SIVAN MOODLEY - ST10084225

OPSC7311 - PART 2

README FILE

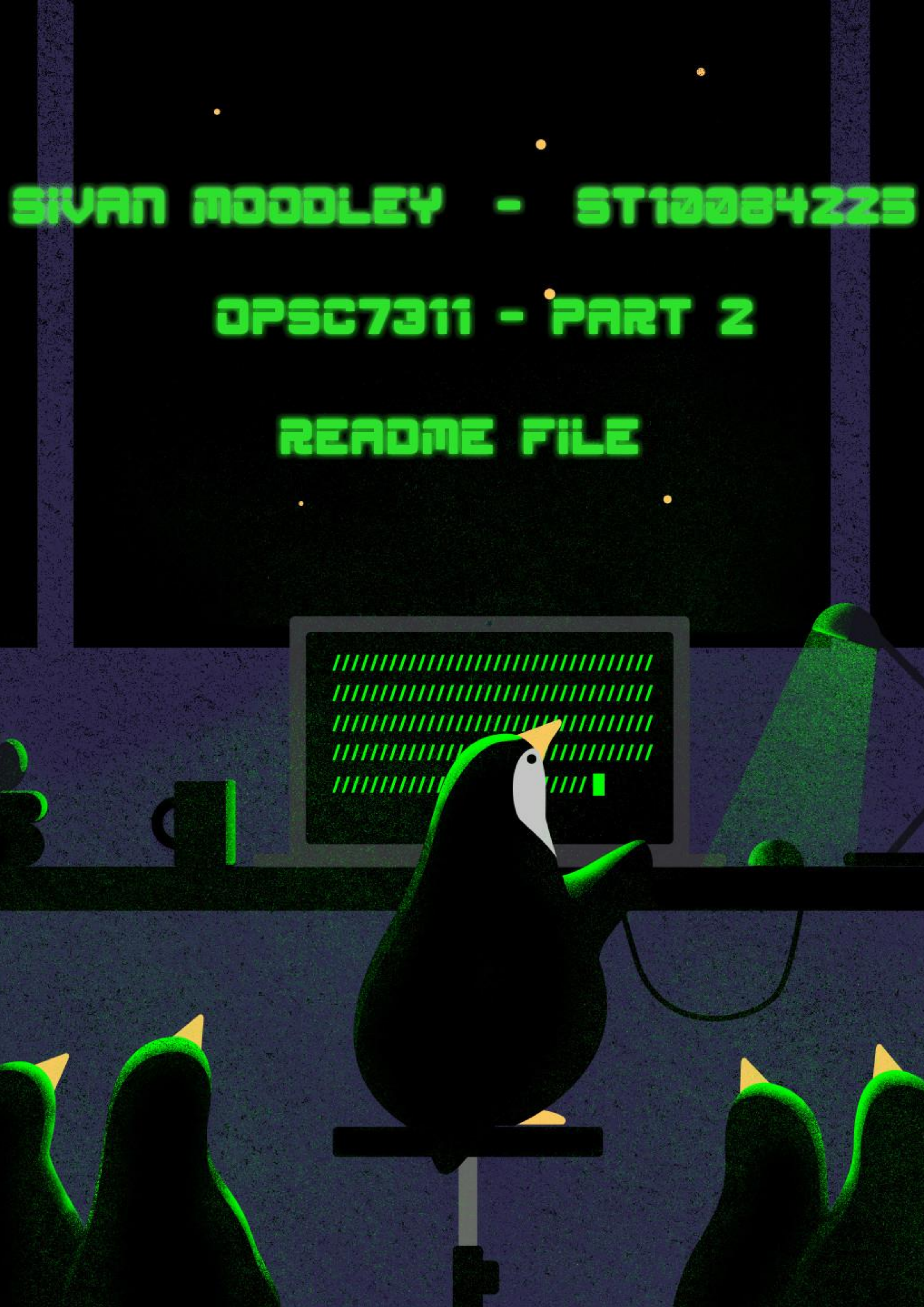


Table of Contents

Changes in Part 3

Note: These changes are further detailed in the Bonus_Feature_Review.m4v in my GitHub.

1.) Migrated the app from Java to Kotlin
2.) UI Rework
3.) UI Optimization
4.) Focus Group Testing
5.) Feedback Implemented
6.) Added Stat Charting Features (Min/Max)
7.) Charting Feature Optimization (Min/Max)
8.) Added Game Stat Charting Features (K/D, Avg KDA)
9.) Added Charting Date Filters (Min/Max, User Specified)
10.) Added Charting Date Filters (Both, 1 Month/Week)
11.) Charting Date Filters Optimization
12.) Final UI Rework
13.) Navigation Rework
14.) Dashboard Rework
15.) Dependency Optimization
16.) Glide Module Optimization
17.) Firebase Storage Optimization
18.) Firebase Firestore File Structure Rework for Kotlin
19.) Firebase In-App Read/Write Operation Optimization
20.) Final Focus Group Testing
21.) Implement Feedback
22.) Deploy Final App to GitHub Classroom

Description

I have been tasked with building a time tracking app for the Android OS. I have decided to design and develop a time tracker based on competitive video gaming, as I noticed there is a niche for them during my research in part 1 of the POE.

The app will allow users to upload categories(Video Games), and timesheets(play sessions) to these categories, to help the user keep track of their playtime and their in-game stats.

This version of my app, is a prototype for the final version which will be complete in Part 3 of the POE.

The prototype implements the following minimum requirements:

- The user must be able to log in to the app using a username and password.
- The user must be able to create categories that the timesheet entries will belong to.

• The user must be able to create a timesheet entry, specifying at least the date, start and end times, description, and category.

• The user must be able to optionally add a photograph to each timesheet entry.

• The user must be able to set a minimum daily goal for hours worked, as well as a maximum daily goal.

New Requirements in Part 3

- The user must be able to view a chart that displays the minimum and maximum goals of each of their categories and their current progress over the course of the previous month at meeting these goals.
- The user's data must be stored in an online database (Requirement already met in my part 2 submission)

Changes made to these requirements to fit the context of my app:

• The user must be able to create categories **(these will represent the different games the user would like to track)** that the timesheet entries will belong to.

• The user must be able to create a timesheet entry, specifying at least the date, start and end times, description, and category, **along with their sessions, kills, deaths, assists.**

• The app must use the user input for a timesheet entry to calculate in-game derived stats, such as **Kill/Death Ratio (K/D)** and total session **Playtime.**

• The user **Must** be able add a photograph to each **Category** and each timesheet entry.

• The user must be able to set a minimum and maximum daily goal for Hours Played of each Category (Game).

New Changes Made to The Requirements in Part 3

- The user must be able to view a chart that displays the minimum and maximum goals of each of their categories and their current progress over the course of the previous ***Month/Week***, ***Or Between a User Specified Date Range,*** at meeting these goals. (Explained further in Bonus Features).
- The user must be able to view a chart that displays the Avg KD and KDA of each of their categories and their current progress over the course of the previous ***Month/Week***, In terms of maintaining consistent progress in their games. (Explained further in Bonus Features).
- The Online Database, and its related In-App Read/Write Operations must be fast enough to complete a first time load, of any page, on any device within 4

seconds. ***(Note: This was already achieved in part 2, since it had used Java, since the migration to Kotlin in part 3 the database file structure had to be reworked from the ground up to meet the same standard in Kotlin).***

Developer Info

Name: Sivan Moodley
Student number: ST10084225
Cell phone number: 0815750712
Email: sivan.moodley02@gmail.com

Software Specs

1.) Android Studio Flamingo V2022.2.1
2.) Android OS version 13.0
3.) GitHub Version 3.5.0
4.) Microsoft Office Pro 2019
5.) Adobe Photoshop 2022
6.) Sublime Text Stable Channel, Build 4126

F.A.Q

1.) Do I require an internet connection to use this app?
~ Yes, the app utilizes Firebase Storage and Firebase Firestore to hold all app related data, so you will need to be connected to the internet on your device (or emulator) before launching the app.

