

Jamk University of Applied Sciences
Course: Android Application Development
Code: TTOW0630

4 x 4

Research assignment
for the course

Mobile Project

Name:	Kornkanok Sangwichien
E-Mail:	N3570@student.jamk.fi
Student number:	N3570
Course of Studies:	Information and Communication Technology

Name:	Anna Paszcza
E-Mail:	N2305@student.jamk.fi
Matriculation Number:	N2305
Course of Studies:	Information and Communication Technology

Name:	Nicole Ebken
E-Mail:	N2309@student.jamk.fi
Matriculation Number:	N2309
Course of Studies:	Information and Communication Technology

Contents

1. Introduction	4
2. Technology and Programs	4
2.1 Android(Kotlin)	4
2.2 Firebase	4
3. Idea	5
4. Library	6
4.1. ReactiveX	6
5. Firebase connection	7
6. Project planning	11
7. Mobile Project	12

Figure Directory

Figure 1	5
Figure 2	6
Figure 3	7
Figure 4	8
Figure 5	9
Figure 6	10
Figure 7	10
Figure 8	11
Figure 9	11
Figure 10	12
Figure 11	13
Figure 12	13
Figure 13	13
Figure 14	14
Figure 15	14
Figure 16	15

1. Introduction

For the research assignment in the course “Android Application Development” we want to create an application - game using Kotlin and Firebase. As a topic we chose Fifteen Puzzle. The result of the research assignment will be continued as a course assignment in TTOW0635 Mobile Project course.

2. Technology and Programs

2.1 Android(Kotlin)

Kotlin is a statically typed programming language that runs on the JVM and is completely interoperable with Java. Kotlin is an officially supported language for developing Android apps, along with Java.

2.2 Firebase

Firebase is a mobile and web application development platform, which provides many services for example: Firebase Analytics, Firebase Storage, Firebase Hosting, Firebase Realtime Database, Firebase Cloud Firestore and many more. We used Firebase Realtime Database for our project to save information about user's scores.

3. Idea

Our game is come up from the fifteen puzzle wooden game which is physically puzzle game for kids to solve the problem by ordering the number from 1-15. Basically, the game has 16 pieces of puzzle which it needed to remove one last piece before playing the game which is number 16. We got an idea to make this game in application instead of physical one and make it more challenge by collecting the score from user then make a high score or rank feature. On rank feature, we would love to learn new technology which made by google. We use Firebase which is real time database to keep the score.



figure 1

4. Library

4.1. ReactiveX

ReactiveX is Java VM implementation RxJava of Reactive Extensions. Reactive Extension(ReactiveX) as a library to compose asynchronous and event-based programs using observable sequences.

Our game considering the score by using play time from user. The highest score is the lowest time that can complete the game. So, we use ReactiveX which has observable function and timer.

How observer work in real life be like this example, the situation in Exam room.

Anna is a student who is taking an exam in the exam room,

Michał also a student who is taking an exam

Michał see something wrong about **Anna**, she was doing something bad in exam room which is breaking the condition of exam room.

Michał was told to watch and report **Anna** as her state changes, and he's to make a **callback** to whoever is listening to him (the teacher)

