

National College of Ireland

BSc in Computing

2016/2017

Name - Ian Arbuckle

Student ID – x13303751

Email – ian.arbuckle@student.ncirl.ie

FitNow

Technical Report



Declaration Cover Sheet for Project Submission

SECTION 1 *Student to complete*

Name: Ian Arbuckle
Student ID: x13303751
Supervisor: Dominic Carr

SECTION 2 **Confirmation of Authorship**

The acceptance of your work is subject to your signature on the following declaration:

I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature: Ian Arbuckle

Date: 8/5/2017

NB. If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the College's Disciplinary Committee. Should the Committee be satisfied that plagiarism has occurred this is likely to lead to your failing the module and possibly to your being suspended or expelled from college.

What constitutes plagiarism or cheating?

The following is extracted from the college's formal statement on plagiarism as quoted in the Student Handbooks. References to "assignments" should be taken to include any piece of work submitted for assessment.

Paraphrasing refers to taking the ideas, words or work of another, putting it into your own words and crediting the source. This is acceptable academic practice provided you ensure that credit is given to the author. Plagiarism refers to copying the ideas and work of another and misrepresenting it as your own. This is completely unacceptable and is prohibited in all academic institutions. It is a serious offence and may result in a fail grade and/or disciplinary action. All sources that you use in your writing must be acknowledged and included in the reference or bibliography section. If a particular piece of writing proves difficult to paraphrase, or you want to include it in its original form, it must be enclosed in quotation marks and credit given to the author.

When referring to the work of another author within the text of your project you must give the author's surname and the date the work was published. Full details for each source must then be given in the bibliography at the end of the project

Penalties for Plagiarism

If it is suspected that your assignment contains the work of others falsely represented as your own, it will be referred to the college's Disciplinary Committee. Where the Disciplinary Committee makes a finding that there has been plagiarism, the Disciplinary Committee may recommend

- that a student's marks shall be reduced
- that the student be deemed not to have passed the assignment
- that other forms of assessment undertaken in that academic year by the same student be declared void
- that other examinations sat by the same student at the same sitting be declared void

Further penalties are also possible including

- suspending a student college for a specified time,
- expelling a student from college,
- prohibiting a student from sitting any examination or assessment.,
- the imposition of a fine and
- the requirement that a student to attend additional or other lectures or courses or undertake additional academic work.

Table of Contents

1. Introduction	1
1.2 Background	1
1.3 Aims	2
1.4 Technologies	2
2. System	4
2. 1 Requirements.....	4
3. Design and Architecture	9
3.1 User class diagram.....	9
3.2 Fitness tracking class diagram	10
3.3 Architecture diagram.....	11
4. Implementation	13
4.1 Overview	13
4. 2 Technologies	13
5. Design Patterns	15
5.1 Model View Controller	15
5.2 Model View Presenter – Passive View.....	15
6. Graphical User Interface (GUI)	17
7. Testing	40
7.1 Manual Testing.....	40
7.2 Automation testing	44
7.3 Continuous Integration.....	46
8. Conclusion	52
9. Further Development or research	53
10. Bibliography	54
Appendix	55
Project Proposal.....	55
Project Plan (Requirements Specifications)	64
Monthly Journals	106
Reflective Journal #1 September	106
Reflective Journal #2 October	107
Reflective Journal #3 November	109
Reflective Journal #4 December	111
Reflective Journal #5 January.....	112
Reflective Journal #6 February.....	113
Reflective Journal #7 March.....	114

1. Introduction

The objective of the software application is to allow the user to track and improve their fitness by using gamification methods. Gamification “is the concept of applying game mechanics and game design techniques to engage and motivate people to achieve their goals (CloudSPM)”. CloudSPM indicates that we can use techniques such as leaderboards to engage and motivate our both competitive and casual users. It is evident that casual users are more interested in comparing their progress to their friends while hard-core users wish to fight for the top spot.

Initial plan was to implement the project on both mobile and wearable platforms. The purpose of the mobile application will integrate a leaderboard. The user will have the option to share their progress with friends via social media (Facebook and/or Twitter) or to prompt for a challenge to beat their record. Whereas the wearable application purpose would be to track the user’s fitness activity and save the dataset to the database. As a result, they can view the overall statistics on the mobile application.

1.2 Background

The background of FitNow is to integrate a social and gamification element to improve our fitness. We all have issues finding ways to motivate ourselves and this is evident per Szczerba who has stated that fitness apps don’t “address the problem of getting people to engage (Szczerba, 2014)”. This further suggests that the most recent apps target only athletes and not casual users who are trying to improve their overall health.

