Software Design Document

Version 1.0

26 January 2022

Party Game Manager

Multiplayer scribble game.

Software Management Team 3

Faculty of Automatic Control and Computers, University POLITEHNICA of Bucharest.

Project purpose **3**

Document Contents **3**

Data model **3**

Temporary data structures 3

Data file format 3

Database structure 3

Database diagram 3

Tables 4

Users 4

Rooms 4

Game Session 4

Architectural model **4**

Architectural template 4

Architectural diagram 5

Component Description 5

Technological limitations 5

Component interaction 6

User Interface model **6**

Interface succession 6

Application screens 7

Log In Screen 7

Sign Up Screen 8

Game Session Browser 9

Admin Dashboard 10

Game Session 11

Testing Environment **11**

Manual tests 11

Testing Infrastructure 11

Test Suite 11

1. Project purpose

This document is intended to accurately and completely describe the solution designed for the multiplayer game manager software system named Party Game Manager. The document serves as a unique solution-building guide for the project development team.

2. Document Contents

The document consists of four essential sections:

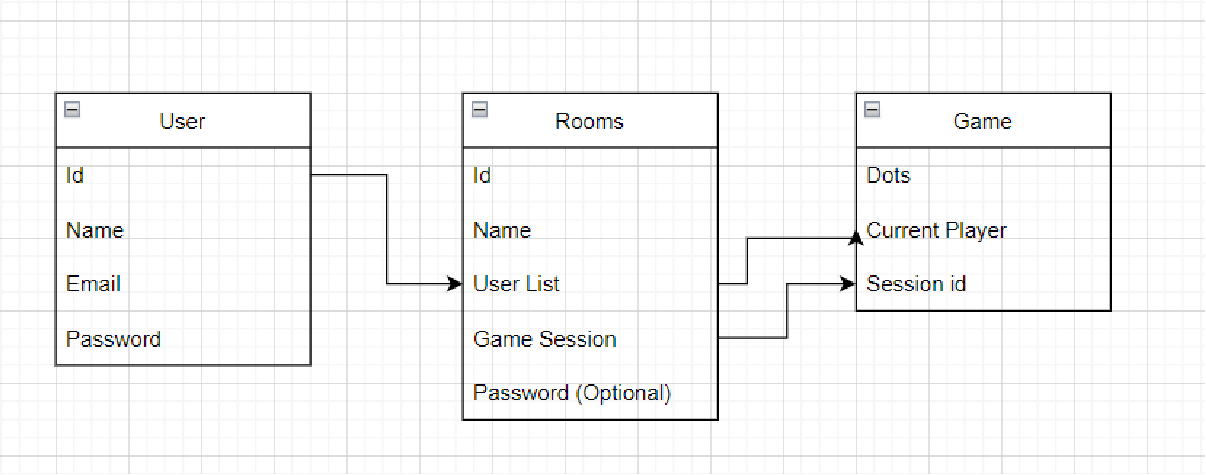
- Data model - presents the main data structures used, as well as the schema of the database data
- Architectural model - presents the architectural templates used, the system architecture and describes the components of the architecture
- User interface model - shows the user interface and the flow between windows
- Testing infrastructure - present all the tests inside the suite

3. Data model

a. Temporary data structures

b. Data file format

c. Database structure



i. Database diagram

ii. Tables

1. Users

The main table used for the login and sign-up systems, also used for the guest direct connect system. It stores all the necessary information about a user (Name, email, password) as well as a unique Id.