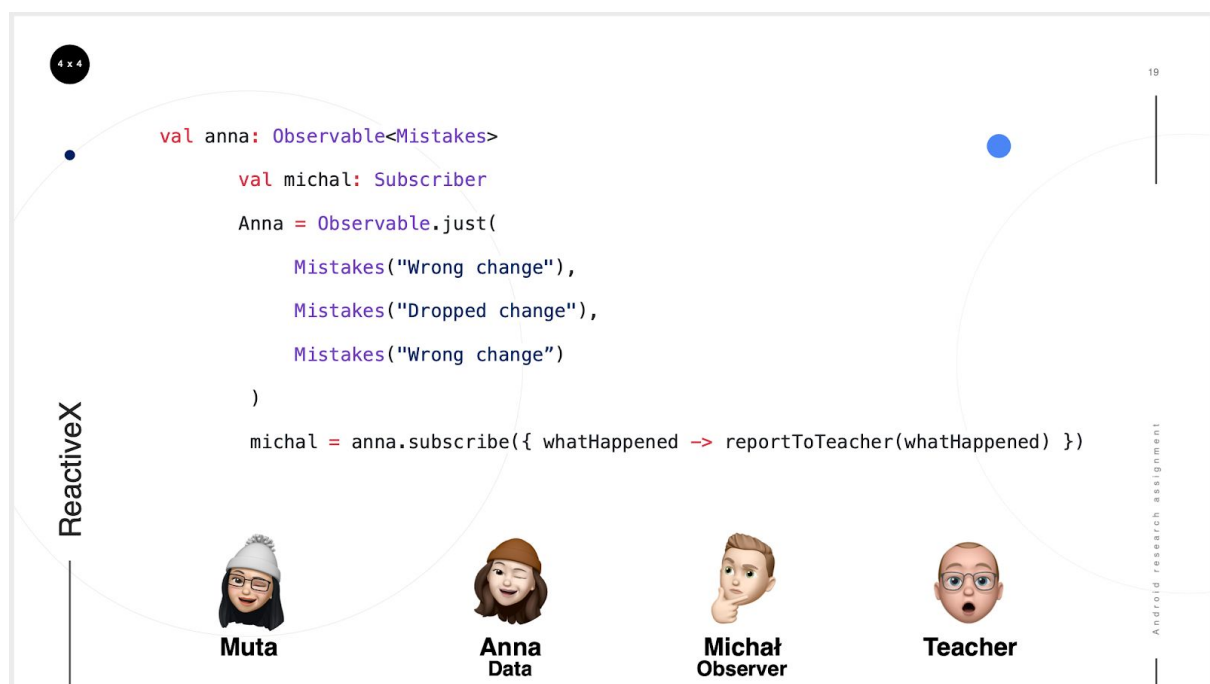


figure 2

5. Firebase connection

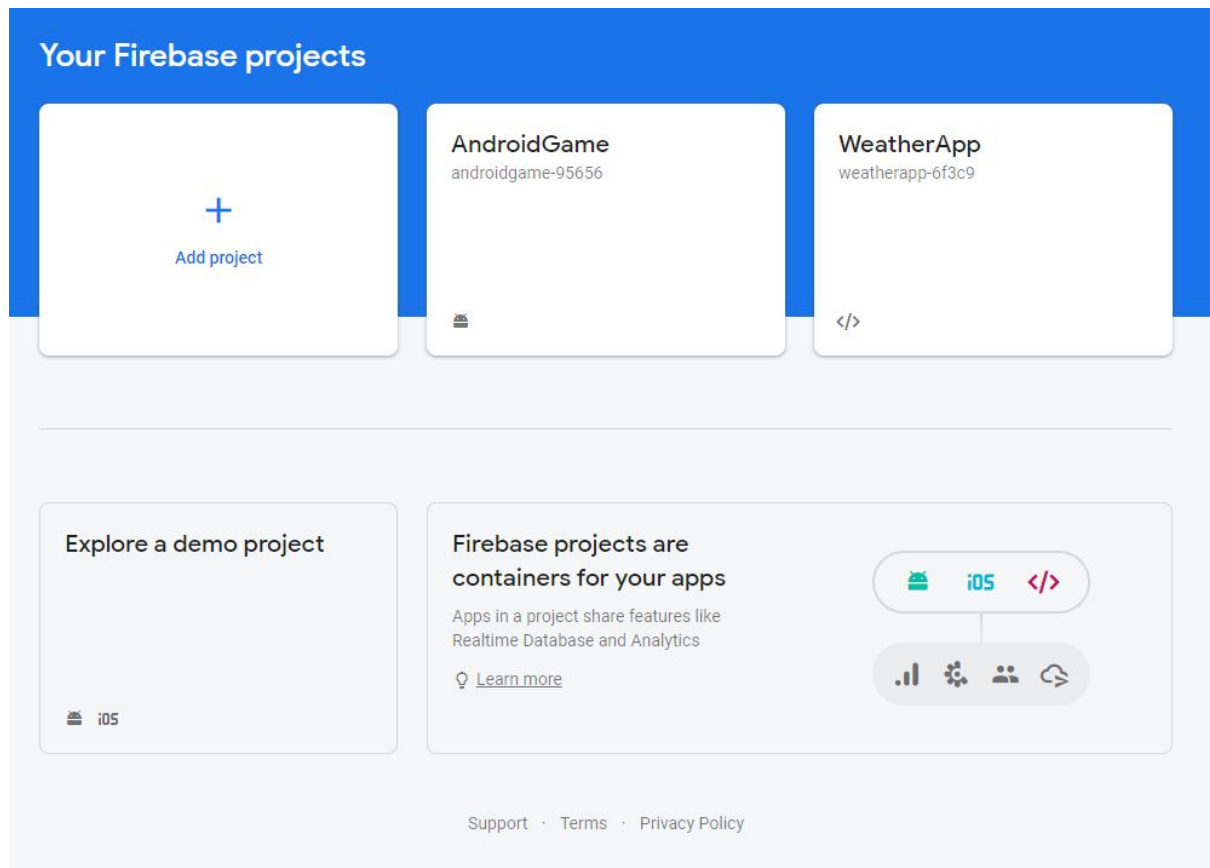



figure 3

We connected our application to Firebase using **Android Studio** and **Firebase website**. Firstly we created a Firebase project on Firebase website then we used Android Studio to connect it to Firebase (**Realtime Database**). There were also other services we could use but they weren't necessary for our project

 **Firebase**

Firebase gives you the tools and infrastructure from Google to help you develop, grow and earn money from your app. [Learn more](#)










- ▶  **Analytics**
Measure user activity and engagement with free, easy, and unlimited analytics. [More info](#)
- ▶  **Cloud Messaging**
Deliver and receive messages and notifications reliably across cloud and device. [More info](#)
- ▶  **Authentication**
Sign in and manage users with ease, accepting emails, Google Sign-In, Facebook and other login providers. [More info](#)
- ▼  **Realtime Database**
Store and sync data in realtime across all connected clients. [More info](#)
[▶ Save and retrieve data](#)
- ▶  **Storage**
Store and retrieve large files like images, audio, and video without writing server-side code. [More info](#)
- ▶  **Remote Config**
Customize and experiment with app behavior using cloud-based configuration parameters. [More info](#)
- ▶  **Test Lab**
Test your apps against a wide range of physical devices hosted in Google's cloud. [More info](#)
- ▶  **App Indexing**
Get your app content into Google Search. [More info](#)
- ▶  **Dynamic Links**
Create web URLs that can be shared to drive app installs and deep-linked into relevant content of your app. [More info](#)

figure 4

Save and retrieve data

Our cloud database stays synced to all connected clients in realtime and remains available when your app goes offline. Data is stored in a JSON tree structure rather than a table, eliminating the need for complex SQL queries.

[Launch in browser](#)

1 Connect your app to Firebase

✓ Connected

2 Add the Realtime Database to your app

✓ Dependencies set up correctly

3 Configure Firebase Database Rules

The Realtime Database provides a declarative rules language that allows you to define how your data should be structured, how it should be indexed, and when your data can be read from and written to. By default, read and write access to your database is restricted so only authenticated users can read or write data. To get started without setting up [Authentication](#), you can [configure your rules for public access](#). This does make your database open to anyone, even people not using your app, so be sure to restrict your database again when you set up authentication.