We can try bridge the gap between athletes and casual users by using gamification techniques. As a result, it will help us to engage and motivate users through the idea of status and achievement.

Our peers use social media to connect and share content. We can utilise the benefits of social media to share fitness content and promote their progress via status updates to their peers.

1.3 Aims

FitNow will allow the user to choose their own specific category of exercise combinations:

- Walking
- Running
- Cycling

FitNow will consist of the following features:

- Leaderboard as a gamification technique to create a competitive environment
- Users will have the opportunity to learn from others by viewing their competitor's statistical data
- The user will be able to view their fitness progress such as distance travelled, calories burned, active time to name but just a few.
- Social media integration to allow users to share or invite friends to challenge them
- Socialise their own fitness by adding photos of their surroundings and rating their experience by level of difficulty

1.4 Technologies

1.4.1 Google Fit API

The application will integrate Google Fit API which will allow tracking of the user using sensors of the hardware device. It will provide the capability of storing session information, using the History API and the Recording API will allow to persist storage. The dataset retrieved from the API will be utilised to display statistical information to users in the form of a leaderboard and a result dashboard.

1.4.2 Google Location API

The location API provides functionality to track the user's location and will allow to track the path using within the Google Maps API.

1.4.3 Firebase API

Firebase API is a cloud backend as a service which is a tech start-up firm acquired by Google. It provides all Google APIs packaged into one single SDK. The services include cloud messaging platform, authentication, Realtime database, storage, test lab and crash reporting.

The app will make use of the authentication, realtime database and crash analytics functionality. To integrate a leaderboard, the realtime database would become extremely useful to display a realtime update of a leaderboard.

2. System

This section will outline the systems requirements of the fitness application. It will evaluate the functional requirements, user requirements and usability requirements.

2.1 Requirements

2.1.1 Functional Requirements

1. User registration via Social Network (Facebook, Twitter, Google+) to collect personal data
2. Selecting specific fitness category – Walking, Running or Cycling
3. Recording exercise activity by using GPS location tracking and detecting health for calories burnt, speed per k/m etc.
4. Viewing dashboard for finished exercise – Fitness health statistics, google map statistics and sharing media for photos
5. Viewing leaderboard to view opponents and send app invites to other users to use the app
6. Viewing your fitness activity – Show past fitness exercises to share with friends or analyse the dashboard to view progress

2.1.2 User Requirements

The user requirements of FitNow is to integrate a social and gamification element to improve our fitness. To solve the problem, we can achieve this by using gamification techniques that will help us to engage and motivate users through the idea of status and achievement.

Per Vironit, majority of the best fitness apps “track, take measurements, analyse and the motivate the user (Vironit.com, 2017)”. This indicates if you decide to use an app you can avoid can personal instructors or training programs to monitor your own fitness activity progress.

Therefore, we can assume that users may expect the following from a fitness app:

- Ability to motivate themselves by challenging and learning from other end users
- Socialise their fitness experience by sharing with friends of photos, statistics and map of distance path travelled
- Learning from users on what equipment, level of difficulty, and distance travelled to improve their own fitness by viewing their statistics via the leaderboard
- Invite friends to use the app to encourage competition

2.1.3 Usability Requirements

This section will highlight and evaluate the usability requirements. It is important to make your application accessible to all users and especially those with a disability. Google has designed a usability and accessibility framework, Material Design. The application will use Material Design principles to enable users to “navigate, understand and use the user interface successfully (Accessibility - usability - material design guidelines, no date).”

