Florida International University
Knight Foundation School of Computing and Information Sciences

Software Engineering Focus

Final Deliverable

Project Title: SkillCourt

Team Members: *Johan Balceiro. Johnny Casares, Luis Chavez*

Product Owner(s): *Gummi Oxndal*

Instructor: *Masoud Sadjadi*

The MIT License (MIT)
Copyright (c) *2016 Florida International University*

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Abstract

This document offers a comprehensive overview of the SkillCourt mobile and web applications, shedding light on their design and implementation. The SkillCourt mobile app, crafted with Dart through Flutter, serves as a valuable resource for soccer players, facilitating targeted training to enhance their skills for the game. The application not only aids players in identifying areas for skill improvement but also contributes to refining specific soccer-related techniques. In the technical realm, the firmware, currently operating on Python on a Raspberry Pi, is undergoing a transition to Rust. Additionally, the sensor pads are being upgraded to incorporate ESP modules, showcasing ongoing technological advancements for an optimized user experience in soccer training.

Table of Contents

Introduction
……..…..6
Current System
……..……..6

User Stories
Implemented User Stories ……..……..7
Pending User Stories
……...……..….. 12

Project Plan
Hardware and Software Resources ……..…
12
Sprints Plan
……...……
……. 13
Sprint 1
……
……... 14
Sprint 2
……
……... 14
Sprint 3
……
……... 14
Sprint 4
……
……... 15
Sprint 5
……
……... 16
Sprint 6
……
……... 17

System Design
Architectural Patterns
…….. 18
System and Subsystem Decomposition
……..… 19

Design Patterns
…….…
.... 20

System Validation
…….
……. 23

Glossary
…….
……. 37

Appendix
…….
……. 23

Appendix A - UML Diagrams
……. 23

Static UML Diagrams
……38

Dynamic UML Diagrams
…….. 40

Appendix B - User Interface Design
…….…... 32

Appendix C - Sprint Review Reports
……...…… 32

Appendix D - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents
…….…… 37

References
…….
……...……... 41

Introduction

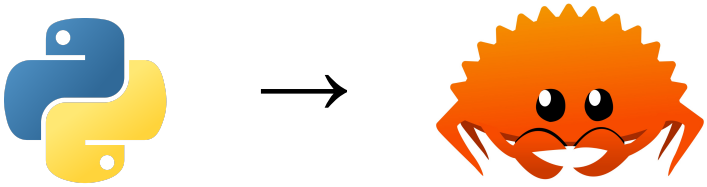
SkillCourt is an environment where players can kick, toss, or throw a ball at a sensor pad. The pad collects data from being hit and once the game concludes the mobile app records statistics based on the player's performance. The phone app also controls parameters of the game, such as game length. During

the Fall semester of 2023, our work was centered around advancing the UI features, the development of a pad emulator, and the transitioning of the communication protocol from mDNS to WebSockets.

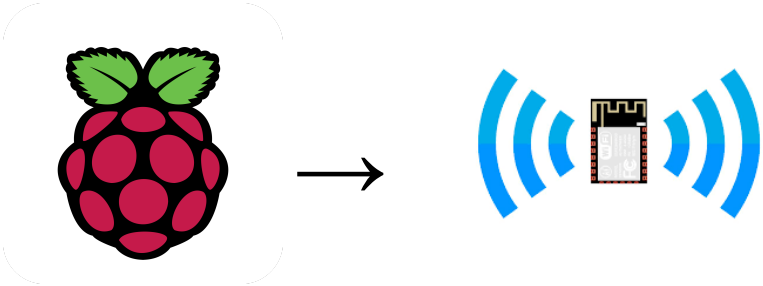
Current System

The mobile app uses Flutter, Dart, SQLite, and Git.

The pad firmware runs on Python, but it is expected to change to Rust.



The pad hardware currently uses Raspberry Pi but it is expected to transition to ESP module.



User Stories

The following section provides the detailed user stories that were implemented in this iteration of the SkillCourt project. These user stories served as the basis for the implementation of the project's features. This section also shows the user stories that are to be considered for future development.

Implemented User Stories

User Story <1. Research Communication Protocols>
Acceptance Criteria − Research and document findings on the communication protocols discussed, such as web sockets and mDNS. Research alternatives that might be applicable.

User Story <2. Run SkillCourt Mobile Application>
Acceptance Criteria − Download, setup, and test application to understand its mechanics.

User Story <3. Current app UI vs Figma design>
Acceptance Criteria − Check Figma and compare it to the current app. Identify aspects to be implemented.

User Story <4 WebSocket connection implementation>
Acceptance Criteria − Establish a simple connection that allows basic communication between app and a server

User Story <5 UI Feature Implementation Epic: Part 1>

Acceptance Criteria − Brightness selection − Play Timer − Play Setup Screen

User Story <6 Singleton Implementation >
Acceptance Criteria − Research dart usage of singletons − Research packages that facilitate the process

User Story <7. WebSocket client-side implementation>
Acceptance Criteria − In app, dart implementation of websockets

User Story <8. UI Feature Implementation Epic: Part 2>
Acceptance Criteria − Play Setup Screen − Saved Screen − Brightness selection

User Story <9. Singleton Implementation >
Acceptance Criteria - Use “get_it” package

User Story <10 Merge>
Acceptance Criteria - Merge branches for all changes and resolve any issues from that

User Story <11. mDNS to Websocket>

Acceptance Criteria − Resolve mDNS to Websocket migration issues

User Story <12. UI Implementation Epic: Part 3 >
Acceptance Criteria − Continue implementing and testing the new screens: Play Set Up, Play Timer, and Color selection

User Story <13. Research Pad Emulators >
Acceptance Criteria: - Evaluate current pad emulator in python - Design and develop a pad emulator that behaves closely to the hardware

User Story <14. Research Flutter>
Acceptance Criteria - Search and learn about the best dart/ flutter development practices to enhance the codebase

User Story <15. Research Rust Implementation of Websocket >
Acceptance Criteria - Document finding that may be useful for the implementation of the firmware in Rust

User Story <16. Setup IoT Hub Azure >
Acceptance Criteria - Setup IoT Hub Azure

User Story <17. UI Implementation Epic: Part 4 >
Acceptance Criteria - Continue implementing and testing the new screens: Play Set Up, Play Timer, and Color selection

User Story <18. Test Pad Features within IoT emulator >
Acceptance Criteria - Emulator is able to receive a persistent connection between client and server over socket. Emulator feature responds when called.

User Story <19. Merge all changes into a working branch>
Acceptance Criteria − Merge all changes and ensure app works properly

User Story <20. Research Rust Firmware for Raspberry Pi >
Acceptance Criteria − Research possible Rust features and packages to use in the future firmware.

User Story <21. Continue setup of IoT Azure emulator for future use >
Acceptance Criteria − Setup a more detailed version of the device and implement as many testable features as feasible. Implement IAM options for user access on Azure.

User Story <22. UI Implementation Epic: Final>

Acceptance Criteria

- Finalize UI epic
- Advance as much as possible the missing features of the Play Set Up page and Play Timer.

User Story 23< Update firmware>
Acceptance Criteria - Change the firmware of the actual pads so that it supports the Websocket connection

User Story 24< Merge all changes into a working branch>
Acceptance Criteria - Final merge

Pending User Stories

User Story <25. UI Implementation>

Acceptance Criteria:
- Play Timer: this page still requires a UI element that displays a number of hits. Also, pad connection is required for this part
- Welcome: this page needs to be developed in order to allow user to connect to available pads
  - Pads Connected: This screen still needs to be developed. The screen checks on the status of the pads and allows user to add more pads.
- Play Set Up:
  - In saved section, show bottom modal sheet to delete all games
  - Pad sequence parameter requires UI that allows user to pick order of the pads

User Story <26 Complete testing environment for pad>
Acceptance Criteria − IoT Hub environment still suffers from many limitations and needs a lot of research to reach its potential usefulness for future development. − Newly implemented UI features need to be assimilated into pad firmware

PROJECT PLAN

Hardware and Software Resources

The following is a list of all hardware and software resources that were used in this project:

Application

- Java JRE > 1.8
- GitHub
- Git
- Flutter SDK
- A phone emulator (eg Android Studio/XCode)
- Python3
- VSCode (optional)

Hardware

-A physical Pad
-Mobile phone (iOS or Android)
-Windows or Mac computer

Sprint Planning Meeting Minutes:

Sprint 1

Attendees: Johan Balceiro, Johnny Casares, Luis Chavez, Gummi Oxndal
Start time: 12: 00pm
End time: 12: 30 pm

After discussion, the velocity of the team was estimated to be 10, we will endeavor to meet this approximation.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority. For now, the product owner has decided to focus the team's efforts into research, future sprints will focus on development.

- User Story #1 (5 points): Research and document findings on the communication protocols discussed, such as web sockets and mdns. Research alternatives that might be applicable.
- User Story #2 (2.5 points): Download, setup, and test application to understand its mechanics.
- User Story #3 (2.5 points): Check figma and compare it to the current app. Identify aspects to be implemented.

The team members indicated their willingness to work on the following user stories.

- Johnny Casares
  - User Story #1
  - User Story #2
  - User Story #3
- Johan Balceiro
  - User Story #1
  - User Story #2
  - User Story #3
- Luis Chavez
  - User Story #1
  - User Story #2
  - User Story #3

Sprint 2

Attendees: Johan Balceiro, Johnny Casares, Luis Chavez, Gummi Oxndal, Joseph Quinn
Start time: 12:00 pm
End time: 12:59 pm

After discussion, the velocity of the team was estimated to be 11.25 assuming all user stories all completed this sprint.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

- User Story # 1: WebSocket connection implementation (5)
- User Story # 2: UI Feature Implementation Epic: Part 1 (5)
  - Play Timer
  - Play Setup Screen
- User Story # 3: Singleton Implementation (2.5)

The team members indicated their willingness to work on the following user stories.