4 Write to your database

Retrieve an instance of your database using `getInstance()` and reference the location you want to write to.

```
// Write a message to the database
FirebaseDatabase database = FirebaseDatabase.getInstance();
DatabaseReference myRef = database.getReference("message");
```

figure 5

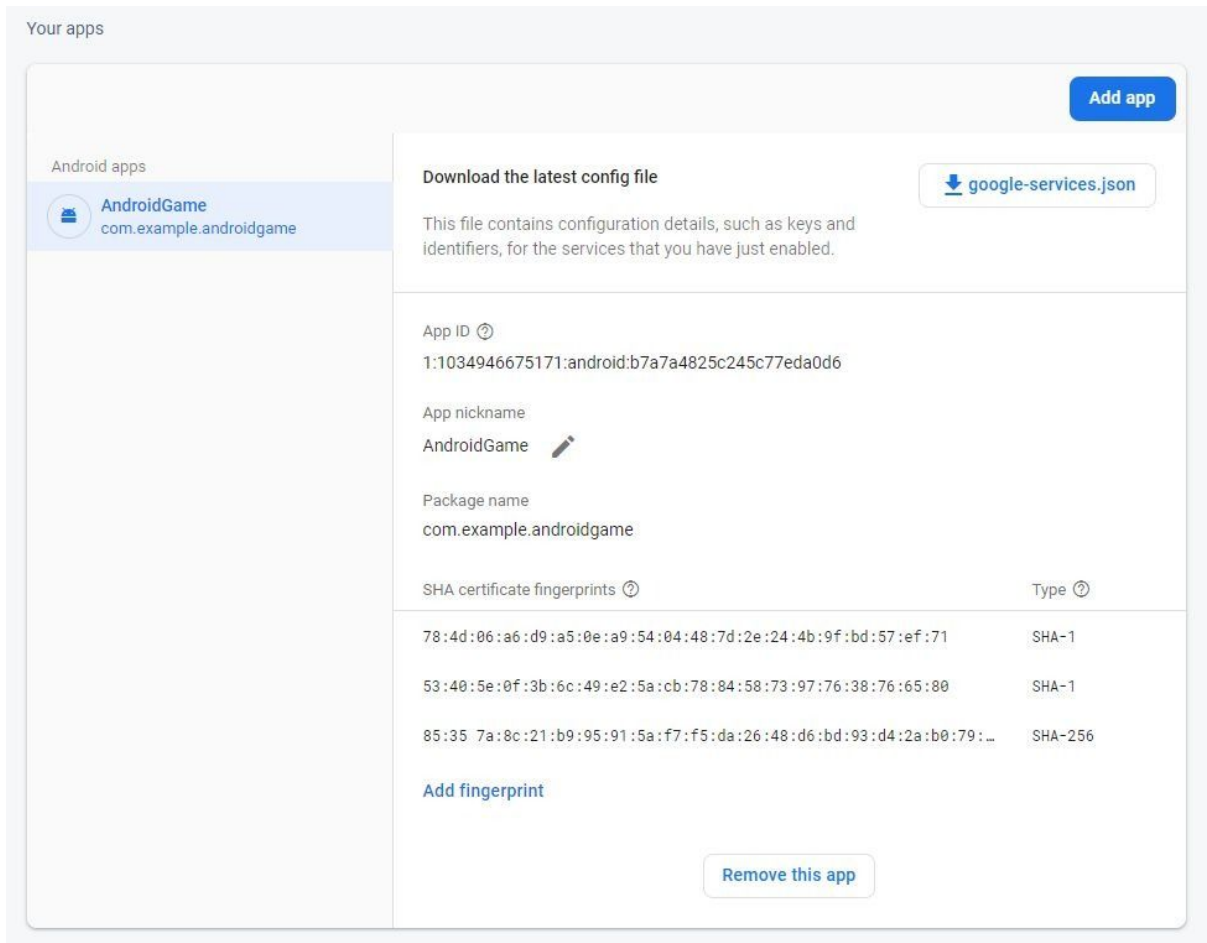


figure 6

After connecting to Firebase we downloaded **google-services.json** file and we put it into the application's folder.