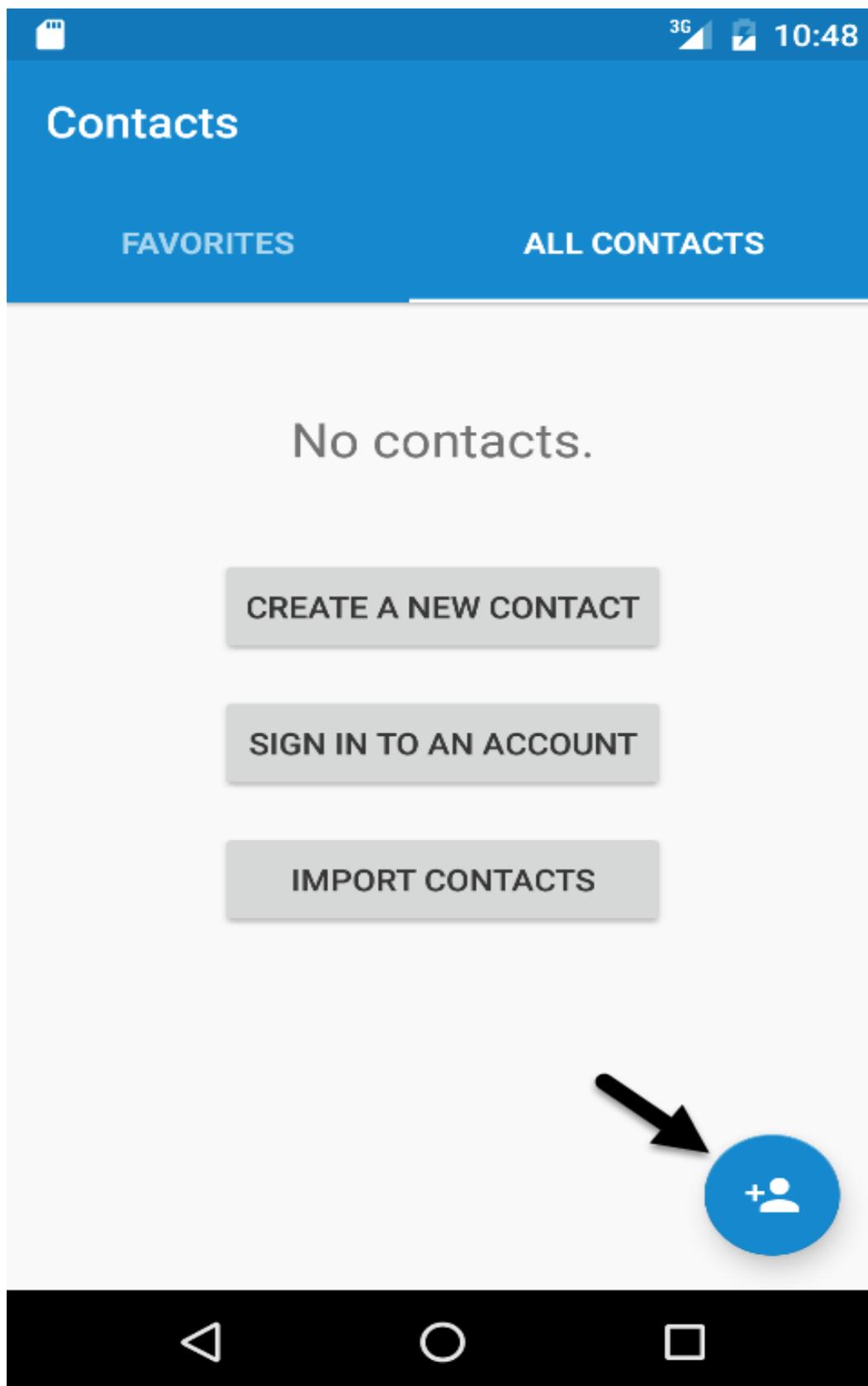
The principles of material design are to provide “accessibility to all users of all abilities, including those with low vision, blindness, hearing impairments or cognitive impairments (Accessibility - usability - material design guidelines, no date).”

It adheres to the principles as follows below:

Clear

Help users to navigate the app by designing clear layouts with descriptive calls to action as shown in figure 1. In the figure below we see a call to action on the floating action button with an icon which indicates to add a new contact.