2. Rooms

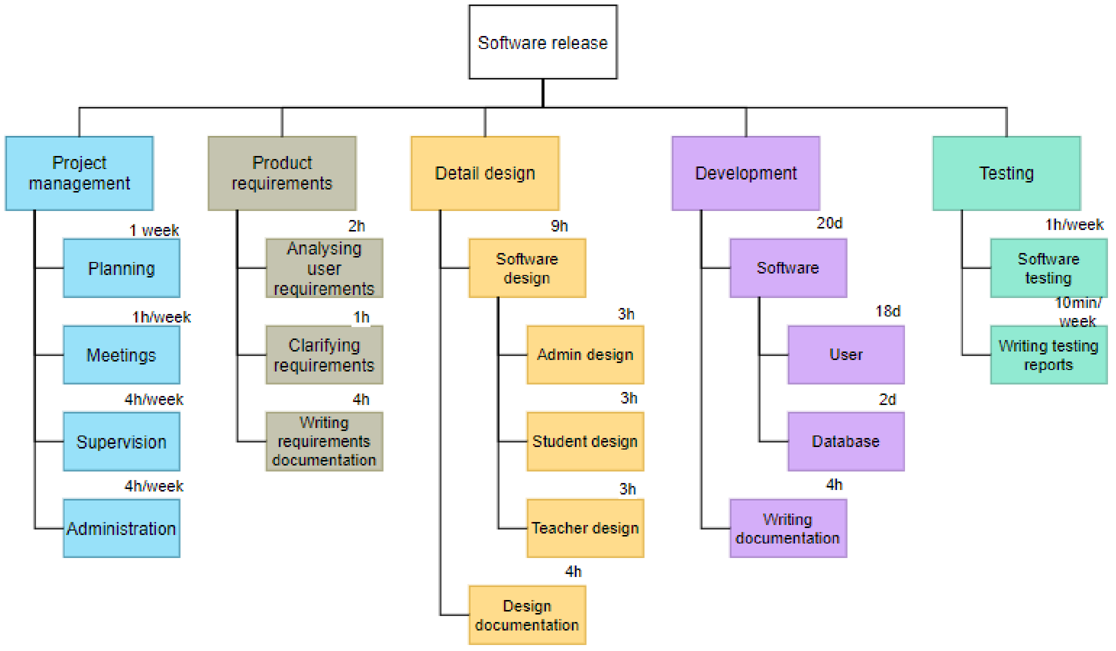
The rooms table has the association between the users and the game sessions, as well as the information about the room (capacity, password, etc.)

3. Game Session

The game session table holds the information necessary to sync the drawing games together as well as the current player who is drawing and a relation with the a room table entry.

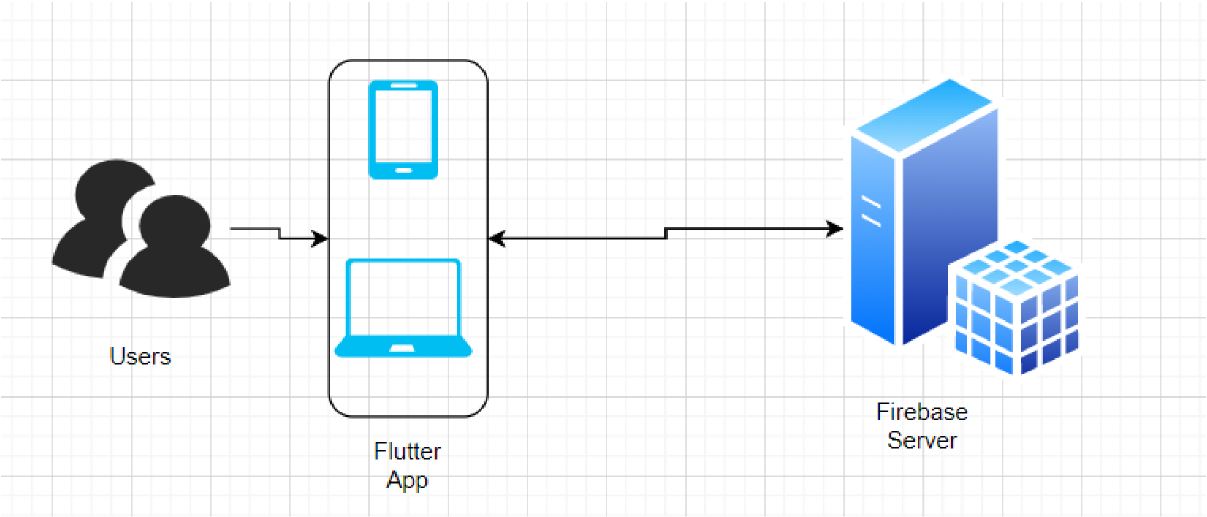
4. Architectural model

a. Architectural template



The project organization is split between the areas showcased in the diagram above. Party Game manager is organized as a mobile/web/desktop app, it is realized using the Google tech stack.
The mobile/web/desktop app communicates with the database through its user types.