2.) How do I add a new category?
~ You can add a new category from the about page by clicking the "Add Category" quick link image, or you can add a new category directly from the dashboard page ("View Categories" quick link image on about page) by clicking the floating action button in the bottom right corner of the screen.

3.) How do I add a new timesheet?
~ You can add a new timesheet from the about page by clicking the "Add a New Timesheet" quick link image, or you can add a new timesheet directly from the category details page ("View Categories" quick link image on about page → click the category you'd like to view) by clicking the floating action button in the bottom right corner of the screen.

4.) How do I view my uploaded categories?
~ You can view your uploaded categories from the about page by clicking the "View Categories" quick link image, or clicking the middle icon (Dashboard) on the bottom-nav-bar.

5.) How do I view the timesheets that belong to my categories?
~ You can view your uploaded timesheets from the about page by clicking the "View Categories" quick link image, or clicking the middle icon (Dashboard) on the bottom-

nav-bar, once on the dashboard page you will see a list of your account's uploaded categories, clicking on any one of these categories will display all your uploaded timesheets for that specific category.

Code Attribution

1.) Introduction to Firestore: https://firebase.google.com/docs/firestore/data-model#java (Accessed: 05/30/2023).

2.) Firestore Read Operations: https://firebase.google.com/docs/firestore/query-data/get-data (Accessed: 06/01/2023).

3.) Firestore Write Operations: https://firebase.google.com/docs/firestore/manage-data/add-data (Accessed: 06/02/2023).

4.) Introduction to Firebase Storage: https://firebase.google.com/docs/storage/android/start (Accessed: 06/02/2023).

5.) Upload Files to Firebase Storage: https://firebase.google.com/docs/storage/android/upload-files (Accessed: 06/03/2023).

6.) Download Files to Firebase Storage: https://firebase.google.com/docs/storage/android/download-files (Accessed: 06/03/2023).

Software Requirements

- Android Studio Electric Eel V2022.1.1 or higher
- Android OS version 4.4 or newer, API target version: 31
- An Android emulator (Android Studio, BlueStacks, etc)

Hardware Requirements

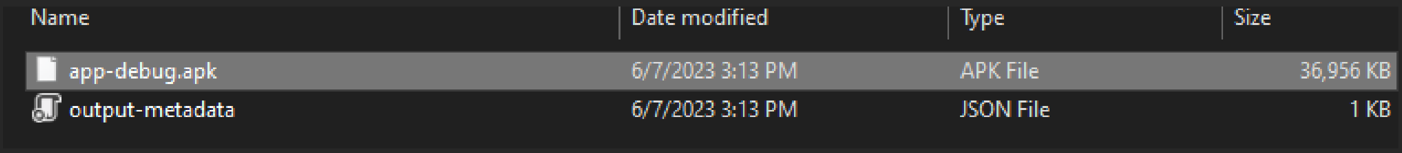
- Android OS version 4.4 or newer, API target version: 31
- 1 GB of RAM; 3 GB of RAM recommended.
- Device Disk Space: Minimum of 64MB.

How to Compile and Run the App in Android Studio Flamingo V2022.2.1

Unzip the project file if necessary.

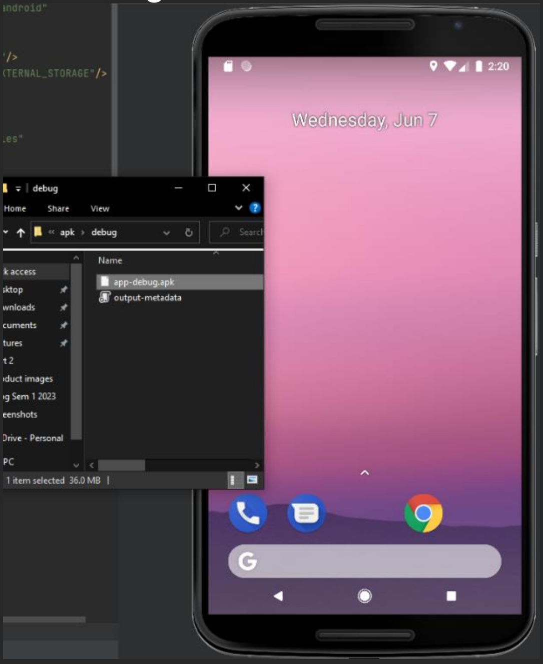
Open the SkillTrack_KotlinV20 folder.

Navigate to the app APK file => SkillTrack_KotlinV20\app\build\outputs\apk\debug



Name	Date modified	Type	Size
app-debug.apk	6/7/2023 3:13 PM	APK File	36,956 KB
output-metadata	6/7/2023 3:13 PM	JSON File	1 KB

Drag and drop the app-debug.apk file onto your android emulator while the emulator is running.



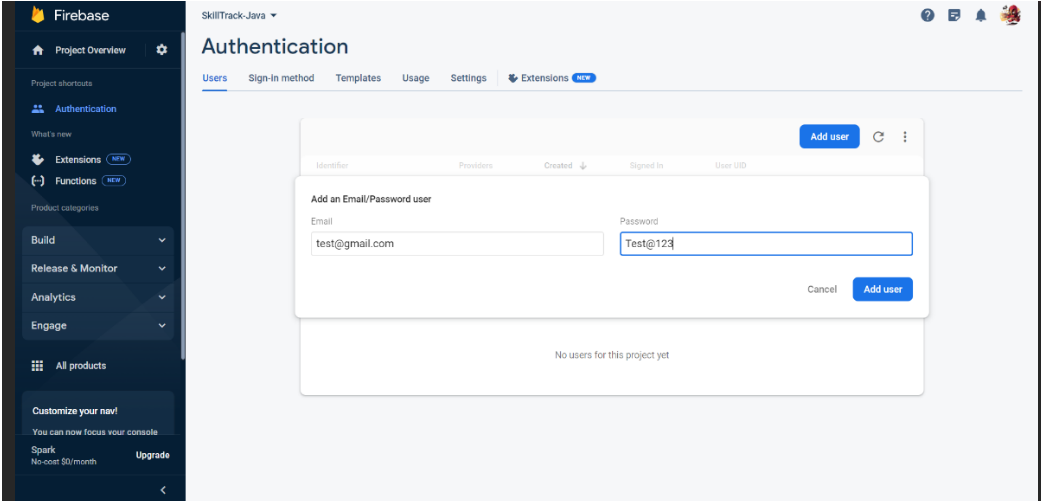
After the APK is done installing, navigate through your emulator device and find the app (SkillTrack), launch it by clicking the Icon.

If you wish to test the app on a physical android device, repeat the same steps, but instead of dragging and dropping your apk onto your emulator, add the apk to your physical devices storage, navigate to the location you saved it in on your physical device. Now run the APK and allow it to install the app on your physical device, once the app is done installing, navigate to it on your physical device and click its icon to launch the app.

Note: you may also run the app by manually importing the project file (SkillTrack_Kotlin) into Android studio and running it as you normally would. If you wish to test the app on a physical android device via this method, you will need import the project file (SkillTrack_Kotlin) into your android studio, configure your physical emulator in your device manager, and then run the app as you normally would in Android studio.