- Johnny Casares
  - User Story #2
  - User Story #3
- Johan Balceiro
  - User Story #1
  - User Story #3
- Luis Chavez
  - User Story #2
  - User Story #3

Sprint 3

Attendees: Johan Balceiro, Johnny Casares, Luis Chavez, Gummi Oxndal
Start time: 9:00 pm
End time: 9:30 pm

After discussion, the velocity of the team was estimated to be 11.25 assuming all user stories all completed this sprint.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

- User Story # 1: WebSocket client-side implementation (5)
- User Story # 2: UI Feature Implementation Epic: Part 2 (5)
  - Play Setup Screen
  - Saved Screen
- User Story # 3: Singleton Implementation (1)
- User Story # 4: Merge branches for all changes and resolve any issues from that (2.5)

The team members indicated their willingness to work on the following user stories.

- Johnny Casares
  - User Story #2
  - User Story #3
  - User Story #4
- Johan Balceiro
  - User Story #1
  - User Story #3
  - User Story #4
- Luis Chavez
  - User Story # 2
  - User Story # 3
  - User Story # 4

Sprint 4

Attendees: Johan Balceiro, Johnny Casares, Luis Chavez, Gummi Oxndal
Start time: 9:00 pm
End time: 9:30 pm

After discussion, the velocity of the team was estimated to be 12 assuming all user stories all completed this sprint.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

- User Story #1: Resolving mDNS to Websocket migration issues (5)
- User Story #2: UI Implementation Epic: Part 3 (5)
- User Story #3: Researching Pad Emulators (1)
- User Story #4: Research Flutter app development further (1)
- User Story #5: Research Rust Implementation of Websocket (1)

The team members indicated their willingness to work on the following user stories.

- Johnny Casares
  - User Story # 2
  - User Story # 4
  - User Story # 5

- Johan Balceiro
  - User Story # 1
  - User Story # 3
  - User Story # 5

- Luis Chavez
  - User Story # 2
  - User Story # 4

Sprint 5

Attendees: Johan Balceiro, Johnny Casares, Luis Chavez, Gummi Oxndal
Start time: 9:00 pm
End time: 9:30 pm

After discussion, the velocity of the team was estimated to be 12 assuming all user stories all completed this sprint.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

- User Story #1: Setup IoT Hub Azure (5)
- User Story #2: UI Implementation Epic: Part 4 (5)
- User Story #3: Test Pad Features within IoT emulator (1)
- User Story #4: Merge all changes into a working branch (2)
- User Story #5: Research Rust Firmware for Raspberry Pi (1)

The team members indicated their willingness to work on the following user stories.

- Johnny Casares
  - User Story # 2

- User Story # 4
- User Story # 5

- Johan Balceiro
  - User Story # 1
  - User Story # 3
  - User Story # 5
  - User Story # 4

- Luis Chavez
  - User Story # 2
  - User Story # 4

Sprint 6

Attendees: Johan Balceiro, Johnny Casares, Luis Chavez, Gummi Oxndal
Start time: 9:00 pm
End time: 9:20 pm

After discussion, the velocity of the team was estimated to be 12 assuming all user stories all completed this sprint.

The product owner chose the following user stories to be done during the next sprint. They are ordered based on their priority.

- User Story #1: Continue setup of IoT Azure emulator for future use (2)
- User Story #2: UI Implementation Epic: Final (5)
- User Story #3: Change the firmware of the actual pads so that it supports the Websocket connection (5)
- User Story #4: Merge all changes into a working branch (1)

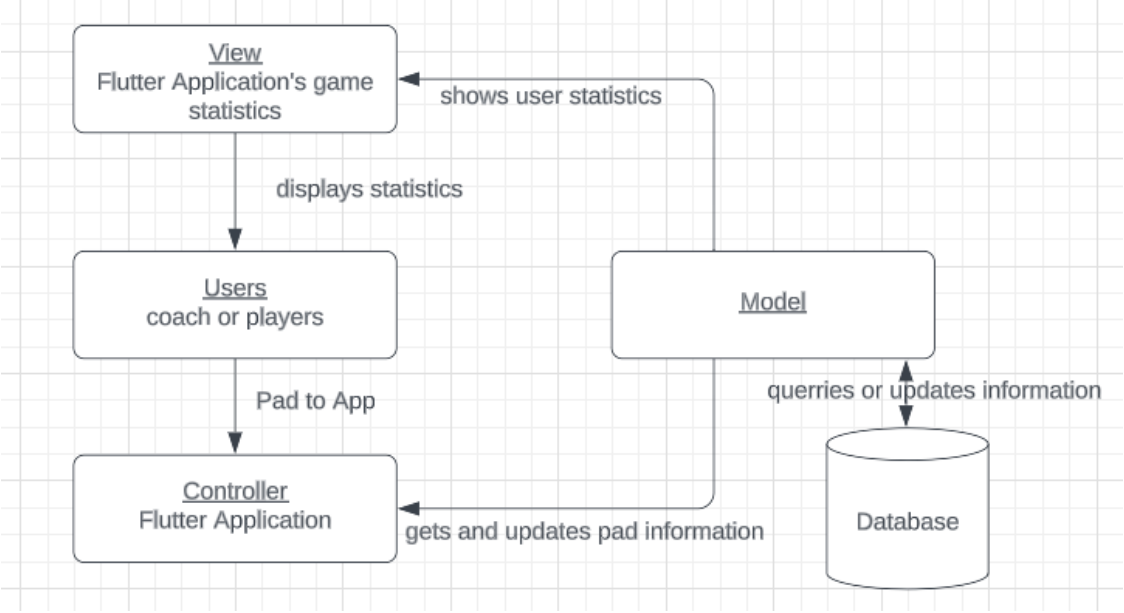
The team members indicated their willingness to work on the following user stories.

- Johnny Casares
  - User Story # 2
  - User Story # 4
- Johan Balceiro
  - User Story # 1
  - User Story # 3

- ➢ User Story # 4
- ❖ Luis Chavez
  - ➢ User Story # 2
  - ➢ User Story # 4

SYSTEM DESIGN

Architectural Patterns

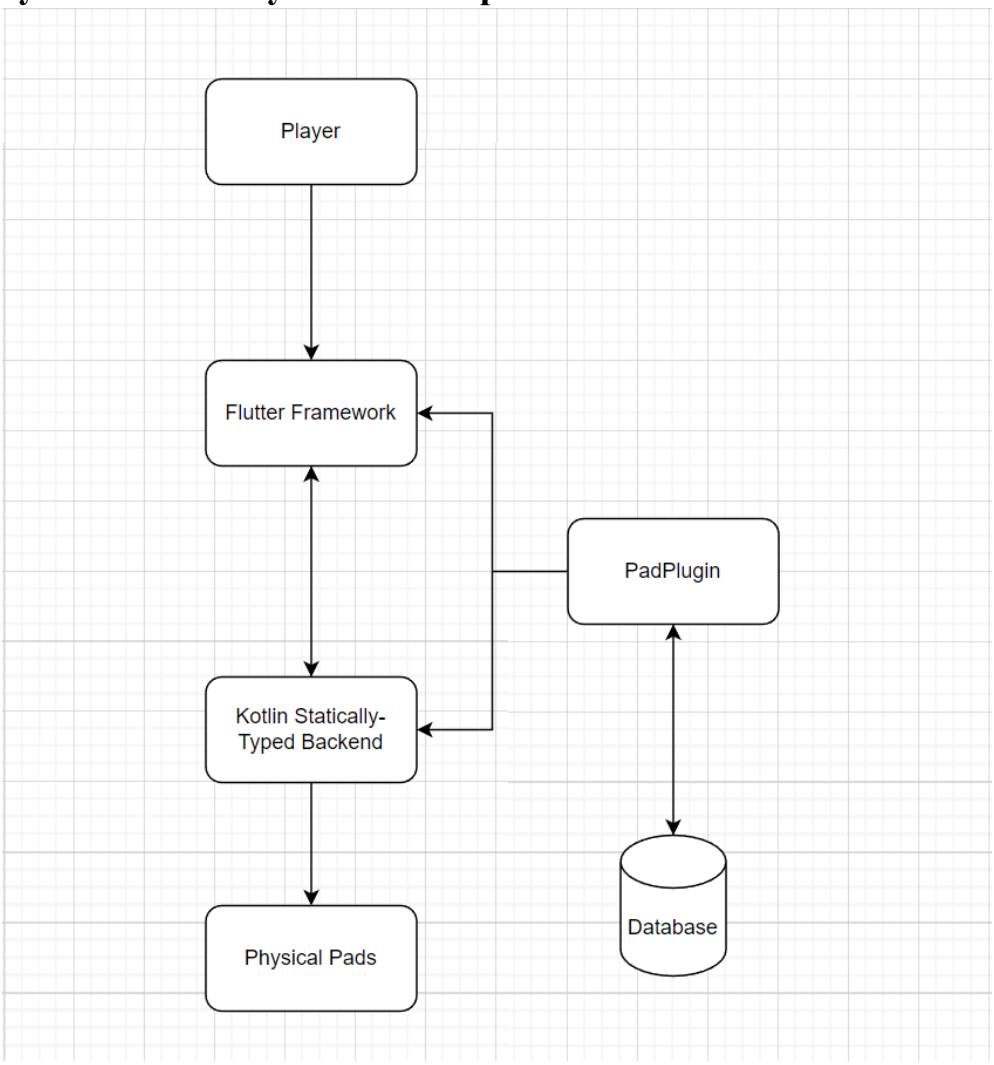


(Source: previous capstone 2 final documentation)

The diagram above was based on the previous team's diagram for the implementation of the skillcourt database design.