b. Architectural diagram



The diagram below shows the design that has been chosen for the application, as well as the interaction between the components

c. Component Description

Software Components:

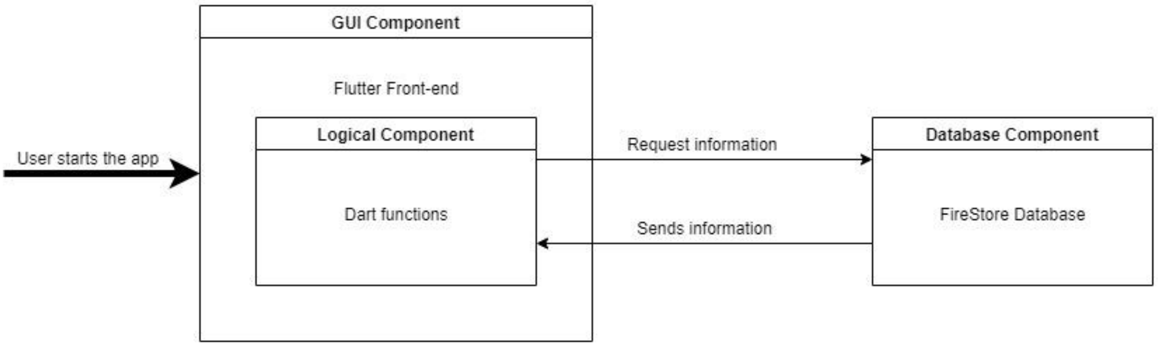
- **GUI Component -** The interface is realized with the flutter, and has a modern and intuitive look. It represents all the elements drawn on the screen.
- **Logical Component -** It is intertwined with the GUI component as they are both realized in dart. It represents all the functionality and actions taken by the app when the users interact with it.
- **Database Component -** It is responsible for the interactions with the database server and CRUD operations (Create, Update, Delete)**.**

d. Technological limitations

- The database that has been chosen is Firebase. It is a real-time (No)SQL database that provides both advantages and disadvantages. It is also limited in the case of large write operations.
- The front-end is developed in the flutter environment using Dart. This limits the ability to use vector graphics and animations and is also limited in application complexity. We are also limited in speed compared to native platform code.