Testing the app

Note: all login and register details are case-sensitive

Testing Account: Email: test@gmail.com Password: Test@123	

Add an Email/Password user

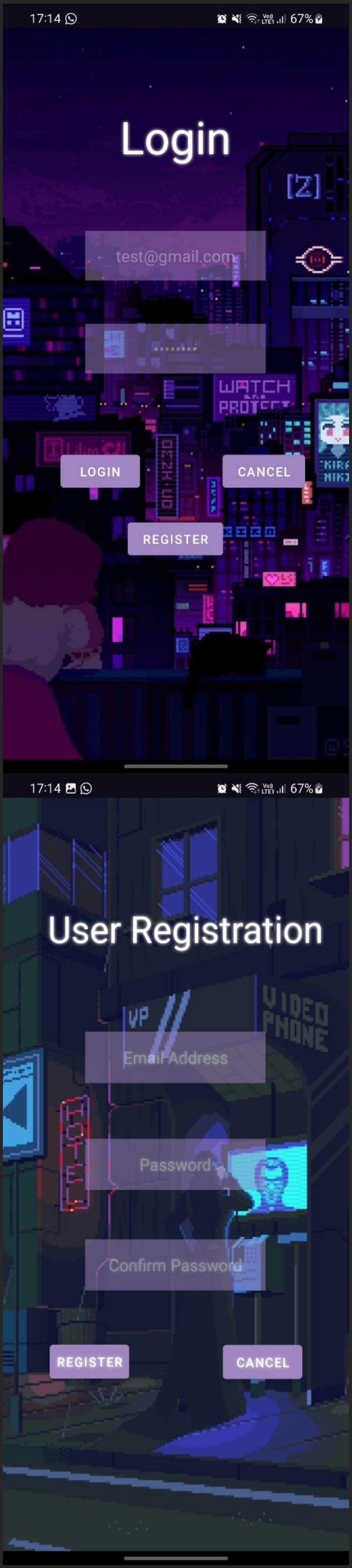
Email: test@gmail.com

Password: Test@123

No users for this project yet

Note: You may create your own custom accounts, however if you'd like to see my image(gif) animation parsing from user uploads, I'd recommend using this test account first, it contains some sample data.

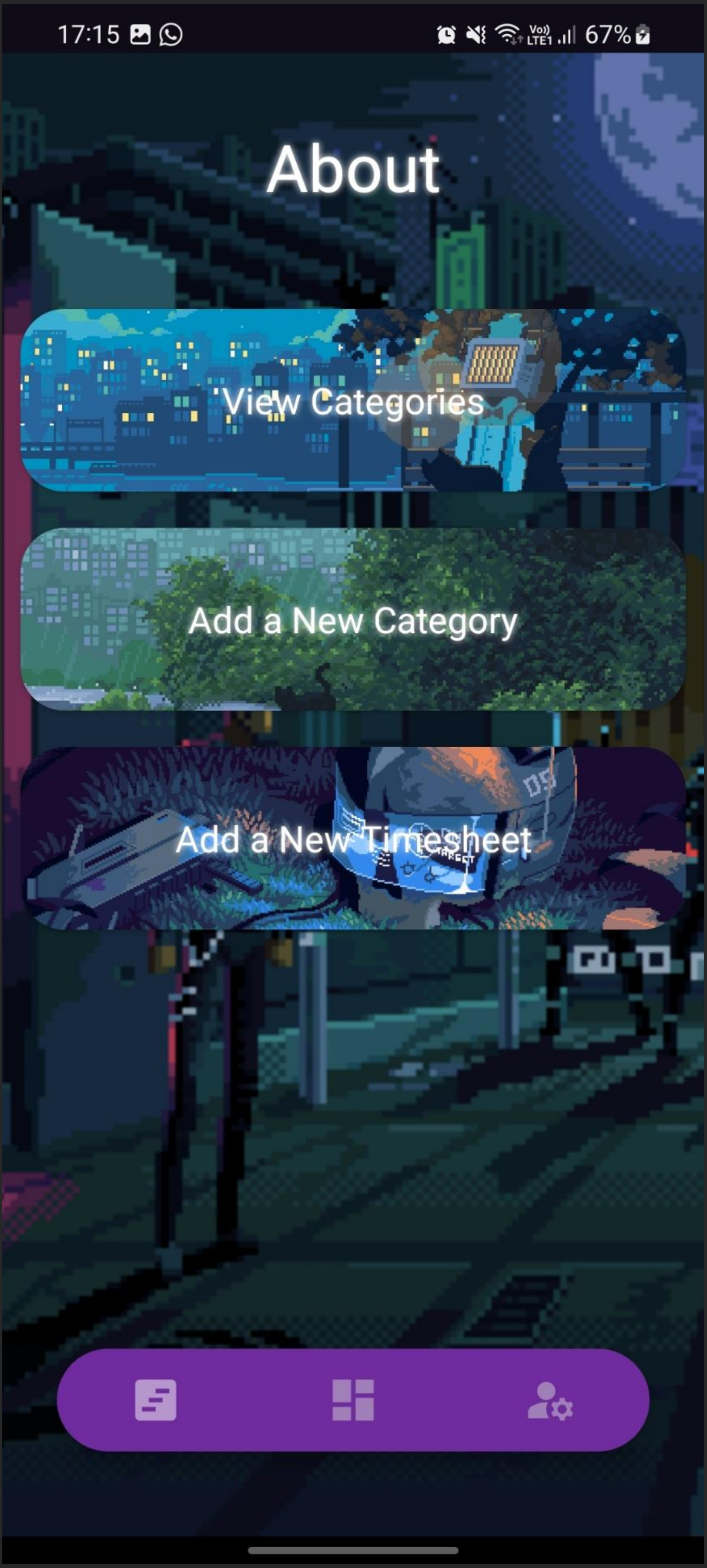
Testing Methodology: (On Next Page)



<Fig 1.1 Login/Register>

Launch the App.

Login using the testing account details, or create your own account by clicking register



<Fig 1.2 About (Home) >

3.) To view your uploaded categories, click the first image button, or the navbar's centre icon

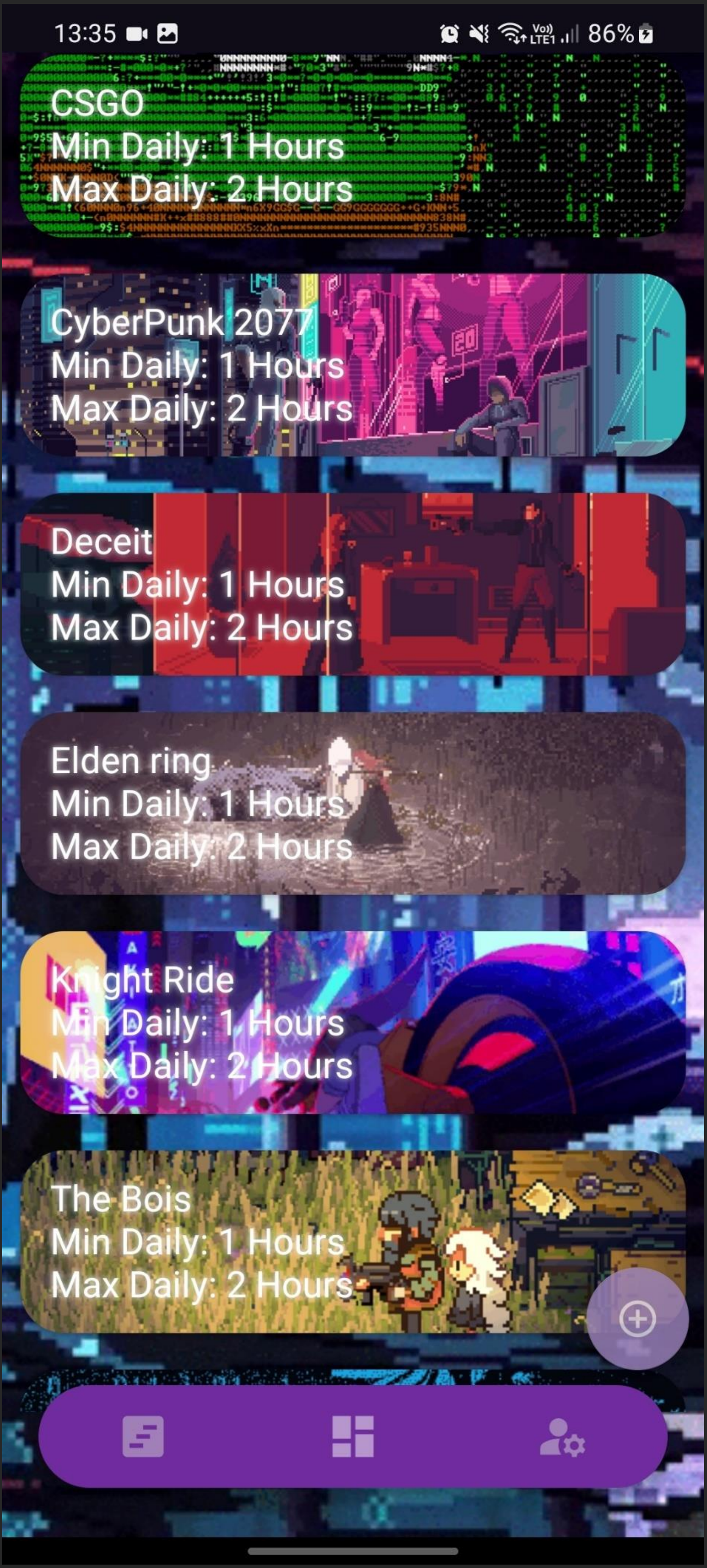
4.) To add a new category, click the second image button.