The image depicts the Model-View-Controller (MVC) architecture, where a controller is a Flutter app utilized by soccer players and coaches. The view represents the display of different game statistics for a player within the Flutter app. The model serves as the underlying application responsible for running game logic while the pad is connected, and it updates the database to store game statistics and information.

System and Subsystem Decomposition

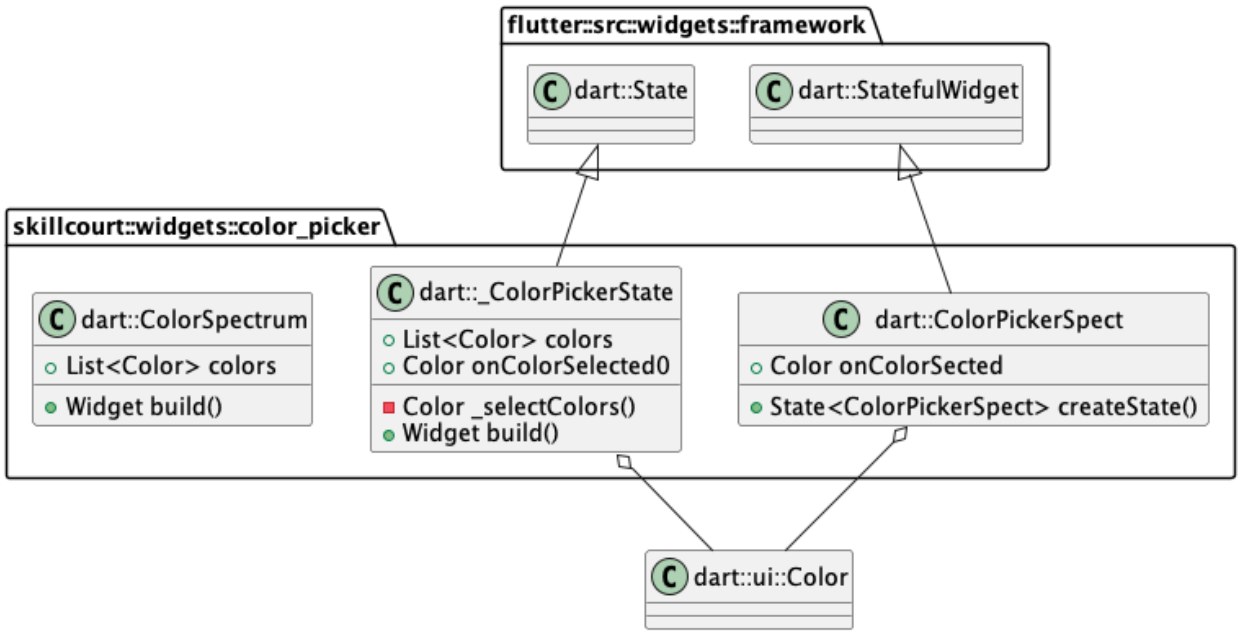


(Source: previous capstone 2 final documentation)

The diagram displayed above exhibits the AgineekPadPlugin that has been disassembled. However, this is not the final outcome of this team or the previous team. The AgineekPadPlugin system should possess many other useful features which can be beneficial to soccer players and coaches. For instance, each coach/player can be permitted to create a personalized account and keep a track of their game progress. If these specifications are fulfilled, the dissection of the system will resemble the figure presented below. In the background, Kotlin tracks the number of hits and misses in the game and passes this log to the Flutter

front end. The Flutter frontend displays this new log, passes it to the database interface and saves it to the database.

Design Patterns



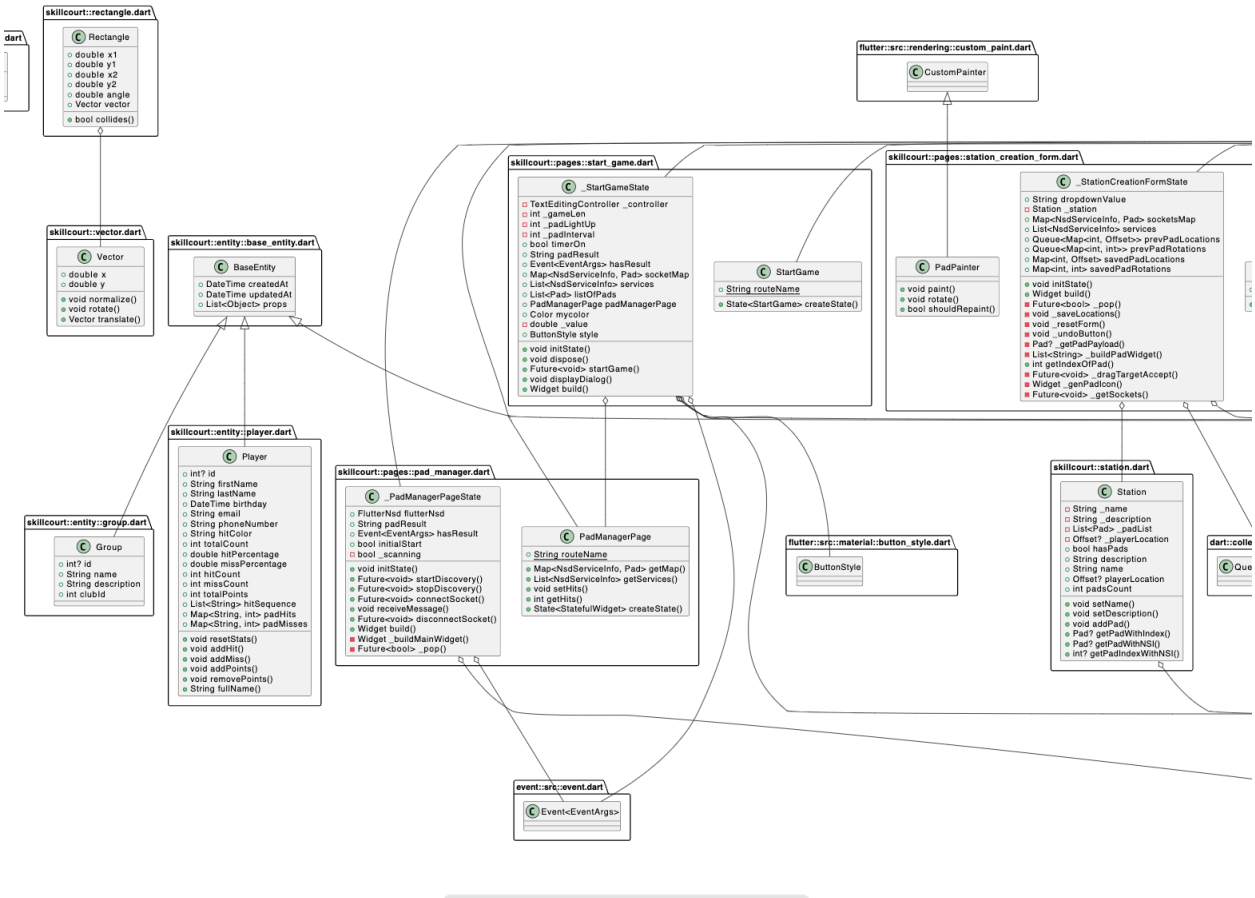
Color Picker Spectrum UML Diagram

The UML diagram above shows the structure of the custom color picker class. This feature was added this semester. The ColorSpectrum class draws the color spectrum with all its colors. The ColorPicker class has the onColorSelected function that does all the math to calculate which of the 5 colors in the list the user is closest to when dragging across the color spectrum.

