

SkillCourt

Students: Edan Rosen

Mentor: Joseph Quinn, Mentor; Guðmundur Traustason, Product Owner

Instructor/Faculty: Dr. Masoud Sadjadi, Florida International University

PROBLEM

Soccer players have trouble keeping track of their game stats. Having another person keep a track of them is difficult and prone to human error. To fix this, SkillCourt was made!

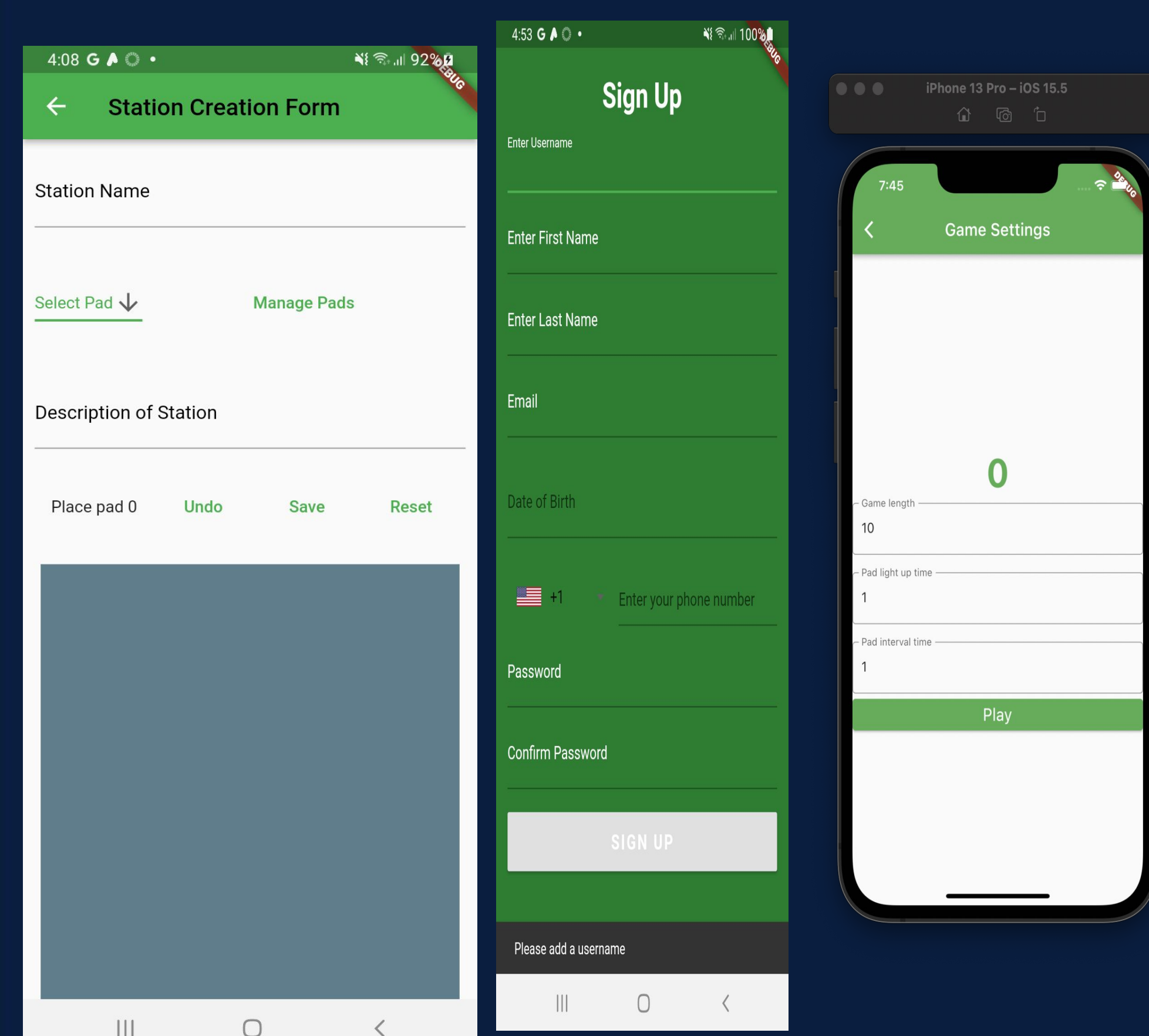
However, we have some current problems:

- There is no working version of the app that is available for both android and iOS.
- There are issues with having the user being able to actually create a game with the pads.
- We are not able to connect the phone application with the website we have set up.