These limitations do not impede the development of Party Game Manager

e. Component interaction

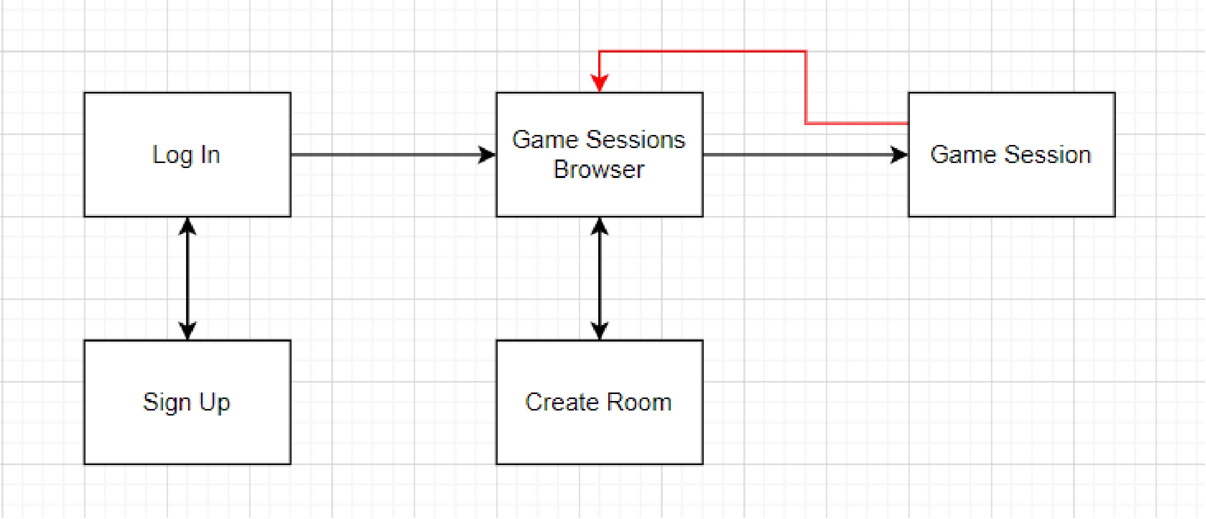


When the user starts the app he is greeted by the flutter front-end, behind it the dart code ensures the functionality and communication with the FireStore database. Firestore holds all the information about the users (email, name, game sessions), information about the available game sessions, and about each ongoing game.

5. User Interface model

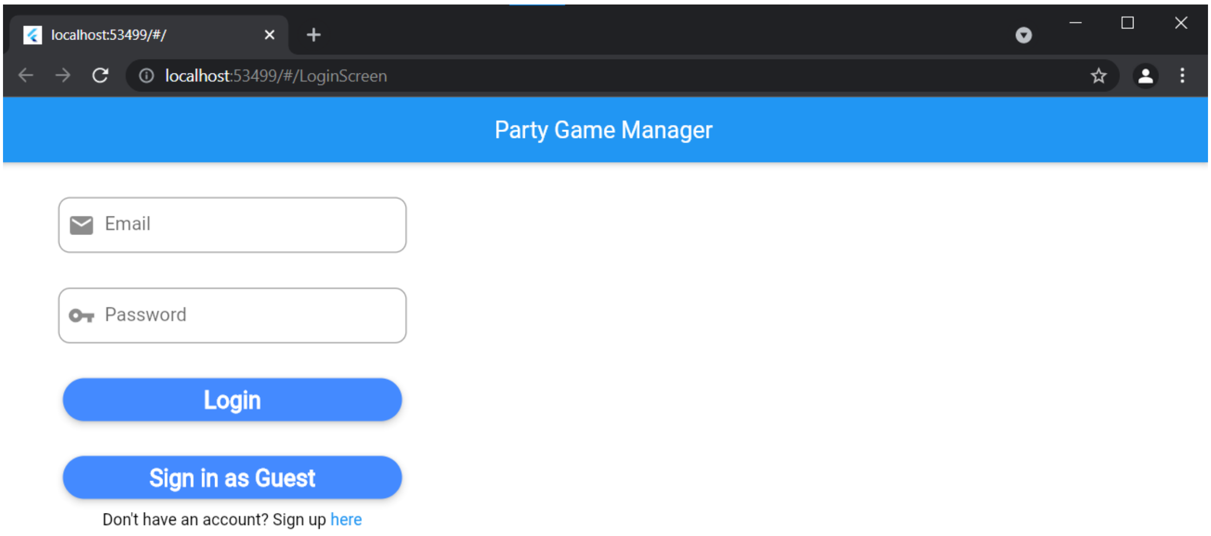
a. Interface succession

The above diagram showcases the interface flow inside the app for every kind of user. From every page, the user can log out and it’s redirected to the first screen. The flow is realized through intuitively labeled buttons and icons. The red arrow signifies the back button redirection.