5.) To add a new Timesheet click the third image button.

~~~~~Navbar items~~~~~~~~
1.) About (Current page)
~ Activity quick links

2.) Dashboard
~View and Create Categories/Timesheets

3.) Profile
~ Logout (Goes back to login and register).
~~~~~~~~~~~~~~~~~~~~~~~~~



13:35 86%

CSGO
Min Daily: 1 Hours
Max Daily: 2 Hours

CyberPunk 2077
Min Daily: 1 Hours
Max Daily: 2 Hours

Deceit
Min Daily: 1 Hours
Max Daily: 2 Hours

Elden ring
Min Daily: 1 Hours
Max Daily: 2 Hours

Knight Ride
Min Daily: 1 Hours
Max Daily: 2 Hours

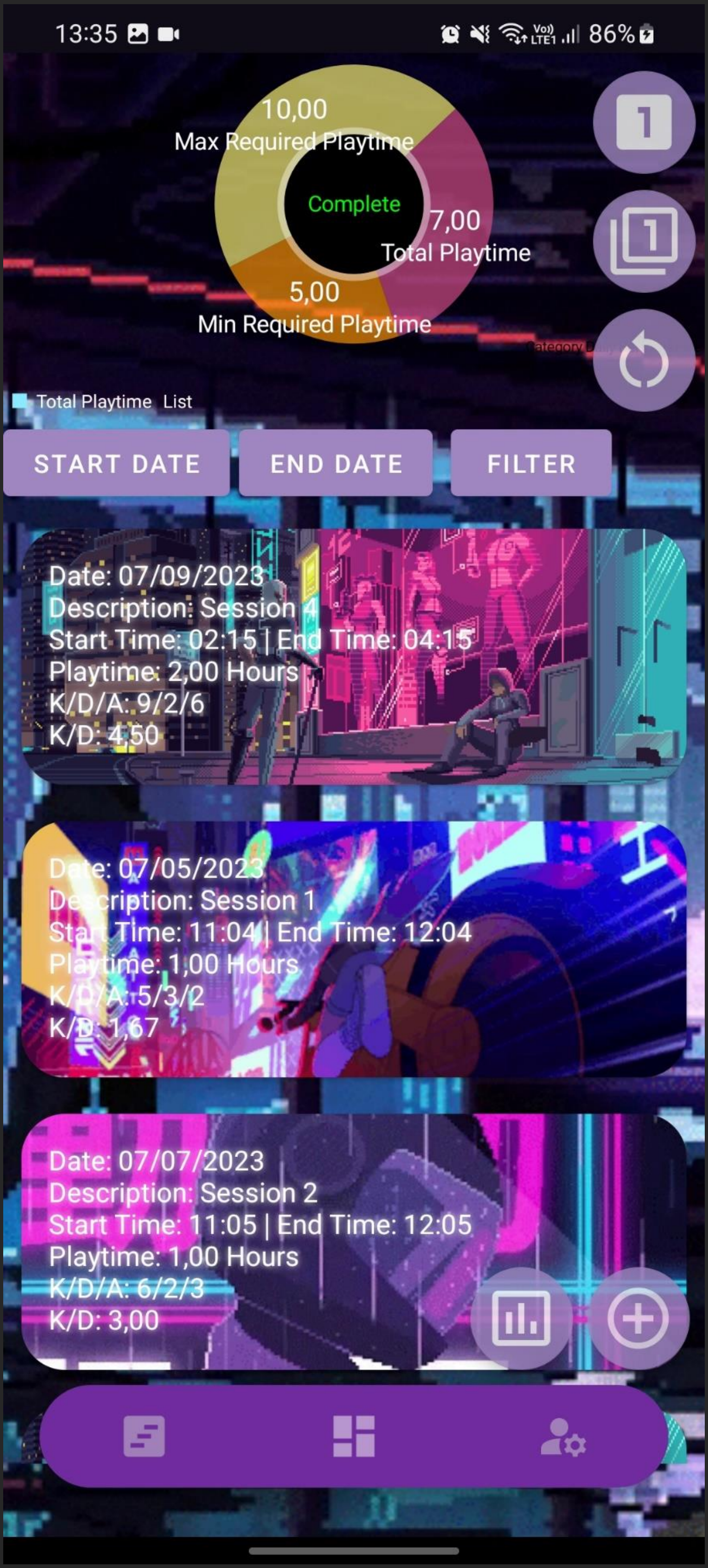
The Bois
Min Daily: 1 Hours
Max Daily: 2 Hours

<Fig 1.3 Dashboard (View Categories) >

6.) To view Timesheet entries for a specific category, click that category’s image button.

To add a new category, click the floating action button in the corner of the screen.

Note: View overflow is scrollable, if there are elements going off the screen try scrolling by swiping your finger across the screen.



Date: 07/09/2023
Description: Session 4
Start Time: 02:15 | End Time: 04:15
Playtime: 2,00 Hours
K/D/A: 9/2/6
K/D: 4,50

Date: 07/05/2023
Description: Session 1
Start Time: 11:04 | End Time: 12:04
Playtime: 1,00 Hours
K/D/A: 5/3/2
K/D: 1,67

Date: 07/07/2023
Description: Session 2
Start Time: 11:05 | End Time: 12:05
Playtime: 1,00 Hours
K/D/A: 6/2/3
K/D: 3,00

<Fig 1.4 Dashboard (View Timesheets) >

To add a new timesheet, click the floating action button in the bottom right corner of the screen.

To View Game Stats, click the second floating action button in the bottom right corner of the screen.

How this chart loads data along with how it handles different load states are further explained in the Bonus Feature segment of this ReadMe.