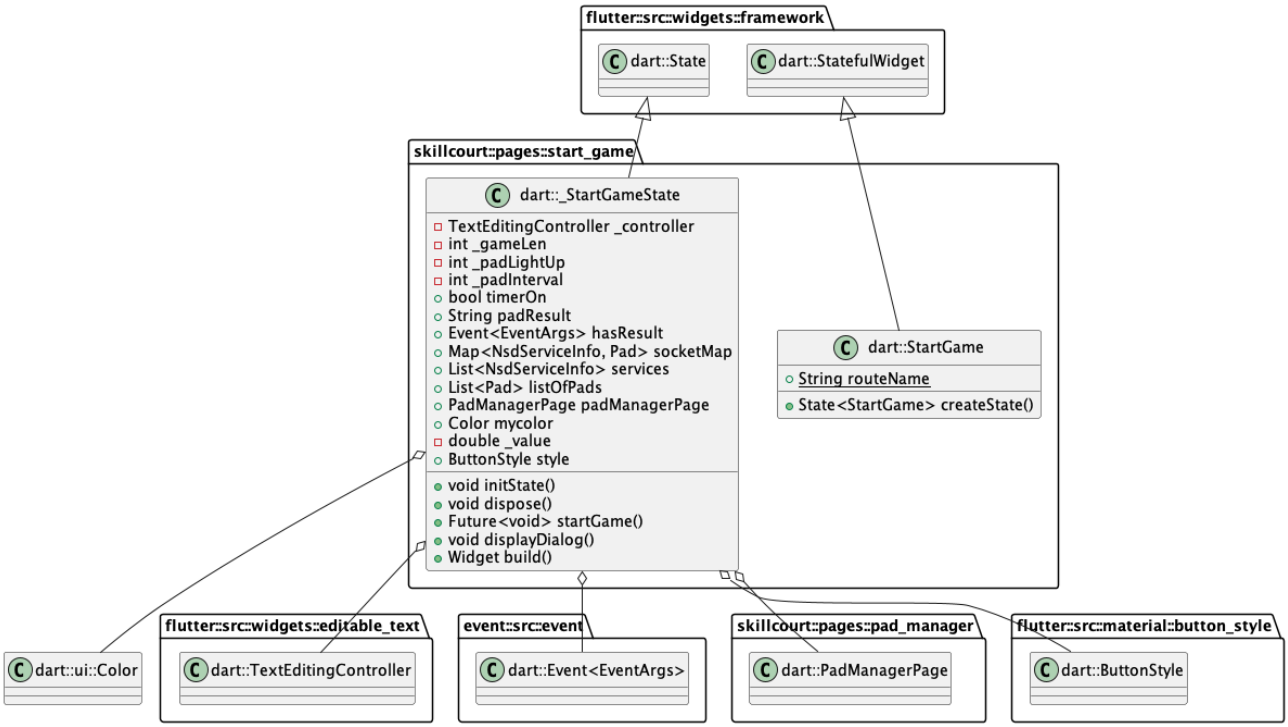
Project UML Diagram

(To view a better photo, please check the UML section)

The diagram above is an exhaustive UML diagram for the project in its entirety. Below we have cropped a portion of it that was of interest to us, as the brightness slider, color picker and custom color picker were added to start_game class this semester. Here you can see a part of the connections to start_game.dart. In our UML Folder of the final deliverable there are screenshots of the UML diagram by parts and the entire UML diagram in .svg, .vdx and .xmi format.



Project UML Diagram Start Game Section

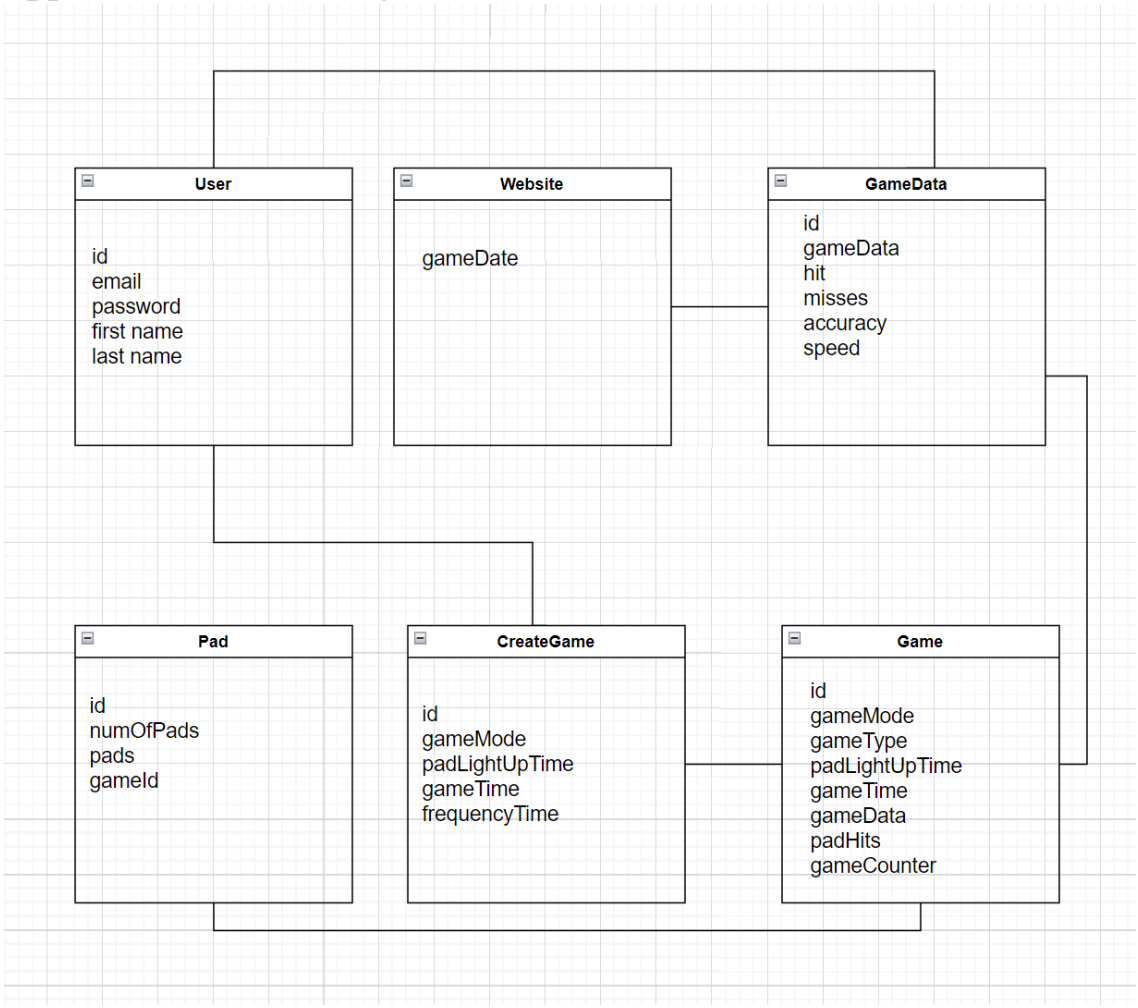


UML Diagram Start Game Class Exclusively

Above is the UML of the class most worked on this semester. All the new features that were added were added to this class. Functionally given to the user, as to controlling the pad and setting up the game was done in this class.