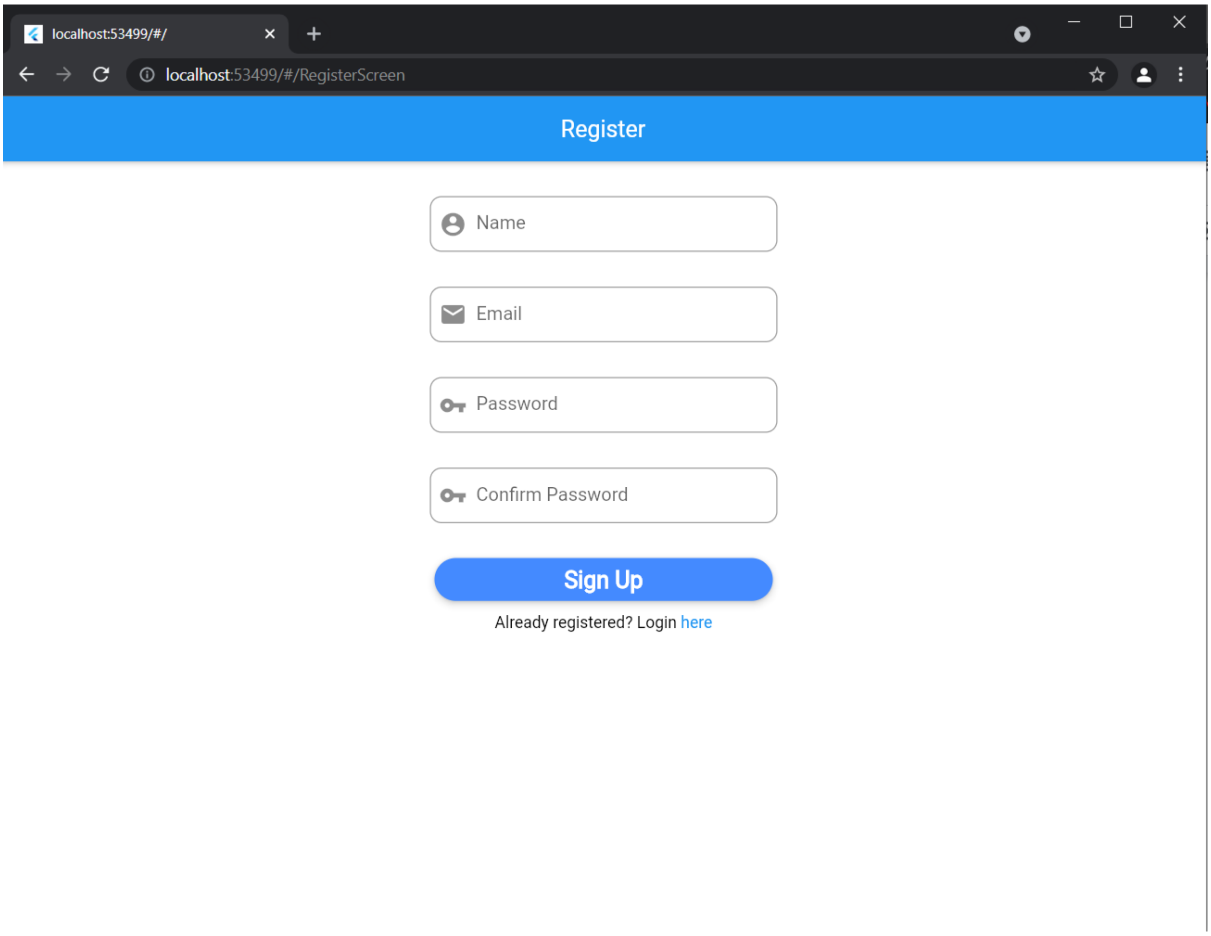
b. Application screens

i. Log In Screen



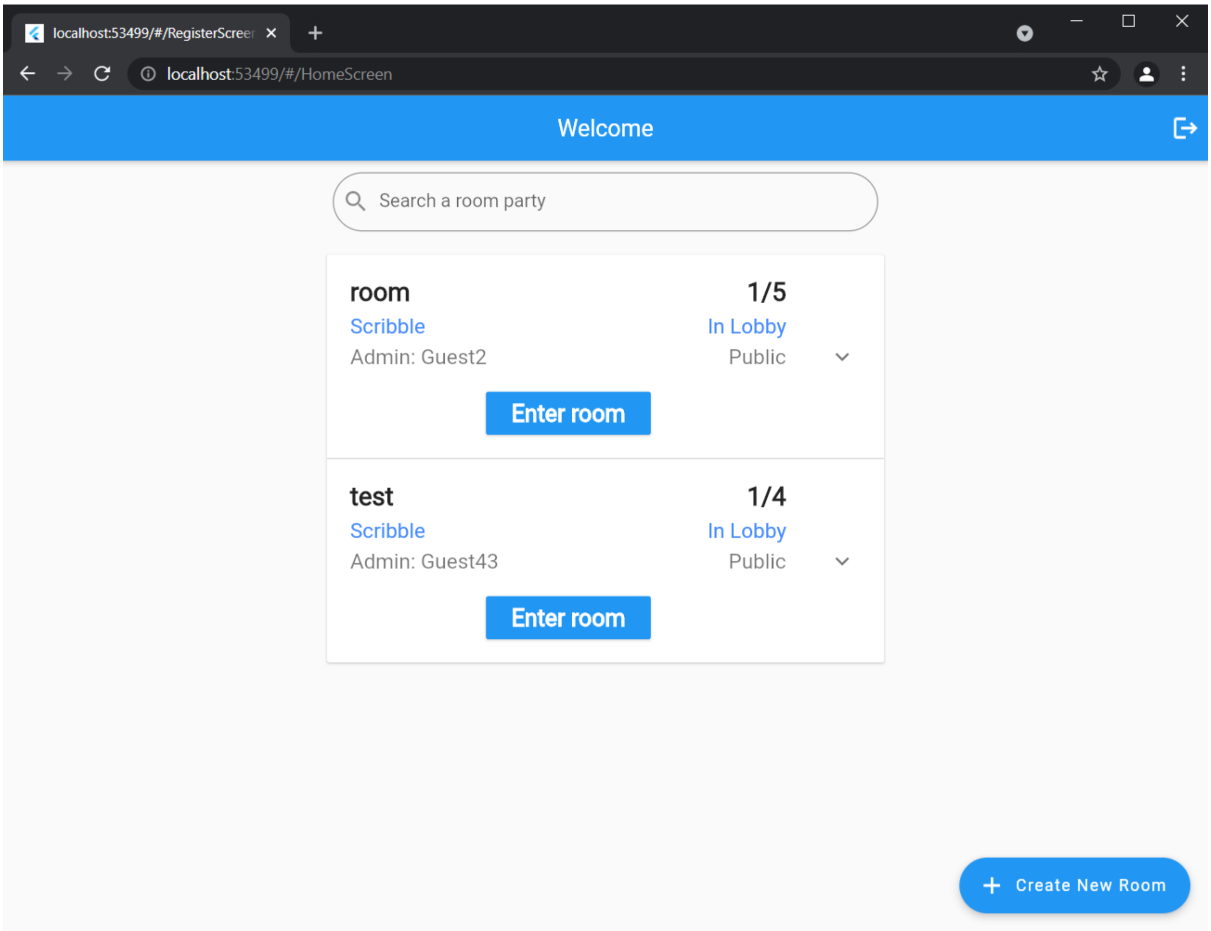
The login screen is the start of every interaction with Party Game Manager. The text fields check for valid emails and for empty values. The possibility to also log in as a guest is available.