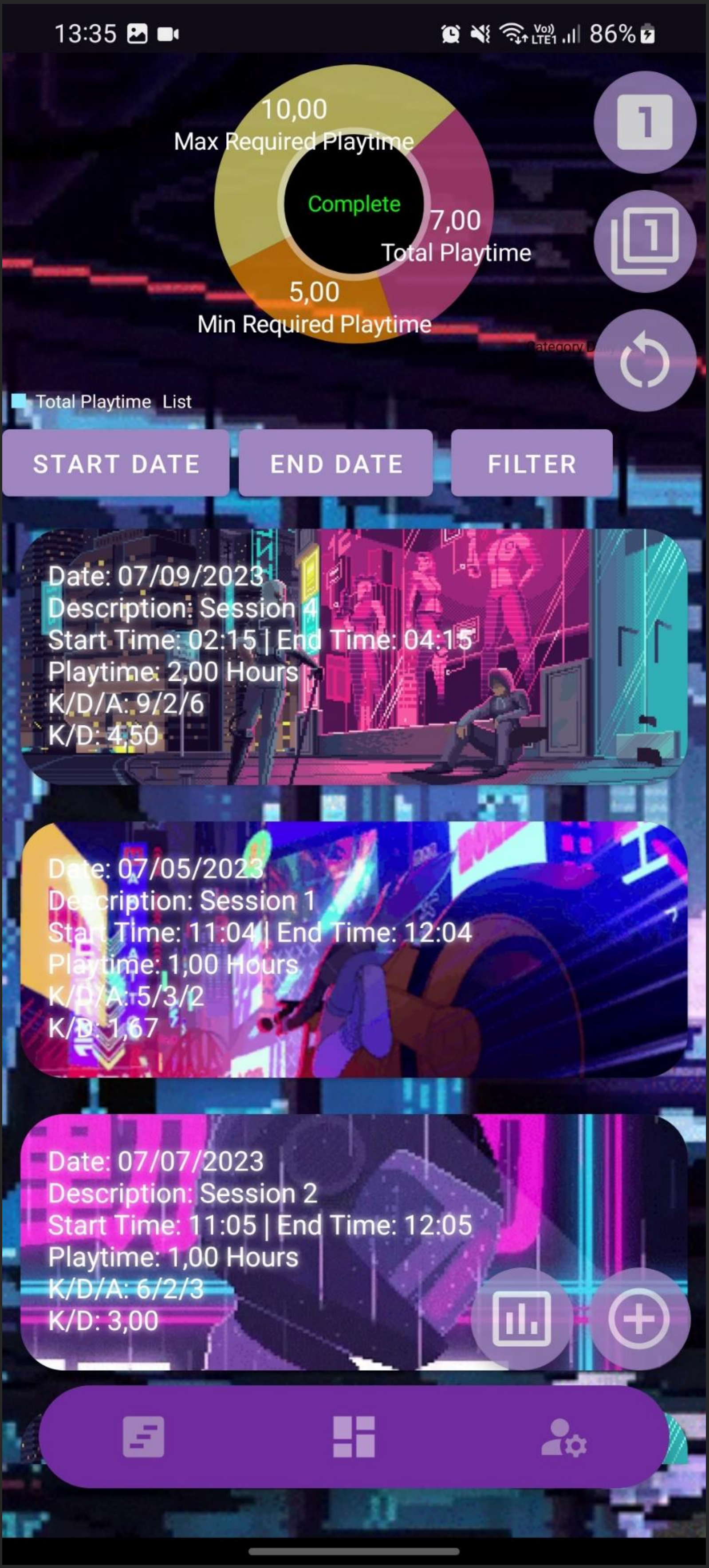
To Filter the data by a Date Range, select your date range using the Start and End Date buttons, then click the filter button. The filtered data will be reflected in the Pie Chart.

To Filter data entered in the previous Month click the First button from the top right corner of the screen.

To Filter data entered in the previous Week click the Second button from the top right corner of the screen.

To Reset the Chart click the Third button from the top right corner of the screen.

Note: This will take you to the default timesheet entry screen, you will need to manually select the timesheet category.



Date: 07/09/2023
Description: Session 4
Start Time: 02:15 | End Time: 04:15
Playtime: 2,00 Hours
K/D/A: 9/2/6
K/D: 4,50

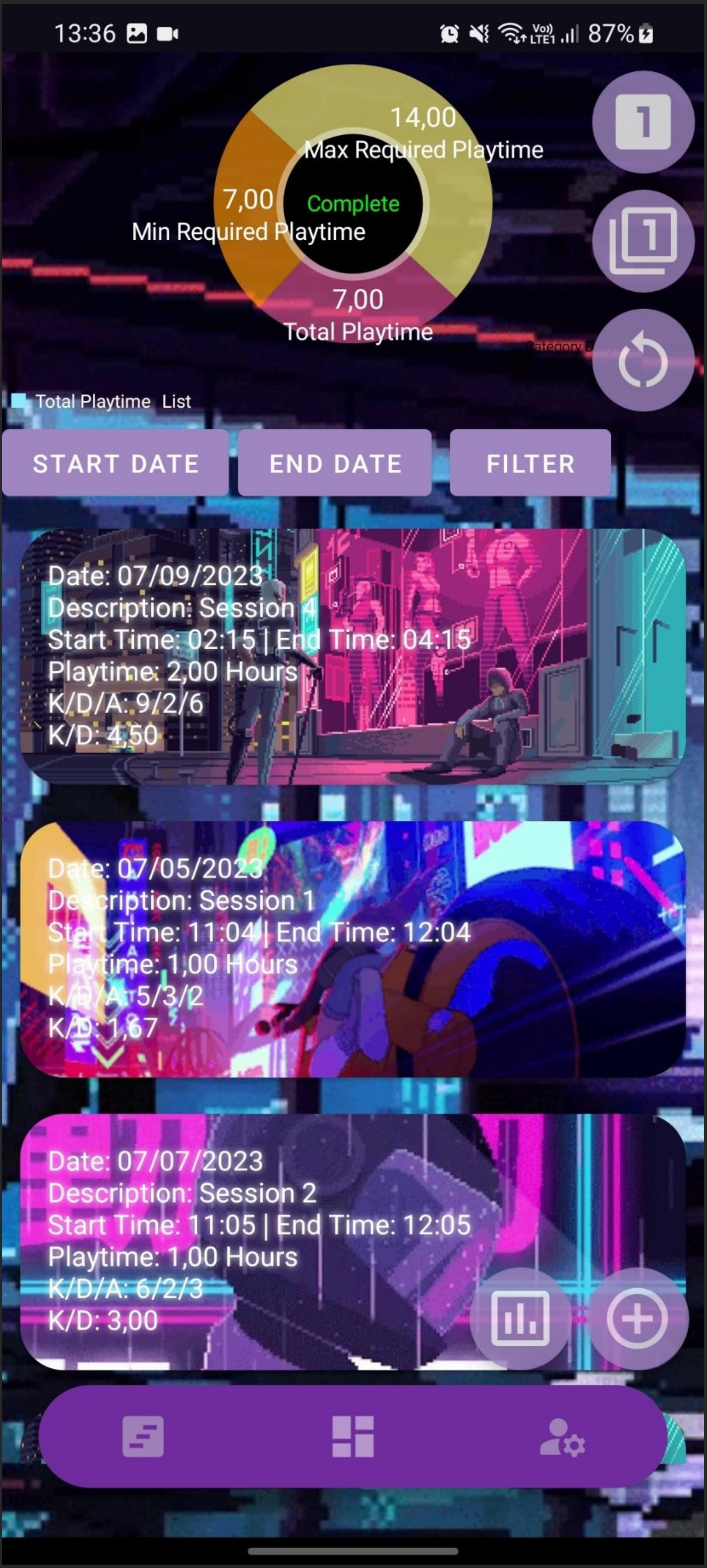
Date: 07/05/2023
Description: Session 1
Start Time: 11:04 | End Time: 12:04
Playtime: 1,00 Hours
K/D/A: 5/3/2
K/D: 1,67

Date: 07/07/2023
Description: Session 2
Start Time: 11:05 | End Time: 12:05
Playtime: 1,00 Hours
K/D/A: 6/2/3
K/D: 3,00

<Fig 1.4.1 Dashboard (Filter by Month) >

Example of the Chart Being Filtered by data entered in the previous month.

Note: The max required and min required hours reflect how many hours the user would of needed to complete within a month, based on the given min and max goals for that specific category.