SYSTEM DESIGN

The current system being implemented is in Flutter. In the previous semester, pages related to interacting with the pads and creating games were created in Swift and Kotlin to support native multi-threading, as multi-threading on Flutter was not possible. However, with the recent release of Isolates from Flutter, we were able to continue with Flutter the whole application.

SCREENSHOTS



ACKNOWLEDGEMENT

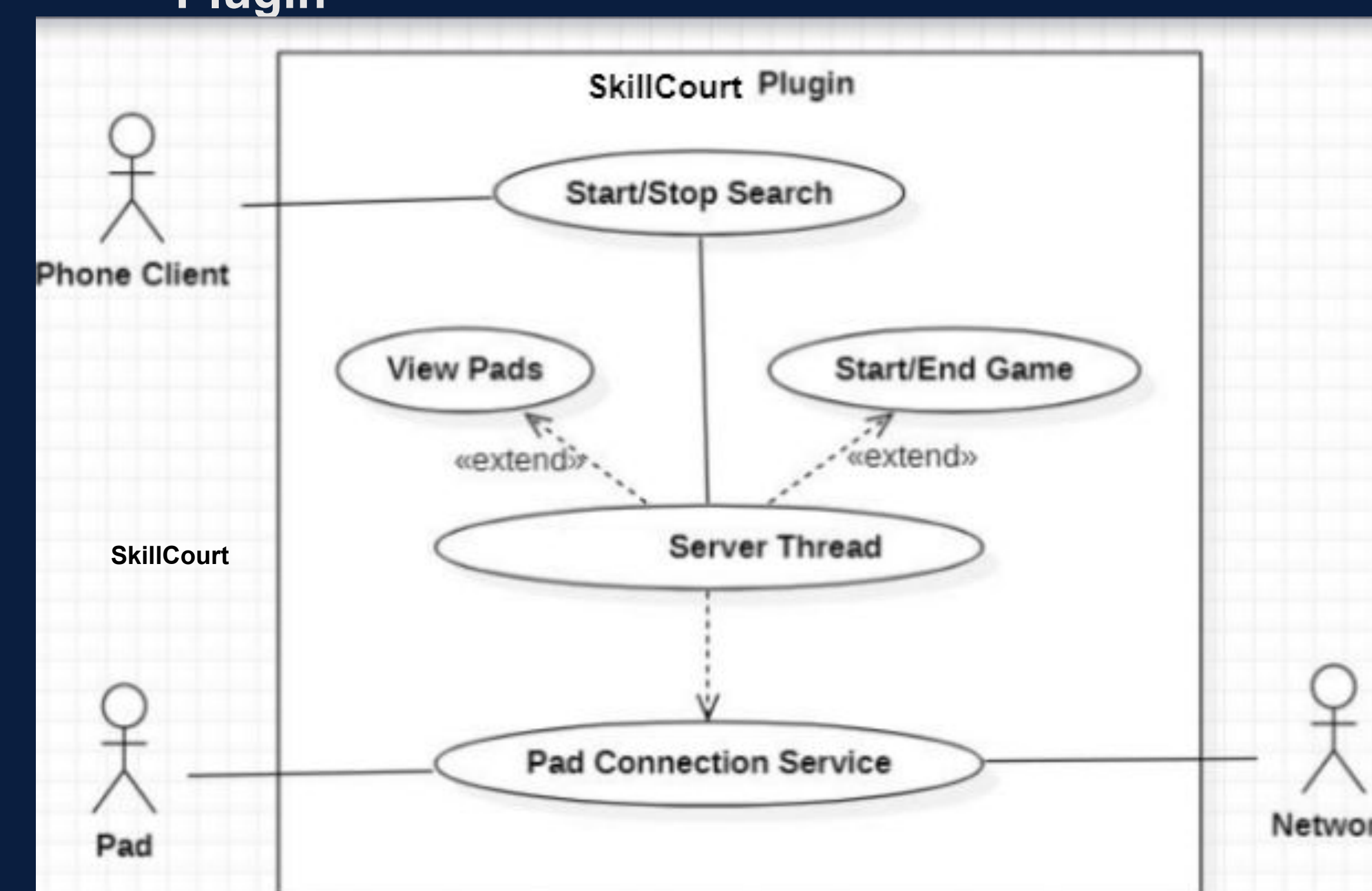
The material presented in this poster is based upon the work supported by Guðmundur Traustason. We thank the product owner and Joseph Quinn for the assistance and mentorship that we received throughout the Capstone 2 Project.