ii. Sign Up Screen



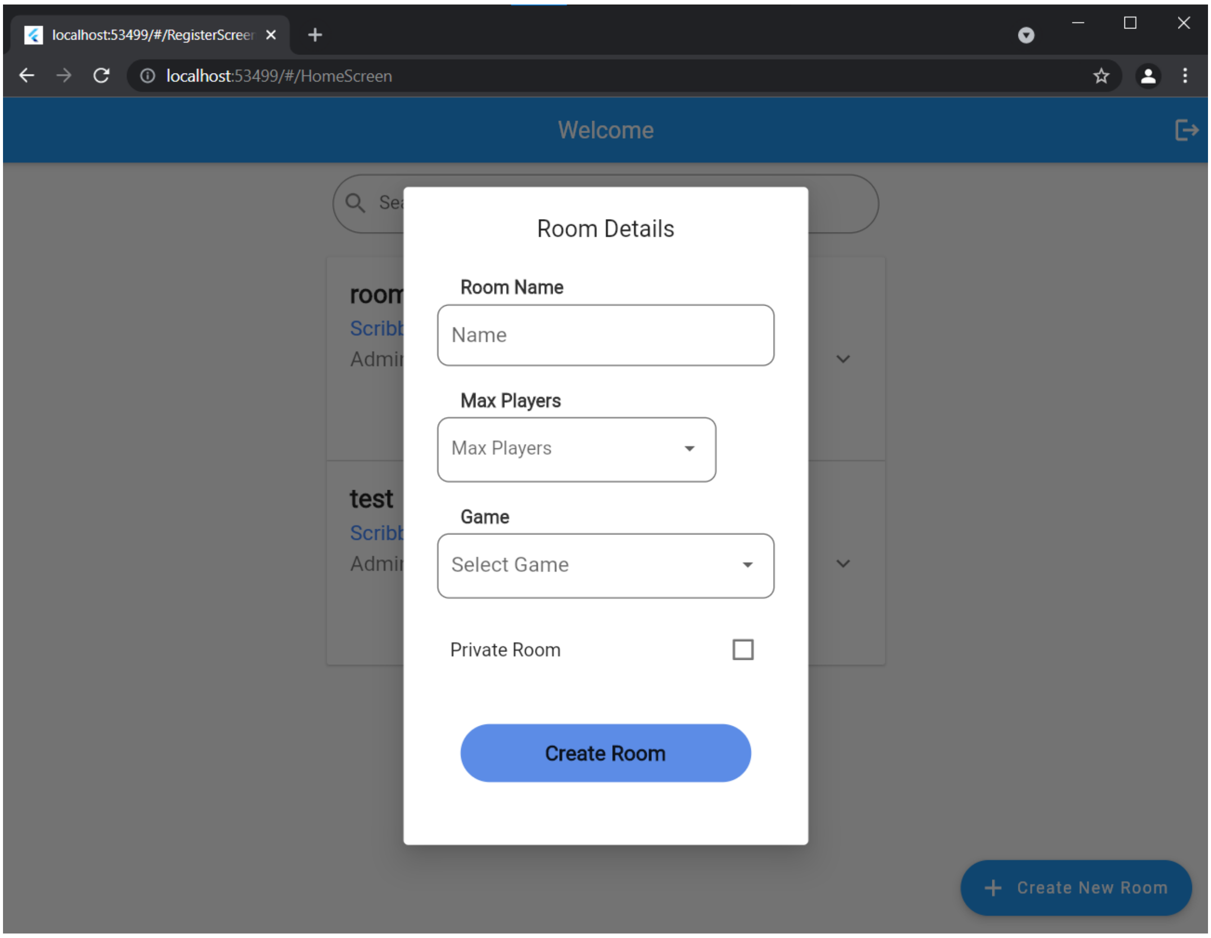
Every user has to go through the sign up screen to get registered in the database of Party Game Manager. Every filed in the page is required and cheks for wiring inputs.