Figure 1

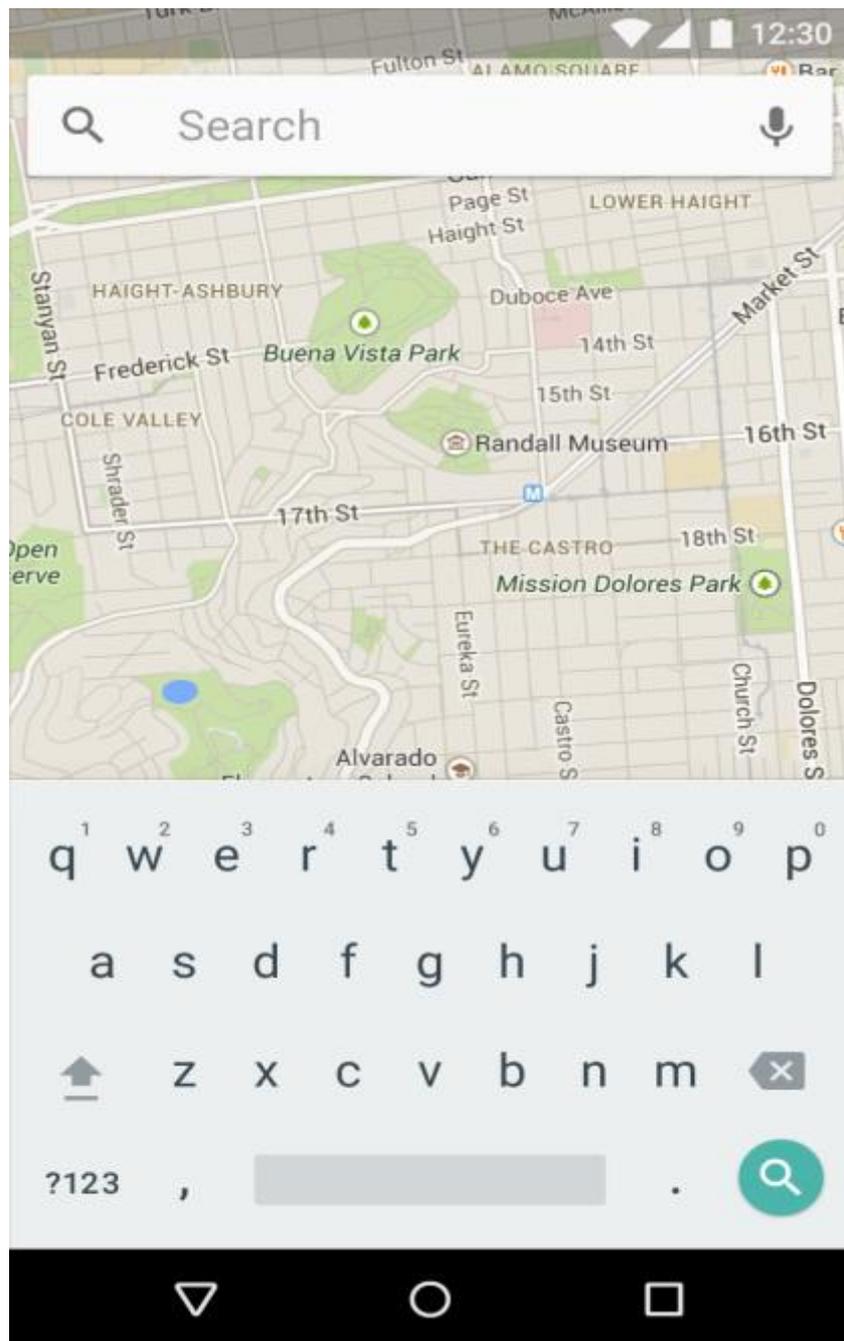


Robust

The application should accommodate all variety of users who may short attention span, new to a product or use a screen reader. Therefore, the app will ensure the user can:

- Navigate the app with confidence and aware of what's important
- Reinforce the users of what's important by using visual and textual cues as shown in figure 2

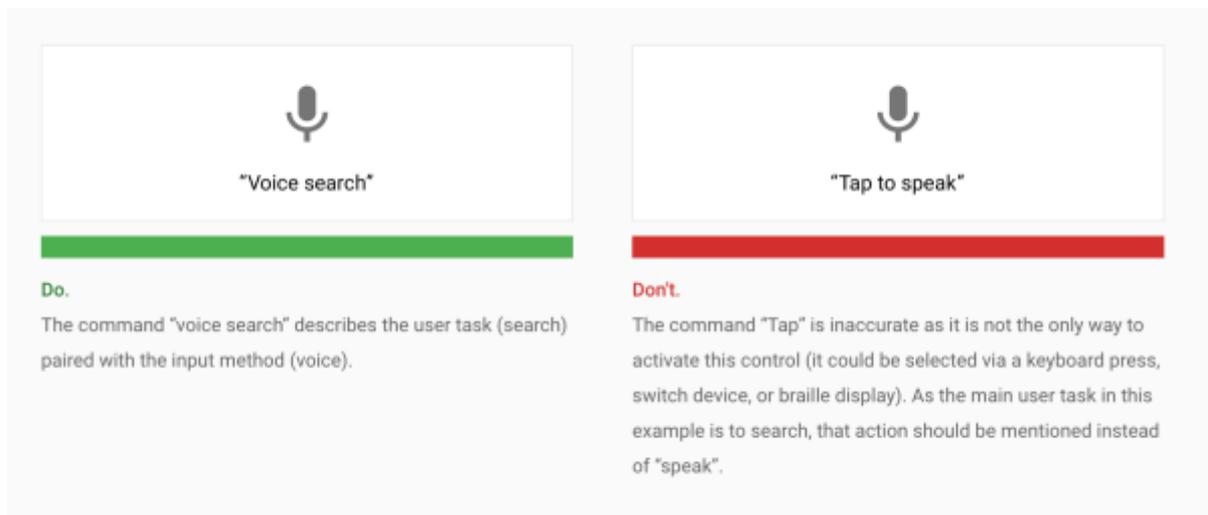
Figure 2



Specific

Using assistive technology to support input methods of touch, keyboard and mouse. It will help increase and improve the functional capabilities of the user with disabilities. For instance, in figure 3 it is considered good usability practice not to mention the exact gesture or interaction but instead be specific about the action.