APPENDIX

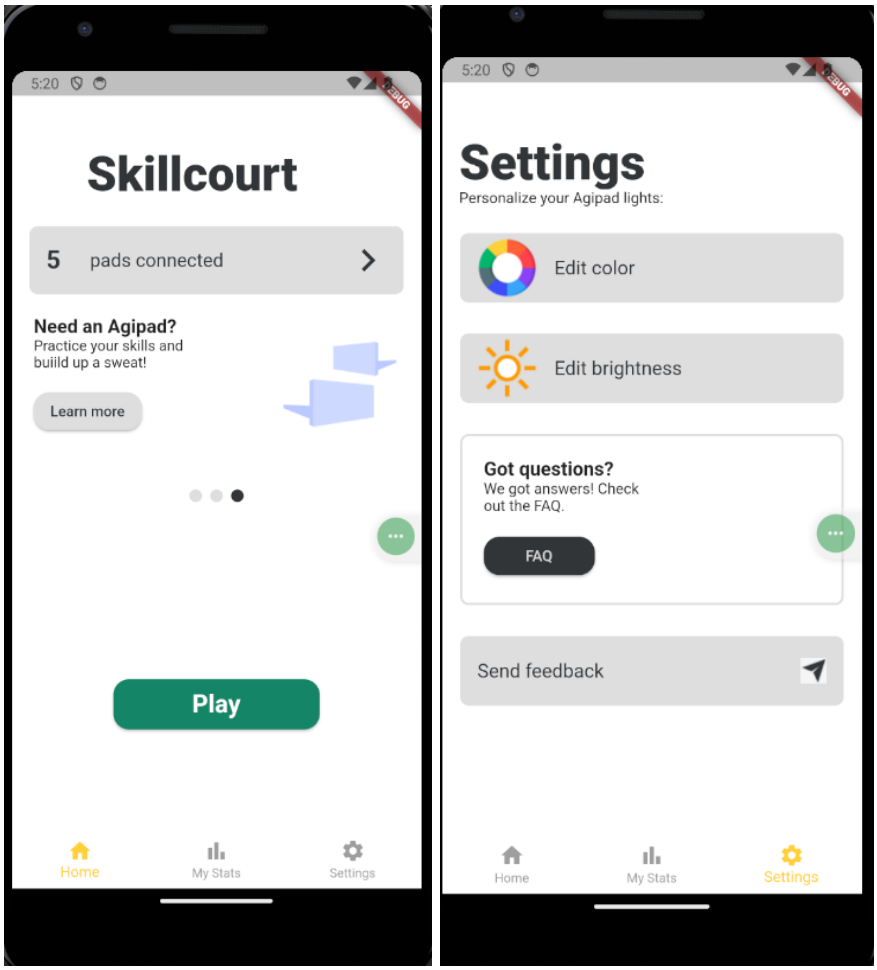
Appendix A - UML Diagram

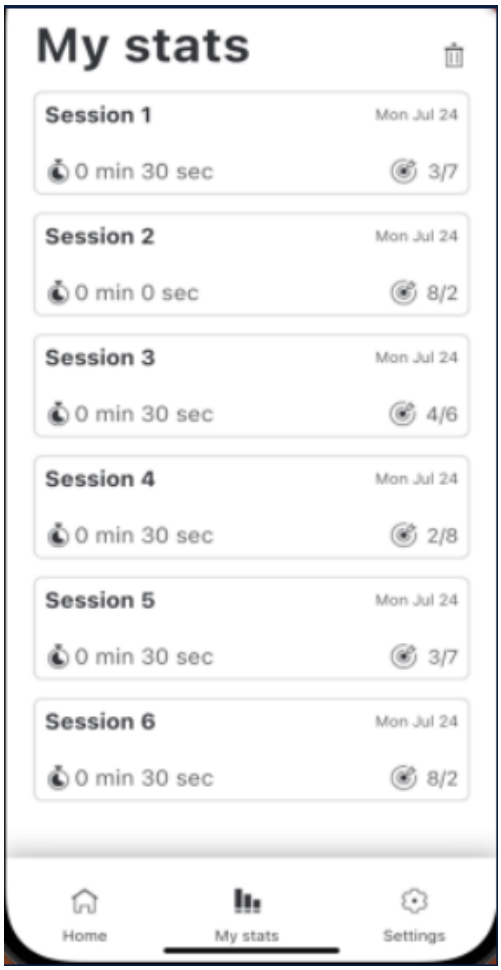


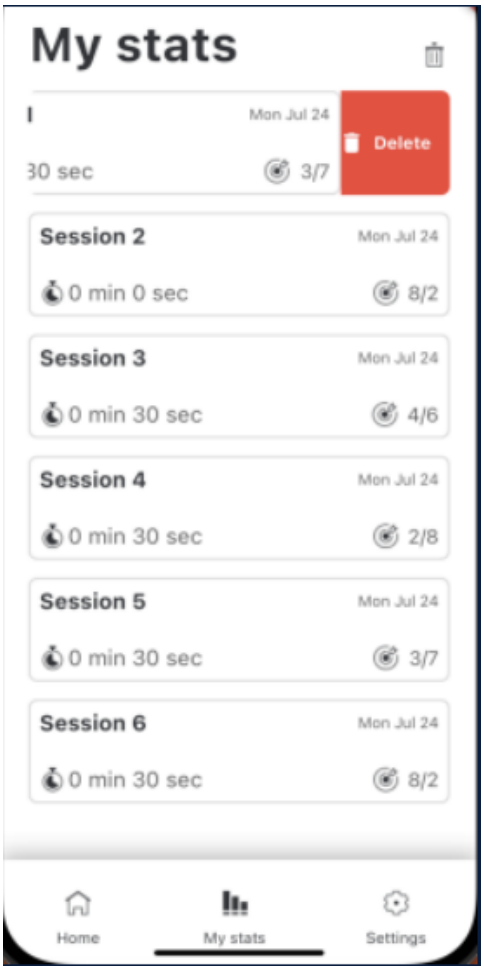
(Source: previous capstone 2 final documentation)

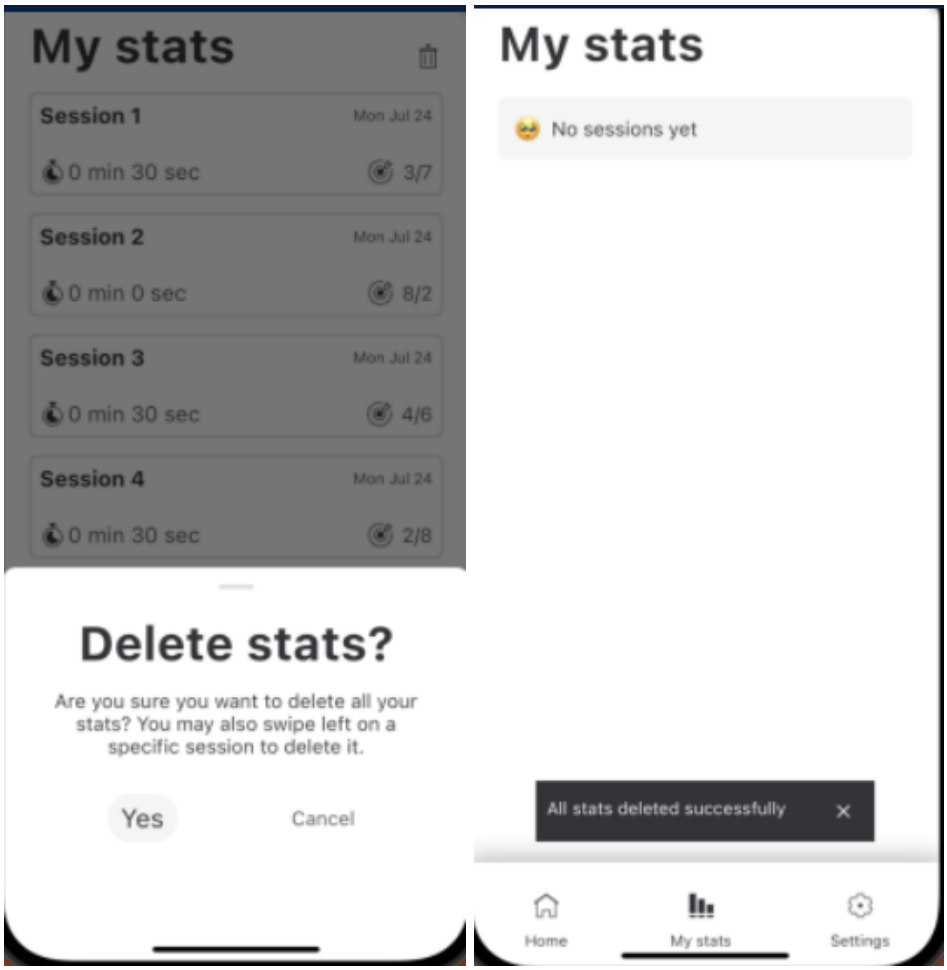
Appendix B - User Interface Design

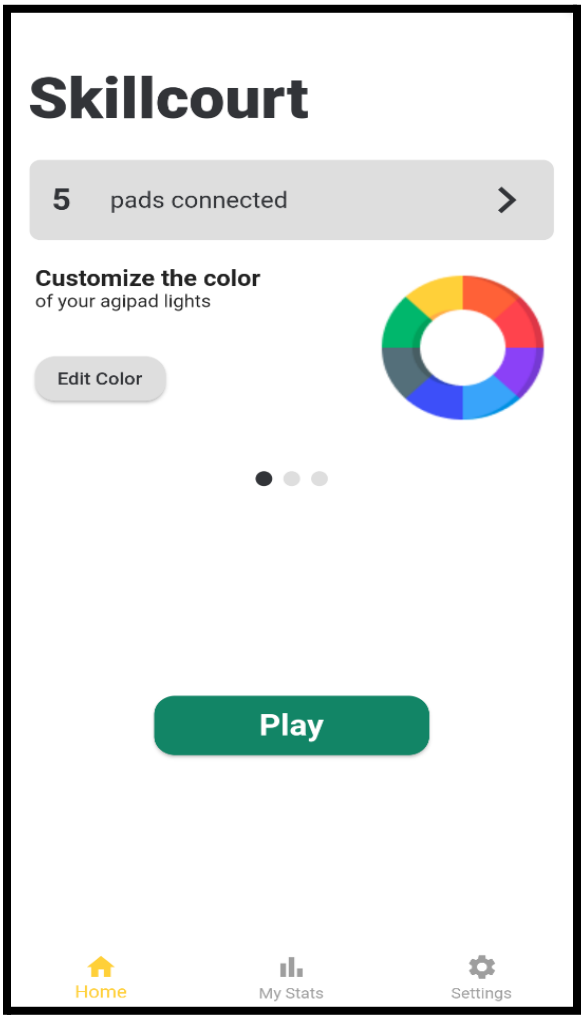
The images below show the current version of the SkillCourt Mobile application, including the Home page, Stats page, Brightness page, Color Picker page, Start Game page (new and saved), and Play Timer page



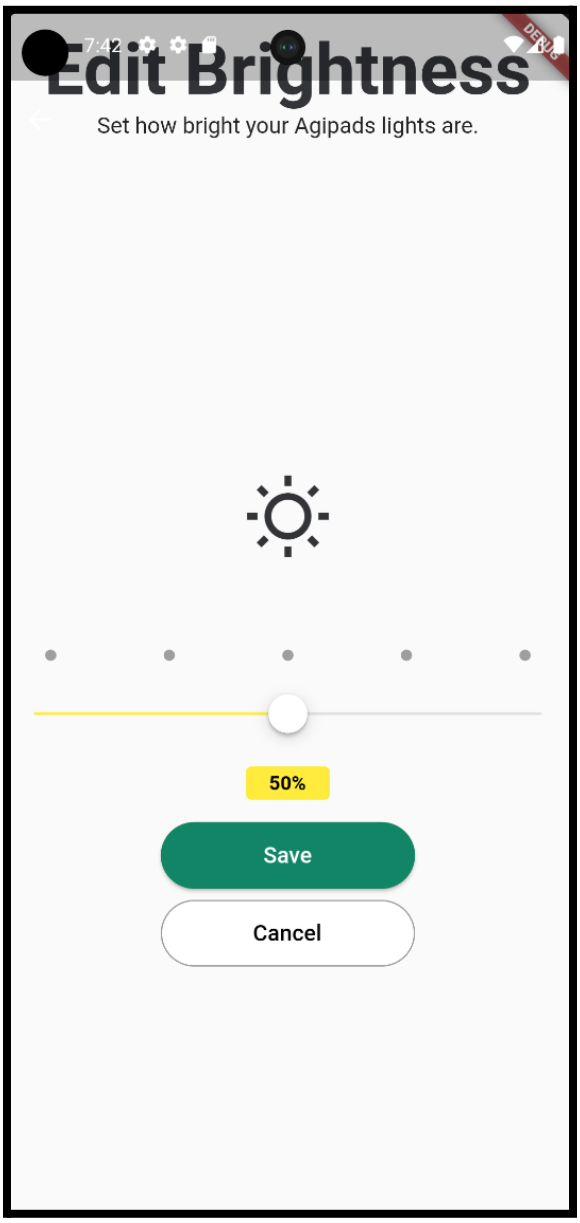

My stats
Session 1
Mon Jul 24
0 min 30 sec
3/7
Session 2
Mon Jul 24
0 min 0 sec
8/2
Session 3
Mon Jul 24
0 min 30 sec
4/6
Session 4
Mon Jul 24
0 min 30 sec
2/8
Session 5
Mon Jul 24
0 min 30 sec
3/7
Session 6
Mon Jul 24
0 min 30 sec
8/2
Home
My stats
Settings