13:36 87%

START DATE | END DATE | FILTER

Date: 07/09/2023
Description: Session 4
Start Time: 02:15 | End Time: 04:15
Playtime: 2,00 Hours
K/D/A: 9/2/6
K/D: 4,50

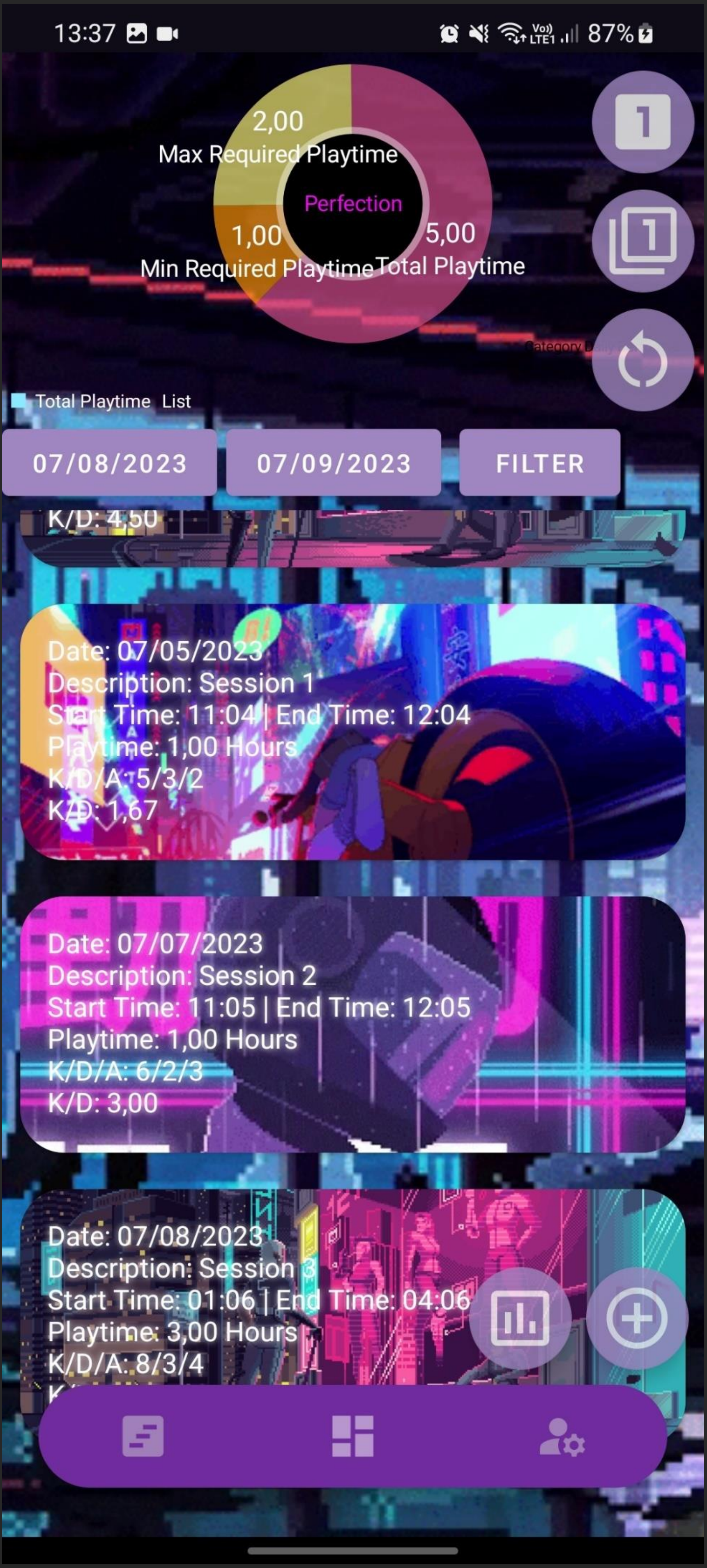
Date: 07/05/2023
Description: Session 1
Start Time: 11:04 | End Time: 12:04
Playtime: 1,00 Hours
K/D/A: 5/3/2
K/D: 1,67

Date: 07/07/2023
Description: Session 2
Start Time: 11:05 | End Time: 12:05
Playtime: 1,00 Hours
K/D/A: 6/2/3
K/D: 3,00

<Fig 1.4.2 Dashboard (Filter by Week) >

Example of the Chart Being Filtered by data entered in the previous Week.

Note: The max required and min required hours reflect how many hours the user would of needed to complete within a Week, based on the given min and max goals for that specific category.



K/D: 4,50

Date: 07/05/2023
Description: Session 1
Start Time: 11:04 | End Time: 12:04
Playtime: 1,00 Hours
K/D/A: 5/3/2
K/D: 1,67

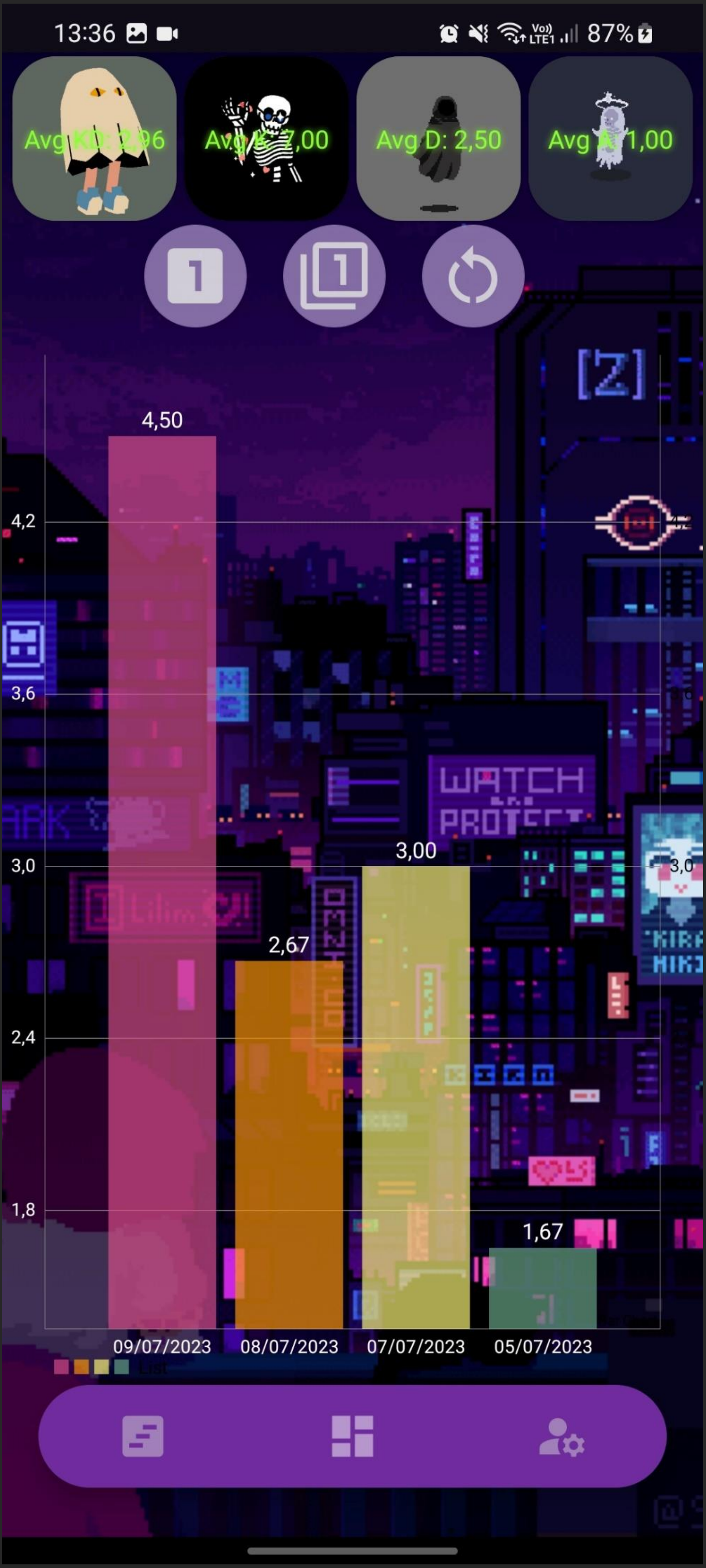
Date: 07/07/2023
Description: Session 2
Start Time: 11:05 | End Time: 12:05
Playtime: 1,00 Hours
K/D/A: 6/2/3
K/D: 3,00

Date: 07/08/2023
Description: Session 3
Start Time: 01:06 | End Time: 04:06
Playtime: 3,00 Hours
K/D/A: 8/3/4

<Fig 1.4.3 Dashboard (Filter by User Date Range) >

Example of the Chart Being Filtered by a User Given Date Range.

Note: The max required and min required hours reflect how many hours the user would of needed to complete within this date range, based on the given min and max goals for that specific category.



<Fig 1.4.4 Dashboard (Game Stats) >

Example of User Game Stats Chart.