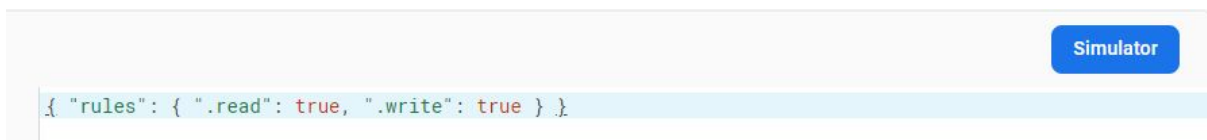


figure 7

The last thing to do was to set **Rules** for the Database on Firebase Website as **'true'** to enable saving to database and reading from it. After changing Rules, we synced our project in Android Studio and Firebase Realtime Database was ready to be used.

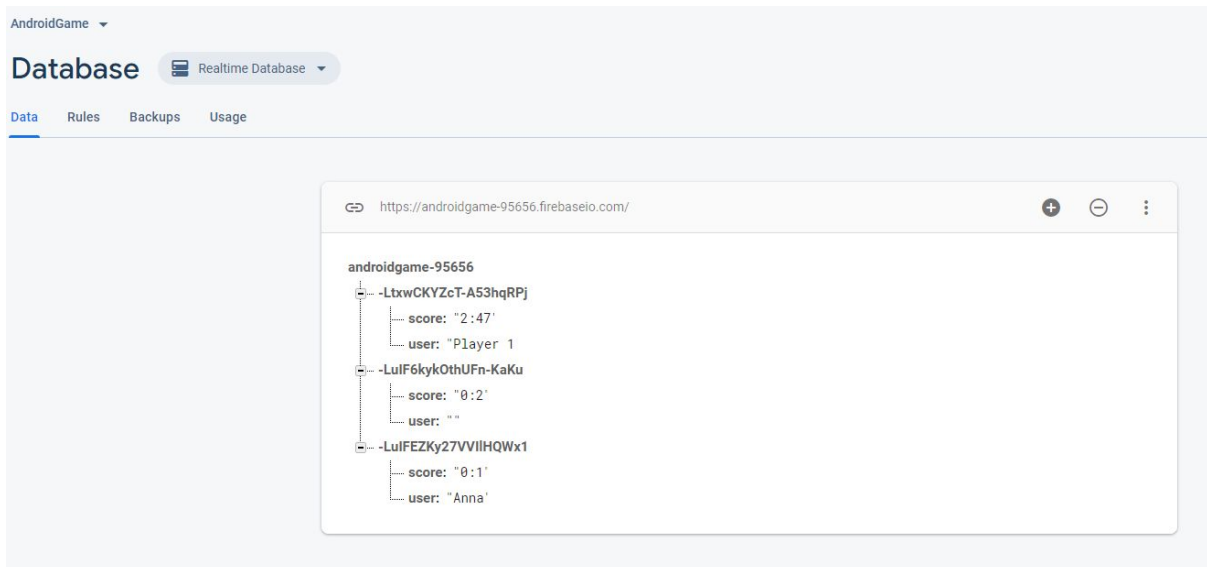


figure 8

Our application works with Firebase Realtime Database and it saves information about user's name and achieved scores. We also read from the database to find the best achieved score and show it on the screen.

6. Project planning

After starting the Project in Android Application Development we planned how to continue it. Our tasks were: Re-designing UI/UX, fixing bugs, adding a menu screen, another game mode, settings screen and possibility to save individual high score.