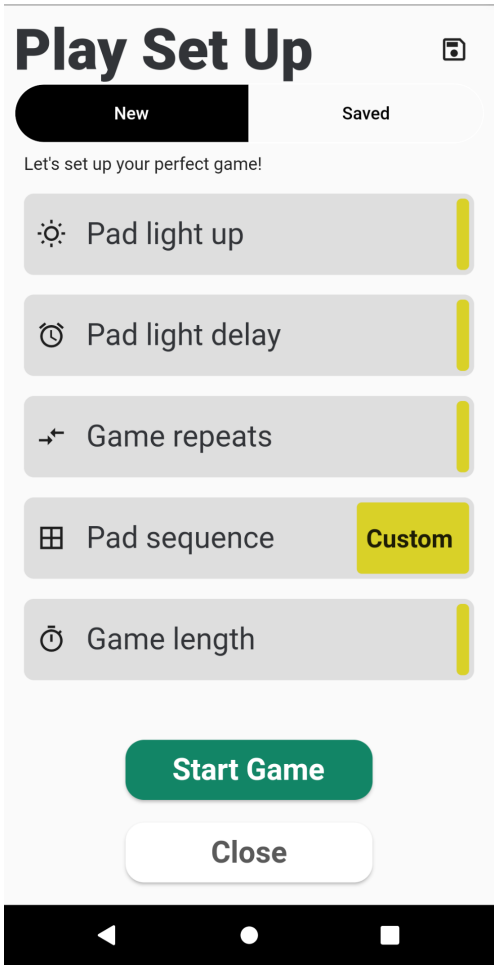

My stats
Mon Jul 24
Delete
30 sec
3/7
Session 2
Mon Jul 24
0 min 0 sec
8/2
Session 3
Mon Jul 24
0 min 30 sec
4/6
Session 4
Mon Jul 24
0 min 30 sec
2/8
Session 5
Mon Jul 24
0 min 30 sec
3/7
Session 6
Mon Jul 24
0 min 30 sec
8/2
Home
My stats
Settings

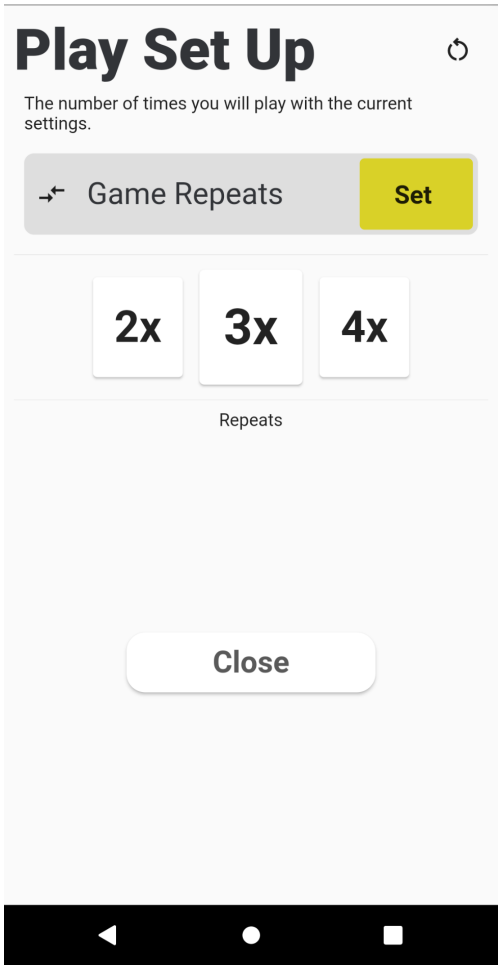

My stats
Session 1
Mon Jul 24
0 min 30 sec
3/7
Session 2
Mon Jul 24
0 min 0 sec
8/2
Session 3
Mon Jul 24
0 min 30 sec
4/6
Session 4
Mon Jul 24
0 min 30 sec
2/8
Delete stats?
Are you sure you want to delete all your stats? You may also swipe left on a specific session to delete it.
Yes
Cancel
My stats
No sessions yet
All stats deleted successfully
Home
My stats
Settings

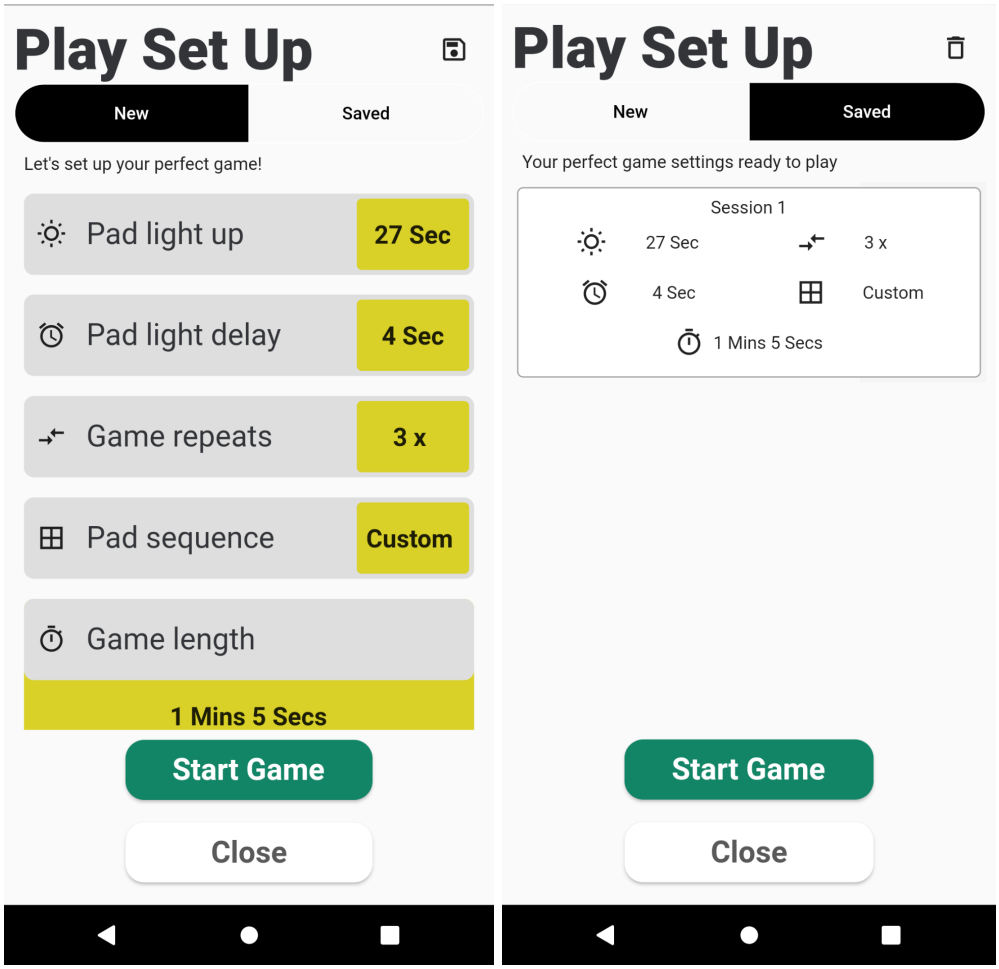

Skillcourt
5 pads connected
Customize the color
of your agipad lights
Edit Color
Play
Home
My Stats
Settings

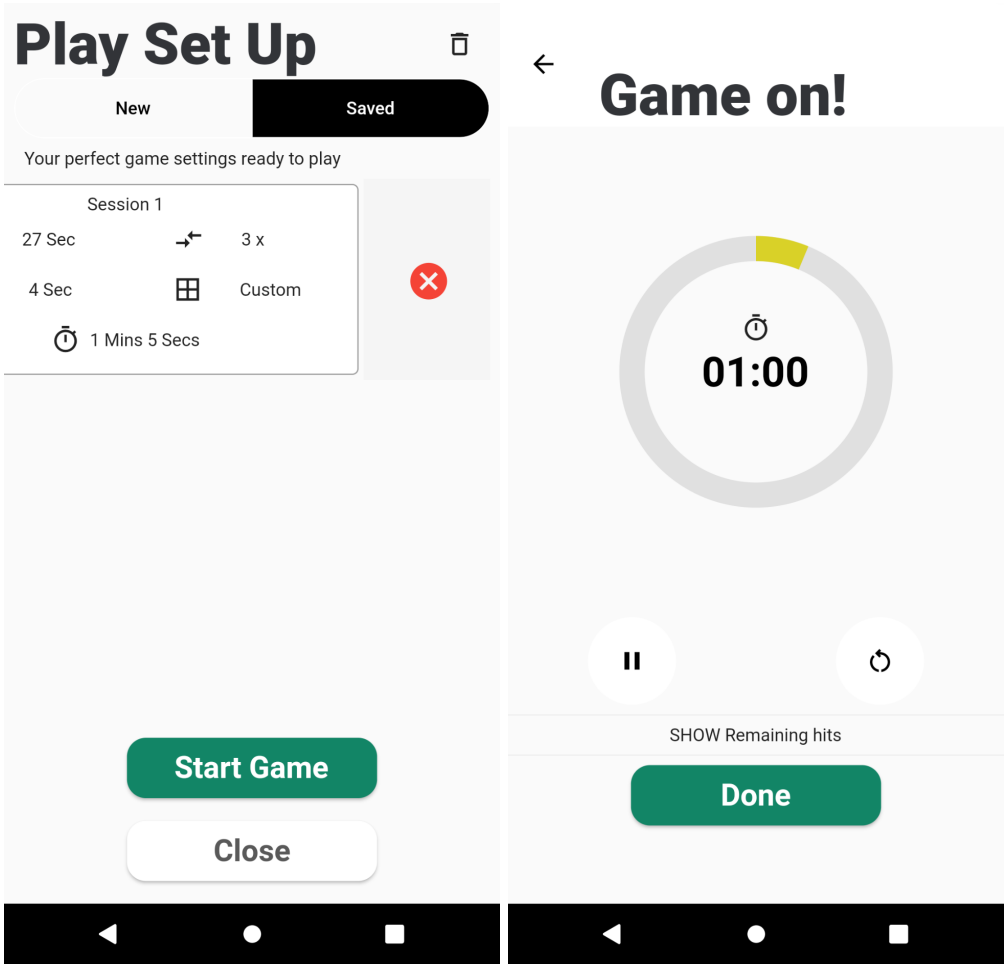
DEBUG
Edit Color
Tap on a color below to update the look of the LED lights.
Yellow
Green
Red
Blue
White
Orange
Violet
Cyan
Pink
Save
Cancel