Note: The chart displays the K/D Ratios for each given timesheet entry associated with the game the user was viewing, (In this example it was CyberPunk 2077). The chart is ordered in descending order by the timesheet entry date, this means the users most recent entry is placed first. (Further explained in Bonus Features)

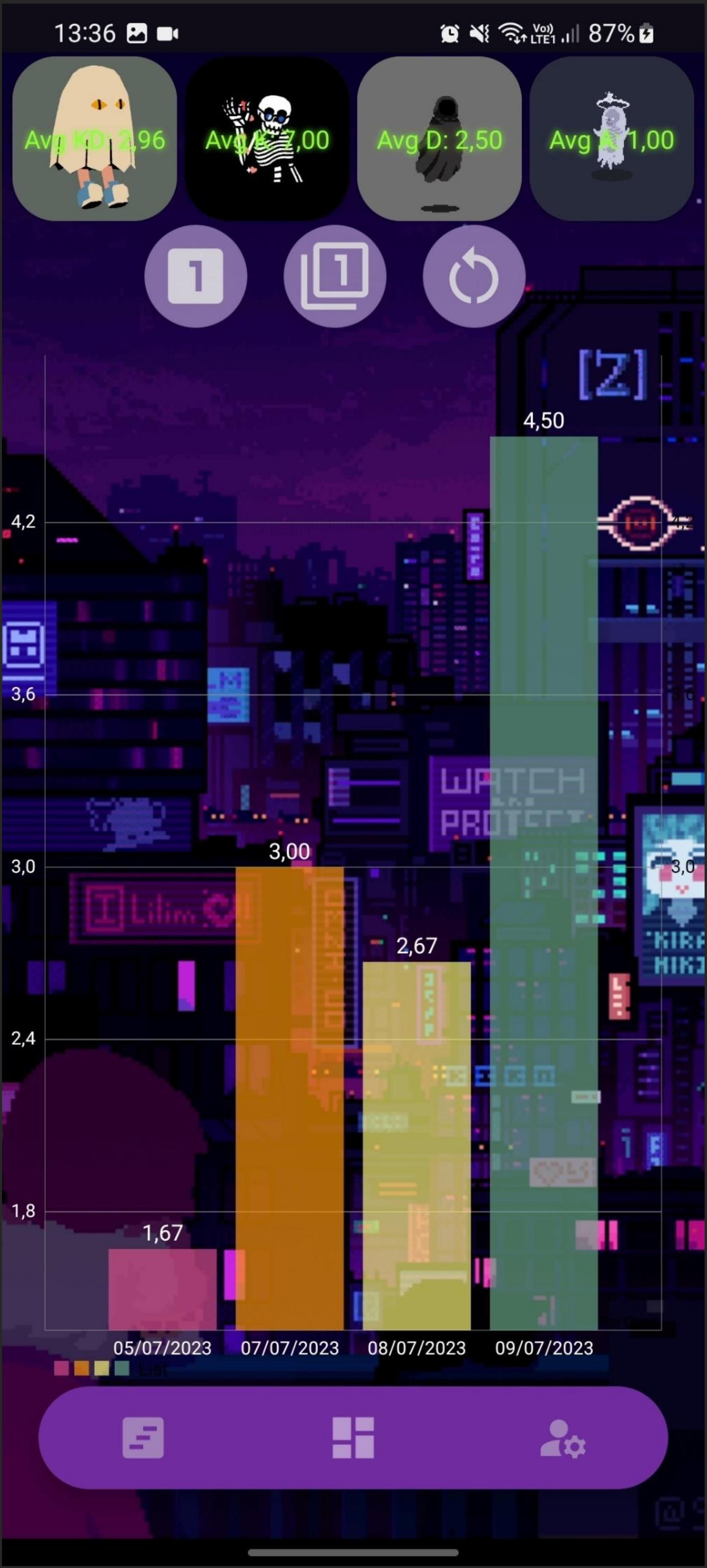
The cards at the top of the screen display Your Average K/D Ratio, Kills, Deaths, and Assists, over the course of all the user's entries for this game.

These values can be filtered by clicking either of the filter options found below.

The first button will filter the chart and the KDA cards to only account for entries that have being made within the last month in their calculations.

The second button will filter the chart and the KDA cards to only account for entries that have being made within the last week in their calculations.

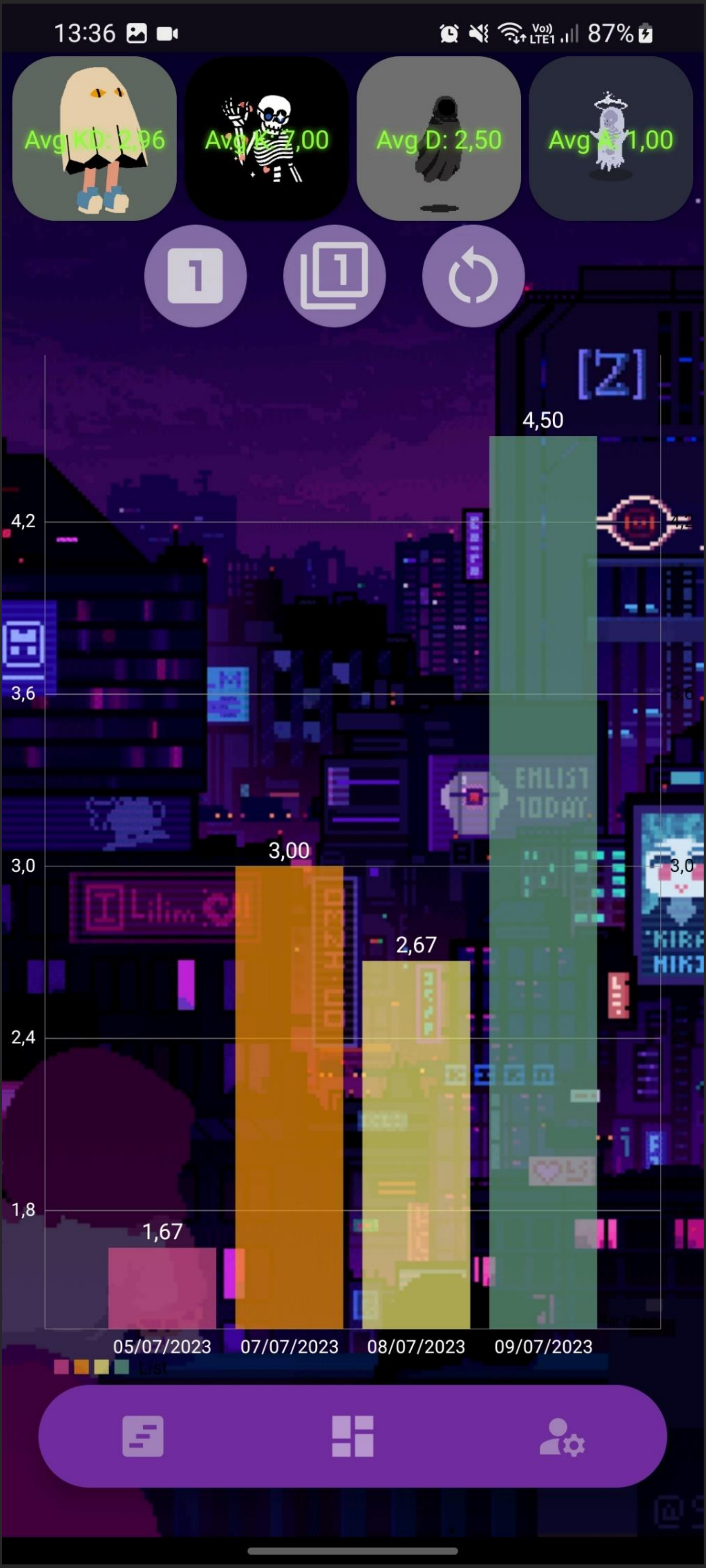
The third button will reset the chart and the KDA cards to their default filters.



<Fig 1.4.4.1 Dashboard (Stats Filtered by Month) >

Example of the Chart Being Filtered by the Previous Month's Data.

