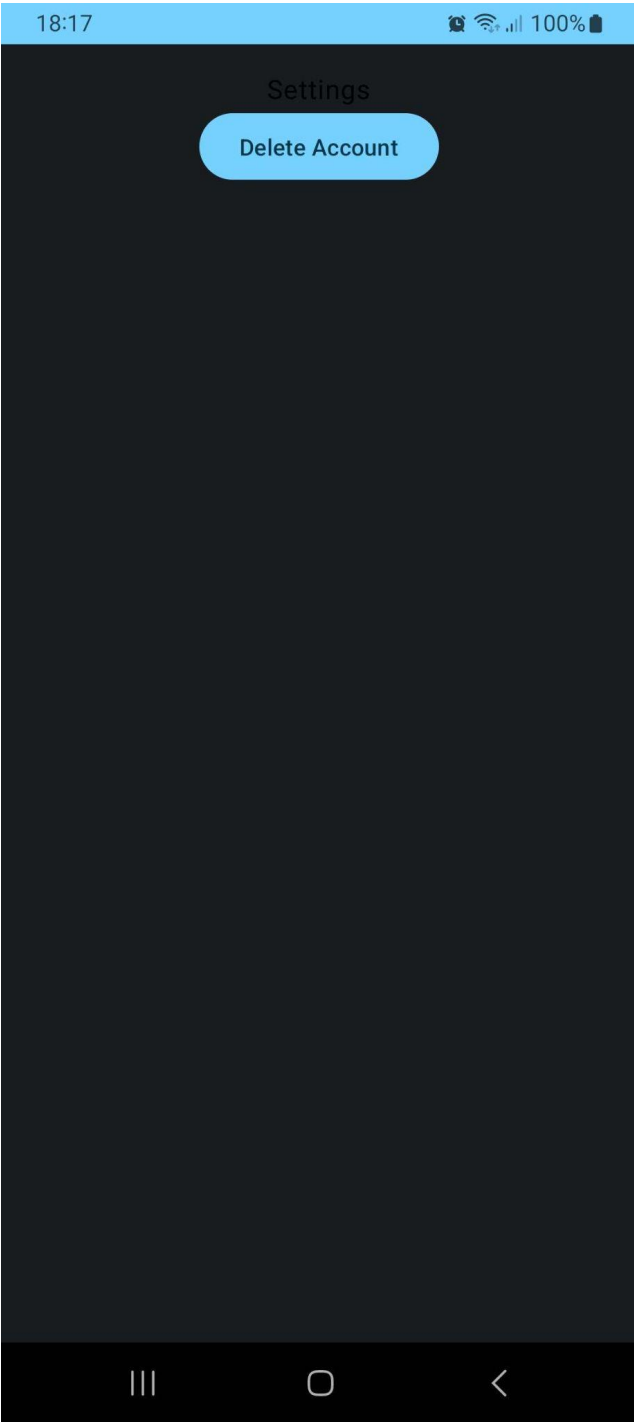
## Spell Book Screen



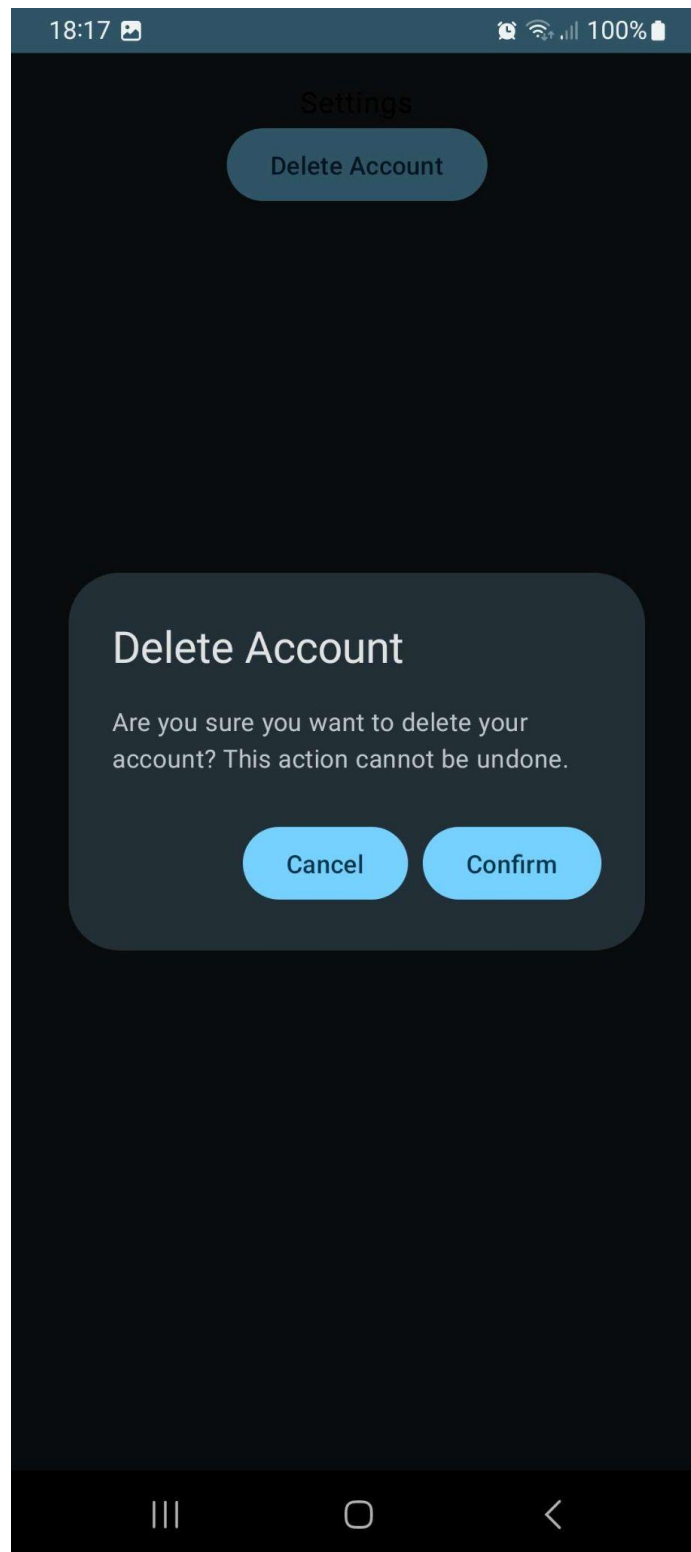
## Leaderboard Screen



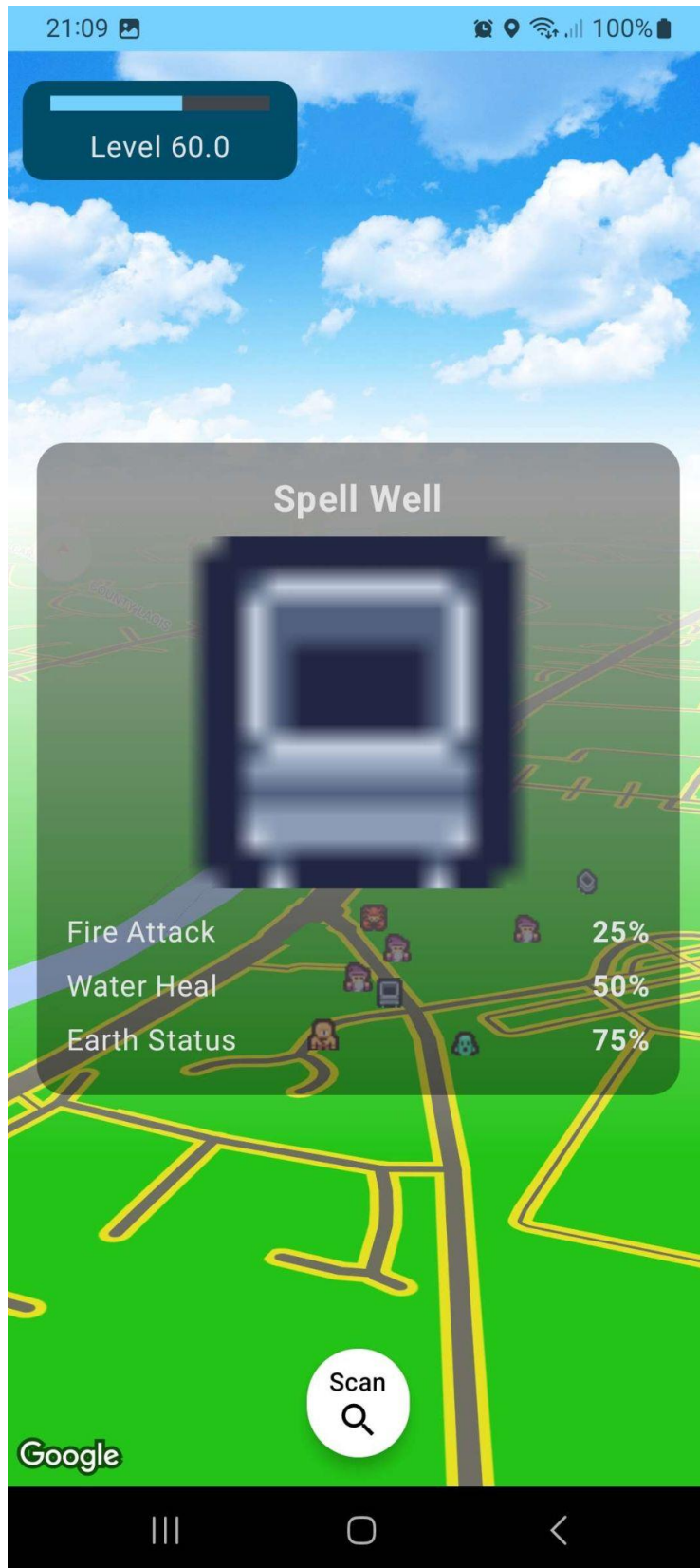
# Settings Screen



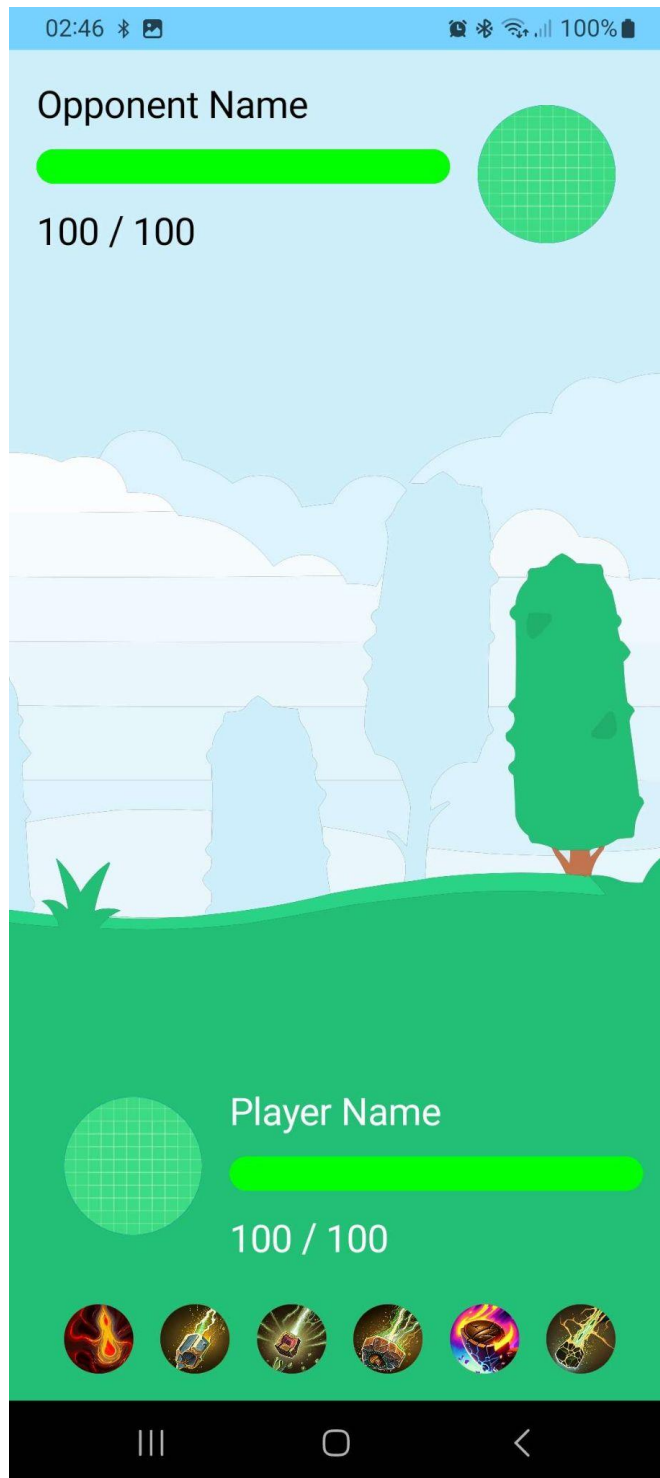