Edit Brightness
Set how bright your Agipads lights are.
50%
Save
Cancel

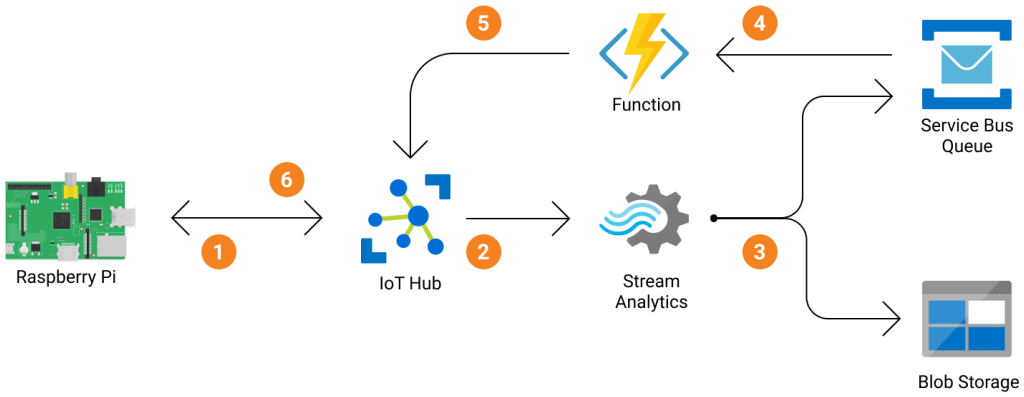

Play Set Up
New
Saved
Let's set up your perfect game!
Pad light up
Pad light delay
Game repeats
Pad sequence
Custom
Game length
Start Game
Close


Play Set Up
The number of times you will play with the current settings.
Game Repeats
Set
2x
3x
4x
Repeats
Close

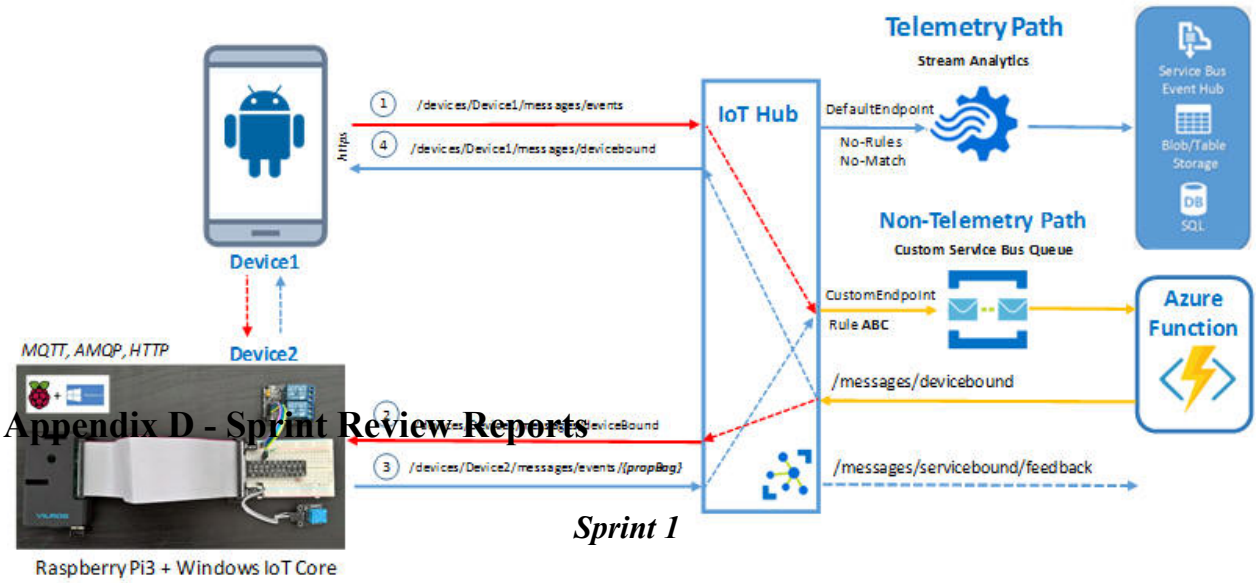

Play Set Up
New
Saved
Let's set up your perfect game!
Pad light up
27 Sec
Pad light delay
4 Sec
Game repeats
3 x
Pad sequence
Custom
Game length
1 Mins 5 Secs
Start Game
Close
Play Set Up
New
Saved
Your perfect game settings ready to play
Session 1
27 Sec
3 x
4 Sec
Custom
1 Mins 5 Secs
Start Game
Close


Play Set Up
New
Saved
Your perfect game settings ready to play
Session 1
27 Sec
3 x
4 Sec
Custom
1 Mins 5 Secs
Start Game
Close
Game on!
01:00
SHOW Remaining hits
Done

Appendix C - Azure IoT Hub Environment Architecture



Appendix D - Sprint Review Reports



Sprint 1

Sprint #1 Review Meeting Minutes

Attendees: Johan Balcerio, Johnny Casares, Luis Chavez
Start time: 4:15
End time: 4:20

- After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:
  - User Story #1 (5 points): Research and document findings on the communication protocols discussed, such as web sockets and Mdns. Research alternatives that might be applicable.
  - User Story #2 (2.5 points): Download, setup, and test application to understand its mechanics.
  - User Story #3 (2.5 points): Check figma and compare it to the current app. Identify aspects to be implemented.

- The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.
  - No user stories were rejected this sprint.

Sprint 2

Sprint #2 Review Meeting Minutes

Attendees: Johan Balcerio, Johnny Casares, Luis Chavez
Start time: 8:00 pm
End time: 8:20 pm

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story #1
- User Story #2

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

- User Story #3 was pushed to a future sprint due to a miscalculation in team velocity and user story points

Sprint 3

Sprint #3 Review Meeting Minutes

Attendees: Johan Balcerio, Johnny Casares, Luis Chavez
Start time: 9:00 pm
End time: 9:15 pm

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story #1
- User Story #2
- User Story #3
- User Story #4

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

- No user stories were rejected

Sprint 4

Sprint #4 Review Meeting Minutes

Attendees: Johan Balcerio, Johnny Casares, Luis Chavez
Start time: 9:00 pm
End time: 9:15 pm

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story #1
- User Story #2
- User Story #3
- User Story #4

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

- No user stories were rejected

Sprint 5

Sprint #5 Review Meeting Minutes

Attendees: Johan Balcerio, Johnny Casares, Luis Chavez
Start time: 8:00 pm
End time: 8:15 pm

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story #1: Resolving mDNS to Websocket migration issues (5)
- User Story #2: UI Implementation Epic: Part 3 (5)
- User Story #3: Researching Pad Emulators (1)
- User Story #4: Research Flutter app development further (1)
- User Story #5: Research Rust Implementation of Websocket (1)

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

- No user stories were rejected

Sprint 6

Sprint #6 Review Meeting Minutes

Attendees: Johan Balcerio, Johnny Casares, Luis Chavez
Start time: 9:30 pm
End time: 10:00 pm

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story #1: Setup IoT Hub Azure (5)
- User Story #2: UI Implementation Epic: Part 5 (5)
- User Story #3: Test Pad Features within IoT emulator (1)
- User Story #4: Merge all changes into a working branch (2)
- User Story #5: Research Rust Firmware for Raspberry Pi (1)

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

- No user stories were rejected

Sprint 7

Sprint #7 Review Meeting Minutes

Attendees: Johan Balcerio, Johnny Casares, Luis Chavez
Start time: 8:00 pm
End time: 8:15 pm

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- User Story #1: Resolving mDNS to Websocket migration issues (5)
- User Story #2: UI Implementation Epic: Part 3 (5)
- User Story #3: Researching Pad Emulators (1)
- User Story #4: Research Flutter app development further (1)
- User Story #5: Research Rust Implementation of Websocket (1)

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

- No user stories were rejected

Appendix D - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents

Mobile Installation:

- To access development of the application, you need to clone the SkillCourt -app repository.
  - You must contact the team leader or team member to gain access to the SkillCourt -app repositories because they are private.
- Download git from https://git-scm.com/downloads for source control management. Install the software for your specific operating system.
  - You can also install the Github Desktop app to access and manage the repositories.
  - We use the gitflow branching model to handle our features. The use of gitflow-avh is highly recommended as a plugin for bash or fishshell to handle the way in which gitflow operates.
  - Follow the branching model provided in each repository. They serve as a guide to the model for gitflow branching.
- For mobile application development, you will need to install Flutter and Dart https://docs.flutter.dev/get-started/install
- Install Android Studio or XCode or IDE of choice for developing and emulating the application
  - Note: It is advised to not work on the app on a virtual machine as the nested virtualization (emulator within an emulator) will likely cause unnecessary problems with development
- Run “flutter doctor” in the IDE to ensure that all dependencies are up to date
- Run “flutter pub get” inside the repository to pull the latest dependencies.
- Run “flutter pub run build_runner build --delete-conflicting-outputs” inside the repository.
- Run the main.dart application using the emulator of your choice.
- (Optional) Install requirements.txt using pip to download all required Python dependencies

Do’s and Don'ts & Helpful Tidbits To Know When Installing the Mobile App:

- Do Not install the mobile app inside of a virtual machine, and place Android Studios inside of that virtual machine. This will need nested virtualization which in turn will cause Android Studio to perform poorly because it requires quite a bit of memory to function.
- When you are in the process of creating the symlink, make sure that it is in the correct order. Linux and Windows have different order placements. If you are prompted to delete some of the test files in the Git area of the program, don’t. You might have your symlink order backwards. The hooks folder in the .git folder has sample code, hence the .sample extension to the files there;

you are to delete the code from the file pre-commit, remove the .sample extension and paste the actual pre-commit code in that file.

- If your sdk's are not working in Android Studio, make sure that the environment variables are properly mapped by checking the local environment variables on your personal workspace or personal computer. When installing the Flutter SDK, it does not always map its paths correctly on install.
- If you are still having issues with path's inside of the SkillCourt project, dart uses a tool called dart migrate. You can use this to localize some of the paths to your environment. Dart also has a tool that enables the user to check for hidden errors that are not listed by the flutter compiler called, dart fix –dry -run. Check the dart documentation for additional troubleshooting tools if you are still having flutter issues.

Website Installation:

- To access development of the application, you need to clone the SkillCourt-Website repository.
  - You must contact the team leader or team member to gain access to the SkillCourt Website repositories because they are private.
- Download git from https://git-scm.com/downloads for source control management. Install the software for your specific operating system.
  - You can also install the Github Desktop app to access and manage the repositories.
  - We use the gitflow branching model to handle our features. The use of gitflow-avh is highly recommended as a plugin for bash or fishshell to handle the way in which gitflow operates.
  - Follow the branching model provided in each repository. They serve as a guide to the model for gitflow branching.
- For website development, you will need to install:
  - Node >= v14.0.0
  - Pnpm >= v6.21.0
  - Docker
- Clone the repository
- Copy the file .env.example to .env and fill in all variables as appropriate.
- Create a file docker-compose.override.yml and include the information stated on the readme.md
- Run the following commands:
  - $ docker compose up [-d] [--build]
  - $ docker compose exec webapp pnpx prisma db push
  - $ docker compose exec webapp pnpx prisma db seed

Do Nots and Helpful Tidbits To Know When Installing the Website:

- If you are getting an Eaccess denied error from the compiler when using docker compose up, Prisma seed, and etc.. , it might not be an actual passcode error that is denying you entry. There might also be packages that are outdated that need to be updated.
- In the linux, vscode, or any terminal that you are using when you are typing docker compose commands, there are two docker compose methods that you could be using:
  - docker compose, which is attached to Docker Desktop
    - Make sure that you are only using one version, and I recommend using the newest version of docker compose.
  - docker-compose, an older version that is completely separate from Docker Desktop
    - If you have to use this version of docker compose, there is an option to only run version 1 or 2 of this inside of the docker desktop settings that you must click. If you don't, you will be in for a world of pain.
- There are two ways to run the website database from Windows:
  - With wsl2 with linux installed
  - With a virtual machine like Oracle VM VirualBox with linux installed
    - If you are running the project inside of the virtual box, you will have to enable nested virtualization from the Windows bios as well as the virtual machine settings. Docker Desktop runs fine inside of a virtual machine with nested virtualization because it is not as demanding as Android Studio.
- If you are using a virtual machine, make sure that you enable plenty of disk space for it. I found 40-50 gigs to be more than enough.

Mobile Development Manual:

- On the IDE of the user's choice (for these instructions, Android Studios will be referenced), install a virtual device using the virtual device manager.
  - Have the virtual device use the most up to date Android OS to ensure the application works on modern systems
- Open the emulator and familiarize with the phone emulator, how to open apps and interact with them for example. Ensure the emulator is running with no issues prior to testing the application.
- Click the play button on the top right to launch the application on the phone emulator
  - This will compile and emulate the application on the phone emulator.
  - Android Studio will either return an error code detailing why the application does not launch or the emulator will begin running the application
- The application should be tested with each and every change to keep track of how specific changes affect the working state on the application.

Team shortcomings

- **Time:** Even though the team has a solid understanding of object oriented programming, and adapting to the flutter framework wasn't difficult, time still was needed to study the framework and make sure best practices were used. However, learning those practices and meeting the deadlines proved to be an adversity.
- **Lack of a pad emulator:** Without a proper way to emulate the pads, we were stuck to developing a UI that lacks methods that communicate data between the phone and the pads.

Mobile App and Firmware Shortcomings

- Navigating the source code and fixing previous misuse of flutter widgets and poor design.
- Inconsistent
- Lack of a proper pad emulator makes it near impossible to develop an application whose main purpose is the interaction between the mobile app and the hardware in the pads

Wishlist:

Mobile Application:

- More development time to be able to test and implement new features rather than updating current ones to work on new hardware.

Firmware:

- More time to work with and update the pad's firmware logic instead of the hyper focus on websocket implementation would have been helpful in order to get more acquainted with the pad's logic and capabilities.

REFERENCES

- Previous Team's Final Documentation:
  - https://drive.google.com/drive/folders/1--5ySXkyESTxCPfavAZrhdscH_UW8Pr-
- Git Download:
  - https://git-scm.com/downloads
- Flutter Download and Documentation:
  - https://docs.flutter.dev/get-started/install
- Mac XCODE Documentation:
  - https://developer.apple.com/xcode/
- Android Studio Documentation:
  - https://developer.android.com/studio/?gclsrc=ds&gclsrc=ds
- Oracle JDK JRE 8 Download:
  - https://docs.oracle.com/javase/8/docs/technotes/guides/install/install_overview.html
- Docker Download:
  - https://www.docker.com/
- Docker Desktop Documentation:
  - https://docs.docker.com/
- Github Desktop Download:
  - https://desktop.github.com/
- Gitflow AVH:
  - https://github.com/petervanderdoes/gitflow-avh
- Angular Typescript Install and Documentation:
  - https://angular.io/
- Nest.js Download and Documentation:
  - https://docs.nestjs.com/
- Prisma.io Install and Documentation:
  - https://www.prisma.io/
- Dart Documentation:
  - https://dart.dev/guides
- PNPM Documentation:
  - https://pnpm.io/
- Docker Desktop Proper Container Install Instructions and Troubleshooting Links
  - https://hevodata.com/learn/docker-postgresql/
  - https://www.docker.com/blog/docker-0-6-5-links-container-naming-advanced-port-redirects-host-integration/
  - https://www.optimadata.nl/blogs/1/n8dyr5-how-to-run-postgres-on-docker-part-1
  - https://www.sqlshack.com/getting-started-with-postgresql-on-docker/
- Docker Unknown Flag Solution:
  - https://www.sitecoregabe.com/2021/07/docker-compose-failing-unknown-flag-d.html#:~:text=unknown%20flag%3A%20--d%20As%20it%20turns%20out%2C%20this,the%20settings%3B%20or%20run%20%60%20docker-compose%20disable-v2%20%60

- Finding The Correct Flutter SDK Path:
  - https://www.youtube.com/watch?v=zcZd59zBoaY
- Upgrading The Flutter App
  - https://stackoverflow.com/questions/64797607/how-do-i-upgrade-an-existing-flutter-app
- Flutter Widget Documentation:
  - https://flutterawesome.com/flutter-platform-widgets/
- Flutter Issues With Gradle, Mapping New NS to Old NS
  - https://navoki.com/flutter-android-error-solved-warning-mapping-new-ns-to-old-ns/
- Virtual Machine Ubuntu Install Instruction Video:
  - https://www.youtube.com/watch?v=_u8qTN3cCnQ
- Mozilla Developer Network. (n.d.). Writing WebSocket servers
  - https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API/Writing_WebSocket_servers
- Microsoft. (n.d.). Understand Different Connection Strings in Azure IoT Hub.
  - https://devblogs.microsoft.com/iotdev/understand-different-connection-strings-in-azure-iot-hub/
- Microsoft. (n.d.). IoT Hub live data visualization in web apps.
  - https://learn.microsoft.com/en-us/azure/iot-hub/iot-hub-live-data-visualization-in-web-apps
- Flutter. Communicate with WebSockets
  - https://docs.flutter.dev/cookbook/networking/web-sockets
- Connect Raspberry Pi to Azure IoT Hub
  - https://learn.microsoft.com/en-us/azure/iot-hub/iot-hub-raspberry-pi-kit-node-get-started