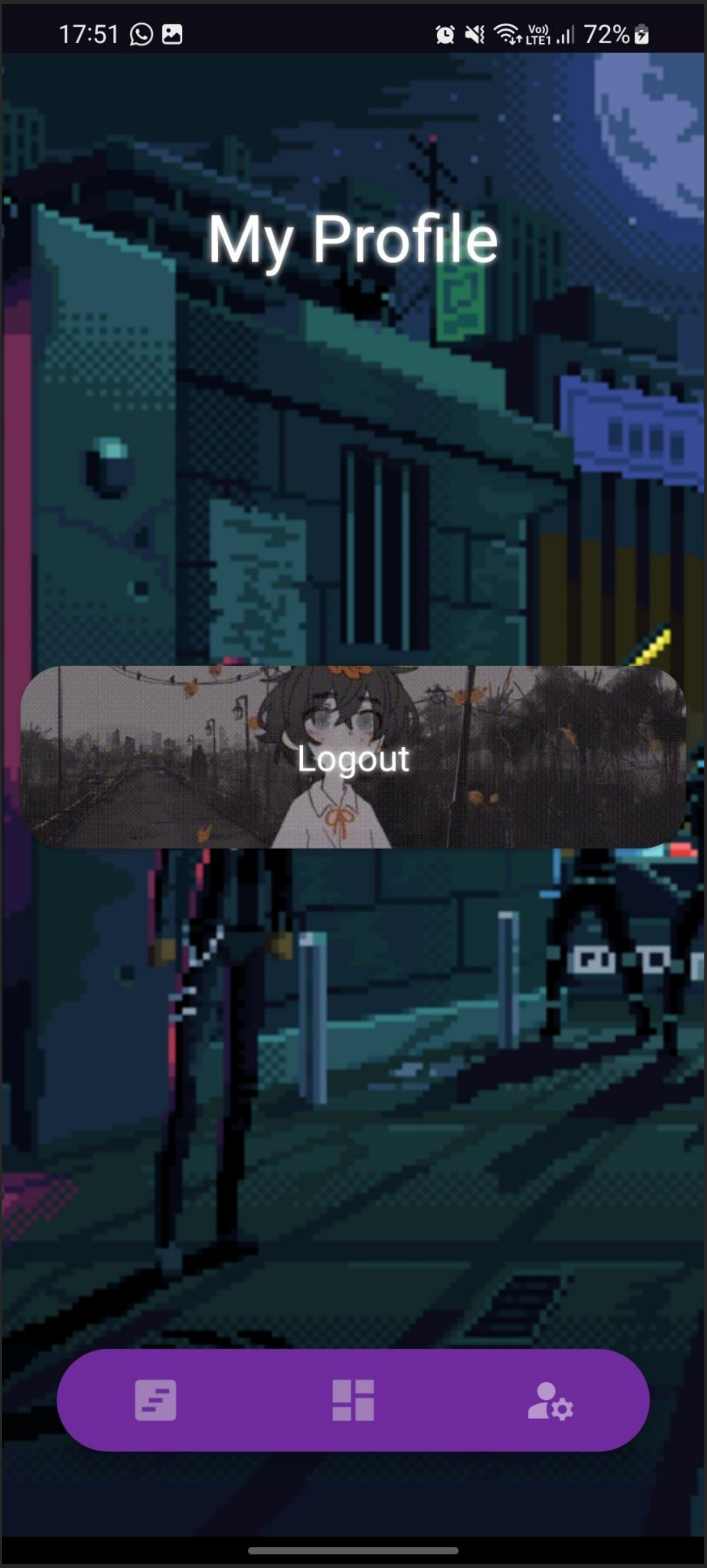
Note: The chart's data is now ordered in ascending order. (Further explained in Bonus Features). The KDA cards stay the same since all our entries were made in the last month.



<Fig 1.4.4.2 Dashboard (Stats Filtered by Week) >

Example of the Chart Being Filtered by the Previous Week's Data.

Note: The chart's data is now ordered in ascending order. (Further explained in Bonus Features). The KDA cards stay the same since all our entries were made in the last week.



<Fig 1.5 Profile (Logout) >

You can logout of your account by clicking the logout image button, this will take you back to the login page.

Note: Login Remember me is always active, unless you logout manually, the app will automatically log you into the previous account you signed into. After logging out, you will need to login again before continuing to use the app.

Bonus Features

Note: If you're interested in a thorough explanation of the thought process and philosophy that went into the design of this app, please view the additional video file I've submitted to GitHub, Bonus_Feature_Review.m4v (Use VLC Media player or Windows Media player to watch it).

Pie Chart Bonus Features

The Pie Chart I designed to display the progress the user has made towards their minimum and maximum goals, is one that can be considered to be rather "Augmented" from the given requirements in the POE document.

How So?

In Summary

On First Load the Pie Chart displayed, is generated using the user's data for that specific category, over the course of their first entry's Date till the current day.

Hence: The system effectively communicates to the user what their current progress should look like according to the min and max goals they set for themselves when creating this category.

Additionally: On the request of the user, the chart can be filtered by data that has only being entered in the previous month, week, or a user specified date range. It will then generate a new pie chart dataset that reflects the given filter's specifications accurately.

In Conclusion

This affords "power users" the ability for micro and macro analysis of progress made in given timeframes that may be important to them, however while doing this, it doesn't undermine the "noob" user who may not understand the importance of such analysis and instead for instance, just wants to see how well they did this week compared to their friend.

Game Stat Bonus Features (Bar Chart and KDA cards)

The Game Stat screen of the dashboard I designed serves two functions, it gives the user a visual representation of their game specific progress, beyond just playtime, and by doing this, it also gives them a conceptual framework to build their idea of progress towards these games (I Highly encourage you to watch the Bonus Feature Review mp4 on GitHub, for a more detailed explanation).

Why does this matter?

In Summary

On First load the bar chart populates data along the X Axis according to individual entry dates (In Descending Order, As stated in the Testing section of this ReadMe), it then plots the corresponding Y values for these X Points according to their entry's K/D Ratio. While doing this, the system uses a separate thread to use the data from the entries that have being loaded to calculate the user's average K/D ratio, kills, deaths, and assists, and then populates the KDA cards with the calculated values.

Hence: The system effectively communicates game specific progress that the user has made, Progress that Playtime alone cannot encapsulate, this system brings an additional layer of depth to the gamification of the app, it does this by giving users a reason to want to do better, as, to put it crudely, it's just human nature to want to see big numbers and pretty colours, get bigger and prettier. (Ironically: Confirmed by focus group testing).

Additionally: On the request of the user, the chart and KDA cards can be filtered by data that has only being entered in the previous month, week. It will then generate a new bar chart dataset that reflects the given filter's specifications accurately, these bar charts will be ordered in ascending order by date, this affords the user the capability to comprehend their progress in given timeframes much more easily. In addition to all of this, the bar chart is fully scalable as seen in my app demo on GitHub, which allows for micro analysis of game statistic progress.

In Conclusion

This affords "power" users the ability for micro and macro analysis of progress made in given timeframes that may be important to them, however while doing this, it doesn't undermine the "noob" users who may not understand the importance of such analysis and instead for instance, just wants to see how well they did this week compared to their friend.

Note: There are many more micro features that have been implemented into the app to improve the overall feel, navigation, and user engagement, these have been discussed in detail in the bonus feature review mp4.

Support

For support please email: sivan.moodley02@gmail.com
If Urgent contact is needed, please phone me on my cell at: 0815750712

GitHub

1.) GitHub POE Part 1 Repo: https://github.com/VCWVL/opsc7311---open-source-coding---part-1-ST10084225

2.) GitHub POE Part 2 Repo: https://github.com/VCWVL/opsc7311---open-source-coding---part-2-ST10084225

3.) GitHub POE Part 3 Repo: https://github.com/VCWVL/opsc7311---open-source-coding---poe-ST10084225

Note: No Kanban added, The VCWVL GitHub Classroom does not support GitHub projects, and therefore no Kanban diagram can be linked to my repos. This is definitely an issue to look into for next semester (hopefully).