## Delete Account Screen



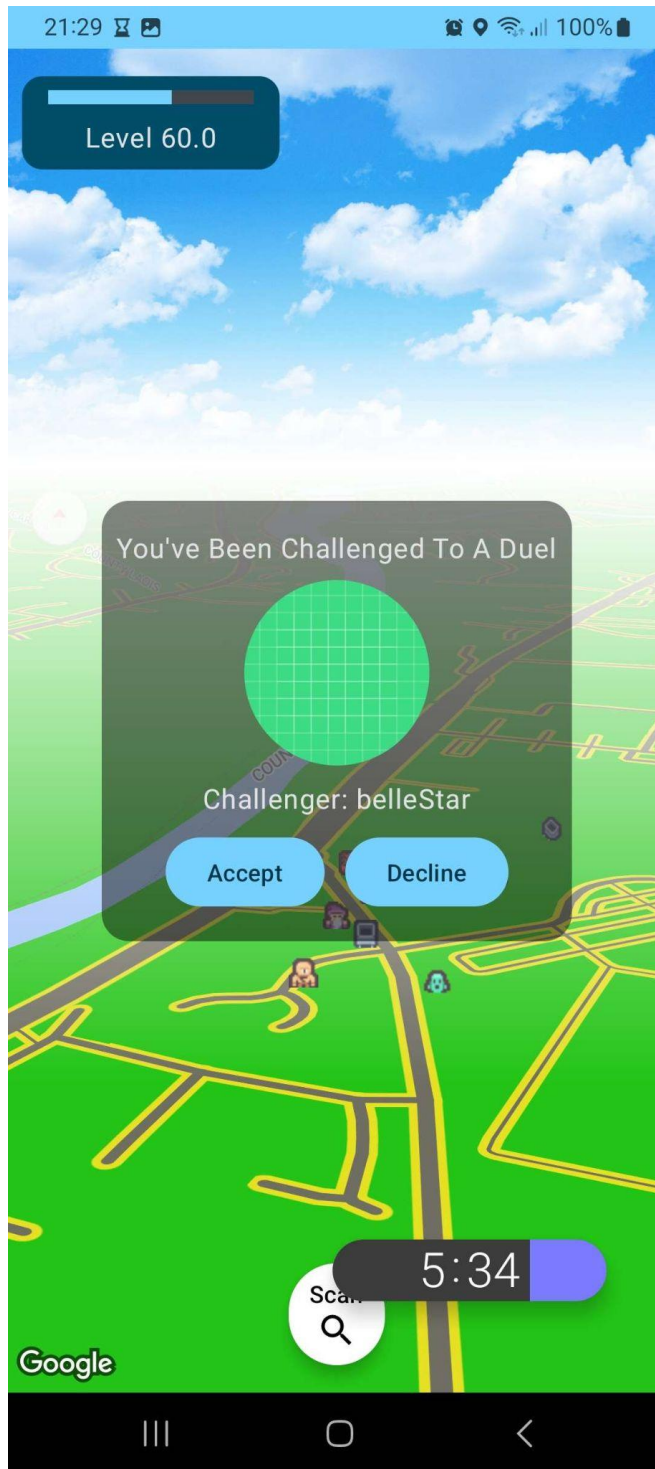
# Spell Well Screen



## Battle Screen



## Battle Invite Screen



# Milestones

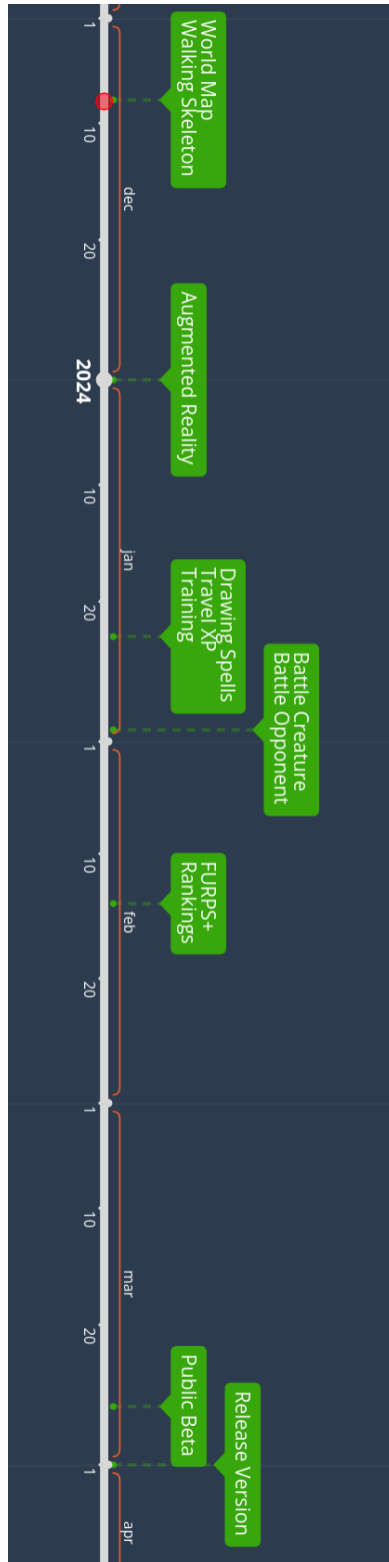


Figure 5 - Milestones



## Attributions

Spell Icons from <https://captaincatsparrow.itch.io/>

Other Assets from <https://www.kenney.nl/assets>

## Conclusion

The features of the SpellSlinger geolocation mobile gaming have been outlined in this document. The reasonings for the particular technologies, tools and design patterns have been detailed. The flow of code in the game has been detailed through Sequence Diagrams and a class diagram. The database layouts were presented to demonstrate the data model used in the application. Prototype screens are used to display the initial style of the game and a guideline going forward.