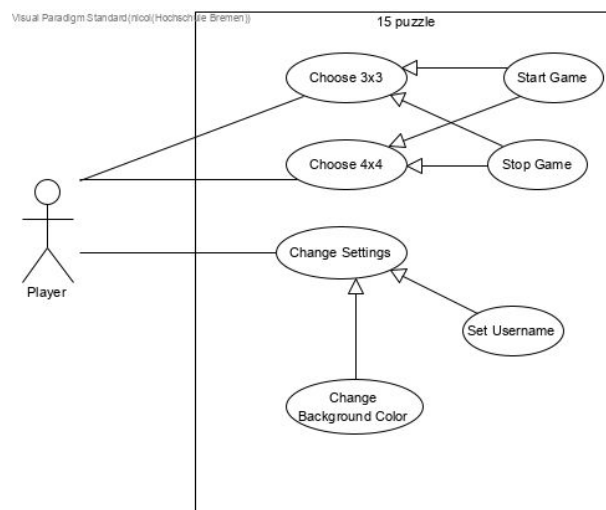


figure 9

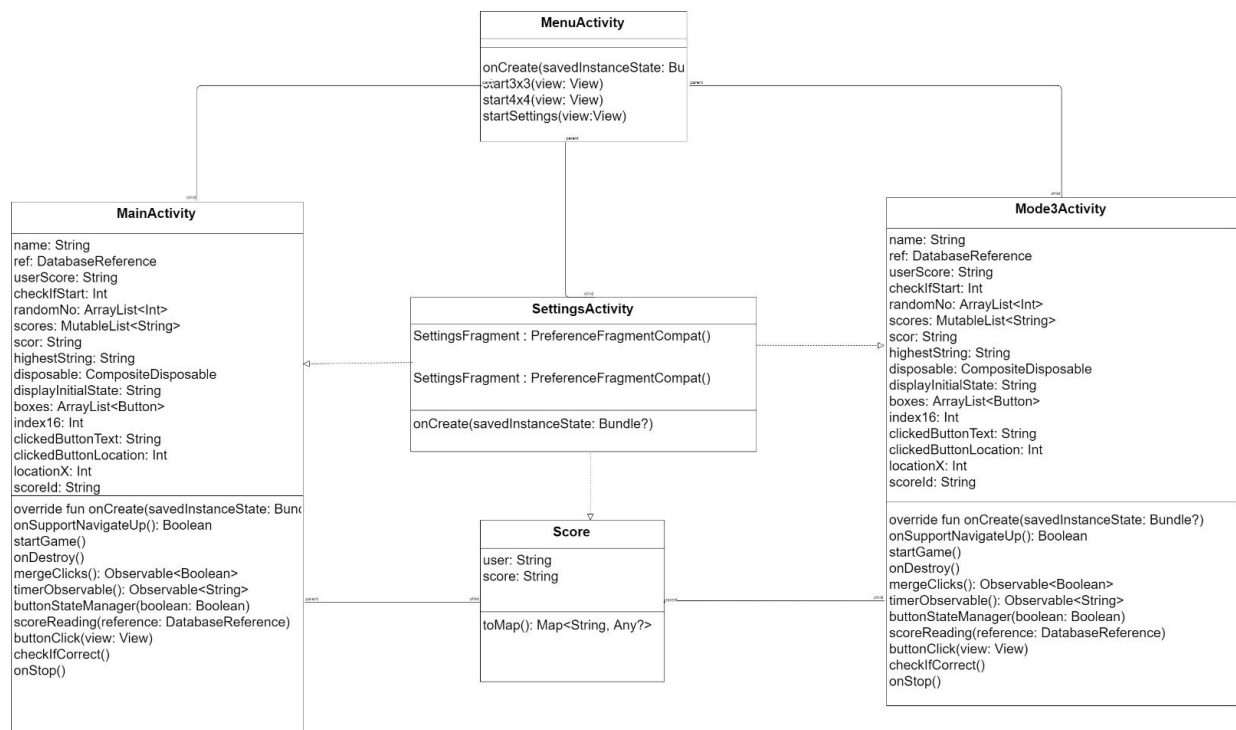


figure 10

In figure 10 you can see our UML diagram. UML is designed to provide a common, semantically and syntactically visual modeling language for the architecture, design, and implementation of complex software systems. Overall, UML diagrams describe the boundaries, structure, and behavior of systems and their objects.