CURRENT SYSTEM

The current SkillCourt app is available for both Android and iOS users, which is being developed in Flutter/Dart. Players can set up LED panels called pads for them to hit with a soccer ball, which connects to the app that connects to those pads as those pads light up when hit. The pads collect stats that are sent to the app based on the hits. The pads also connect to the app and the app tells the pads which pad should light up so the player can hit them.

OBJECT DESIGN

Figure 1: Use Case Diagram for SkillCourt Pad Plugin

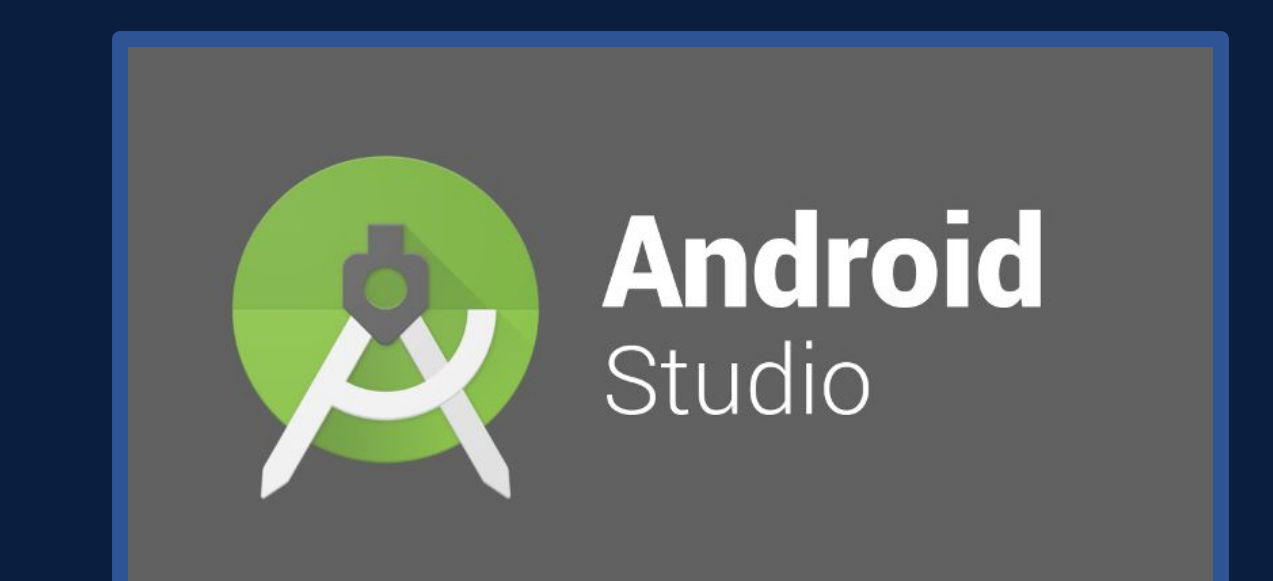


REQUIREMENTS

In order to work on this project, the proper environment is needed. The following is necessary:

- Android Studio (for windows) or XCode (for MacOS)
- git
- git-flow-avh or another git-flow plugin for Git.
- A proper emulator for Android or iOS
- Flutter SDK along with Dart
- Java JRE version 1.8 or greater
- Python 3 (for the pad emulator)

IMPLEMENTATION



SUMMARY

My overall contribution to the project was collaborating with my team member Yasaar Kadery in implementing the algorithm for 'Random' Game. We first had to get the application to connect to emulated pads. Once connected, the player can open a screen which allows for them to customize the amount of seconds. Once we completed this, we had to figure out how to light up the connected pads from the phone application in a pseudo-random, non-configured way. For this to occur, the phone had to designate a pad to light up from the Pad service and then send a code to the pad for it to light up. Pads should stay lit for only the allotted amount of time the user gave. We did this using Flutter Sockets and the combination of Python to directly connect the pads with the emulators to work with their NsdServiceInfo to get the connections at random times. We also had to implement random number generators to decide which pad would be the one to be connected to for a hit, which would be reset once the specific pad has been hit. I also had hoped to work on connected the phone app to the website backend, but due to a lack of time, we were unable to do so.