iii. Game Session Browser



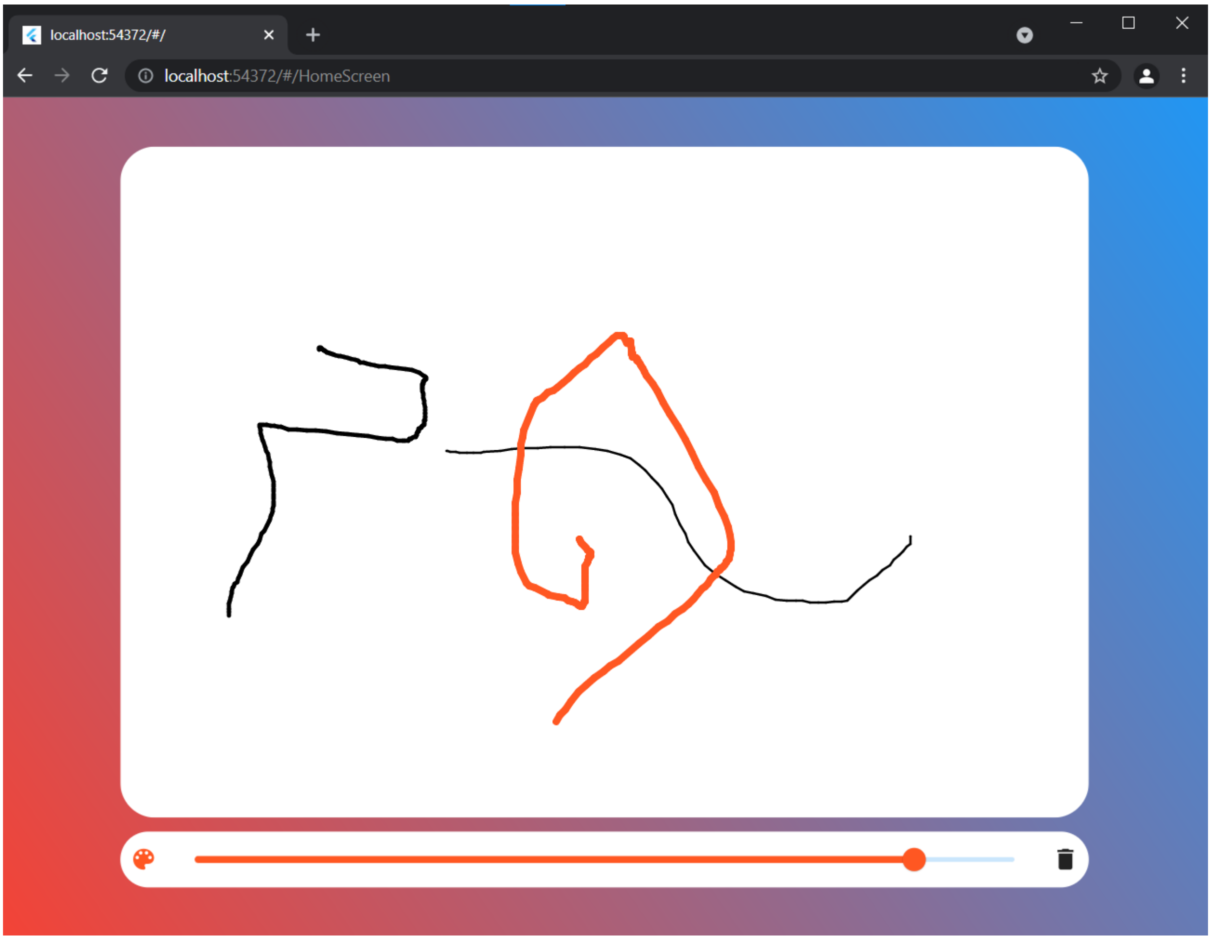
The main screen for every user, and the first screen they will see after the log in. This helps navigate through the available game rooms and shows details about each one.

iv. Admin Dashboard



This screens allows the player to create a new game room that will be instantly available in the game session browser window.

v. Game Session



This screen allows the user to draw on screen, change the color of the pencil and the width of the line.

Testing Environment

c. Manual tests

Every feature is manually tested by the developer before the pull request is created. This is the first step in the testing process.

d. Testing Infrastructure

The testing infrastructure is created using the dart testing capabilities and is deployed using GitHub actions. The testing suite is run on every pull request and every commit to the main branch. All the tests can also be run locally.

e. Test Suite

The test suite includes the following tests:

- Build
- Code format
- Code analysis (static analysis)
- Run flutter tests
- Application deploy
- Cleanup