In the project folder Diagrams you can see our workload diagram made with Proggio. It shows the progress that we made and gives inside of our time management.

7. Mobile Project

As mentioned previously, our goal was to have a menu where you can choose the game mode or go to the settings screen. In the settings screen you can add a player and change the background color. We decided to add a landscape with hills and trees to give the game a relaxed vibe.

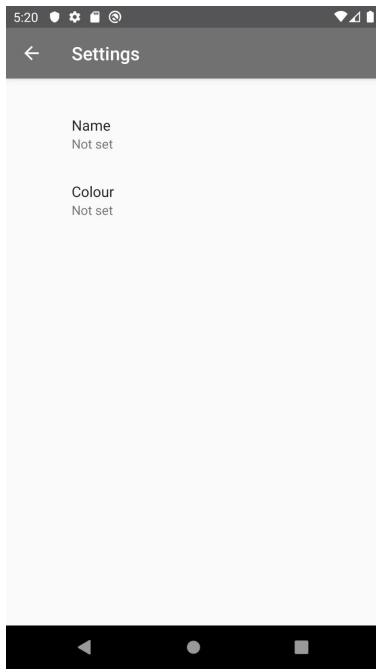


figure 11

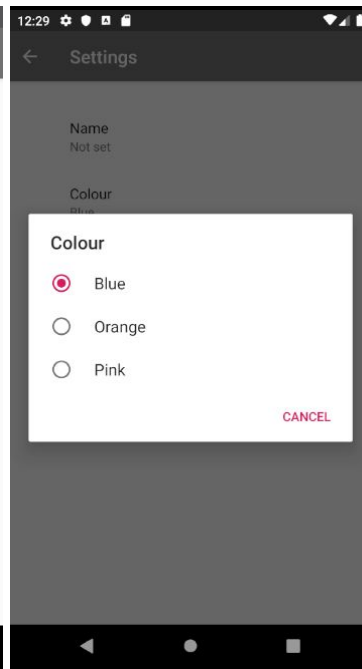


figure 12

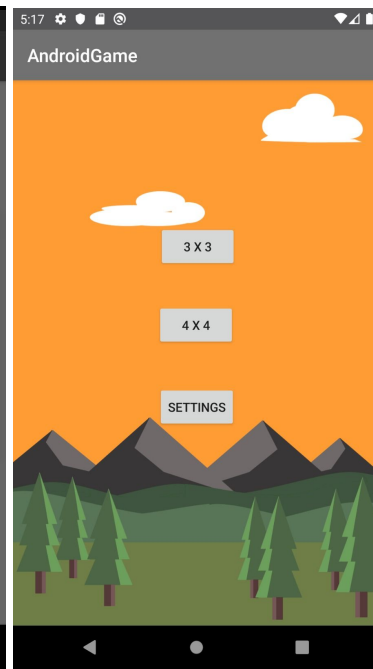


figure 13

The menu screen shows three buttons: 3x3, 4x4 and Settings. 3x3 and 4x4 are two game modes you can choose from. Settings open another screen where you can change the name of the user and the background colour. We put also a small picture of mountains and trees in the menu screen to make it look a little bit more interesting.

The main screen shows the highest achieved score for the user, blocks with numbers on it (game board) and two buttons: START and STOP. After you click START button, one of the blocks (with number 9 or 16) will disappear from the screen and other blocks will be randomly placed on the board. You can stop and reset your game by clicking STOP button. Player's goal in this game is to put all blocks in the correct order (from 1 to 8 or 1 to 15). After it is achieved, the game stops and saves achieved score to Firebase Realtime Database. If the score is better than the best score it will be replaced by better score (shorter achieved time).

Our main goal for this game was achieved, we added another 3x3 mode to our game.

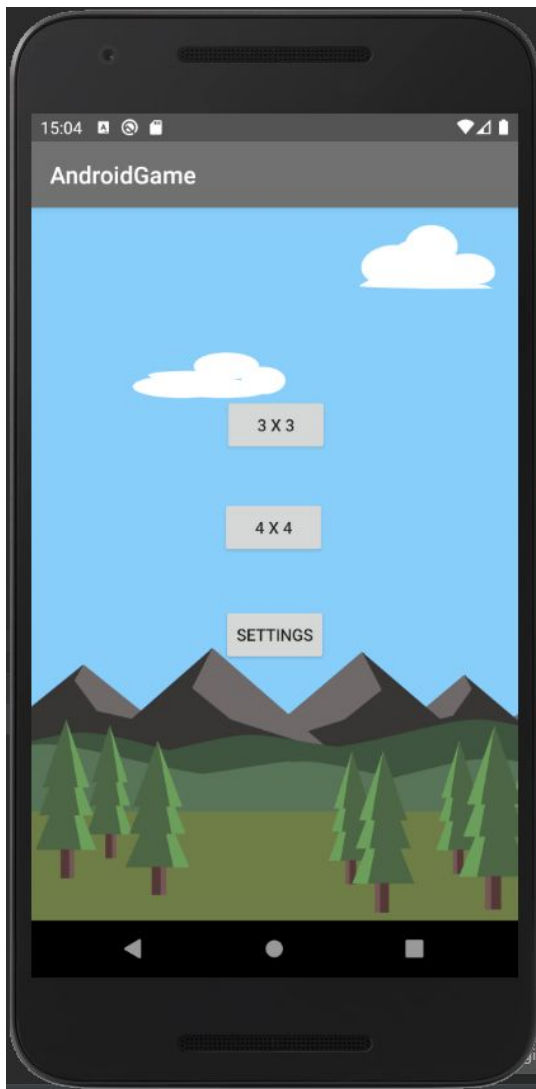


figure 14



figure 15

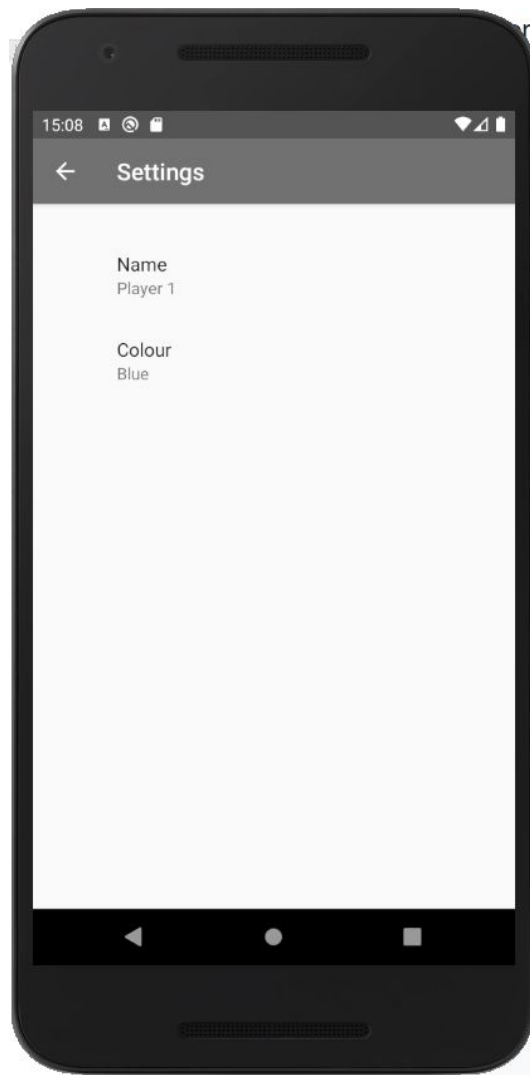


figure 16