Figure 3

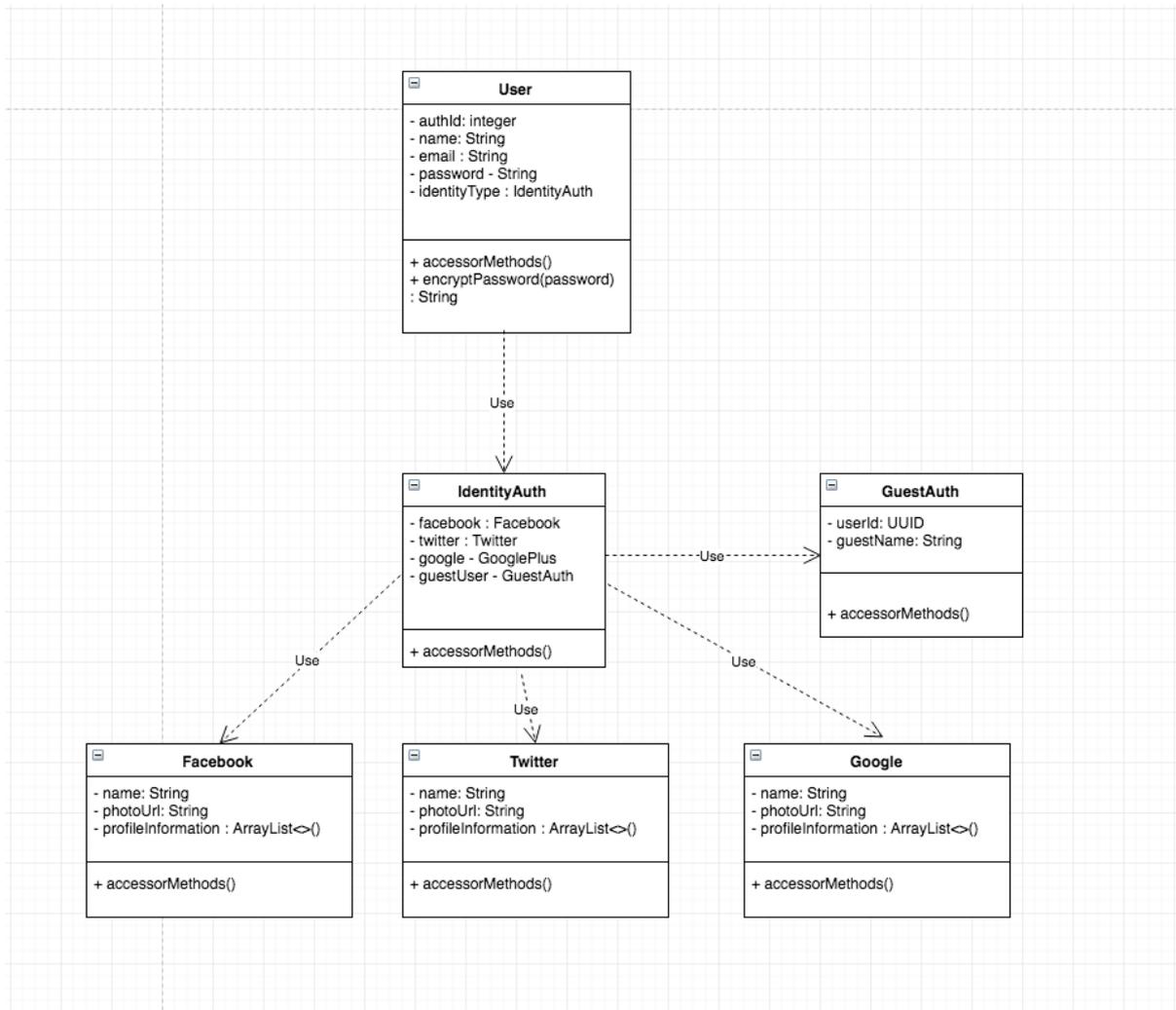


3. Design and Architecture

This section will describe and provide diagrams to demonstrate the architecture and design decisions of the project.

3.1 User class diagram

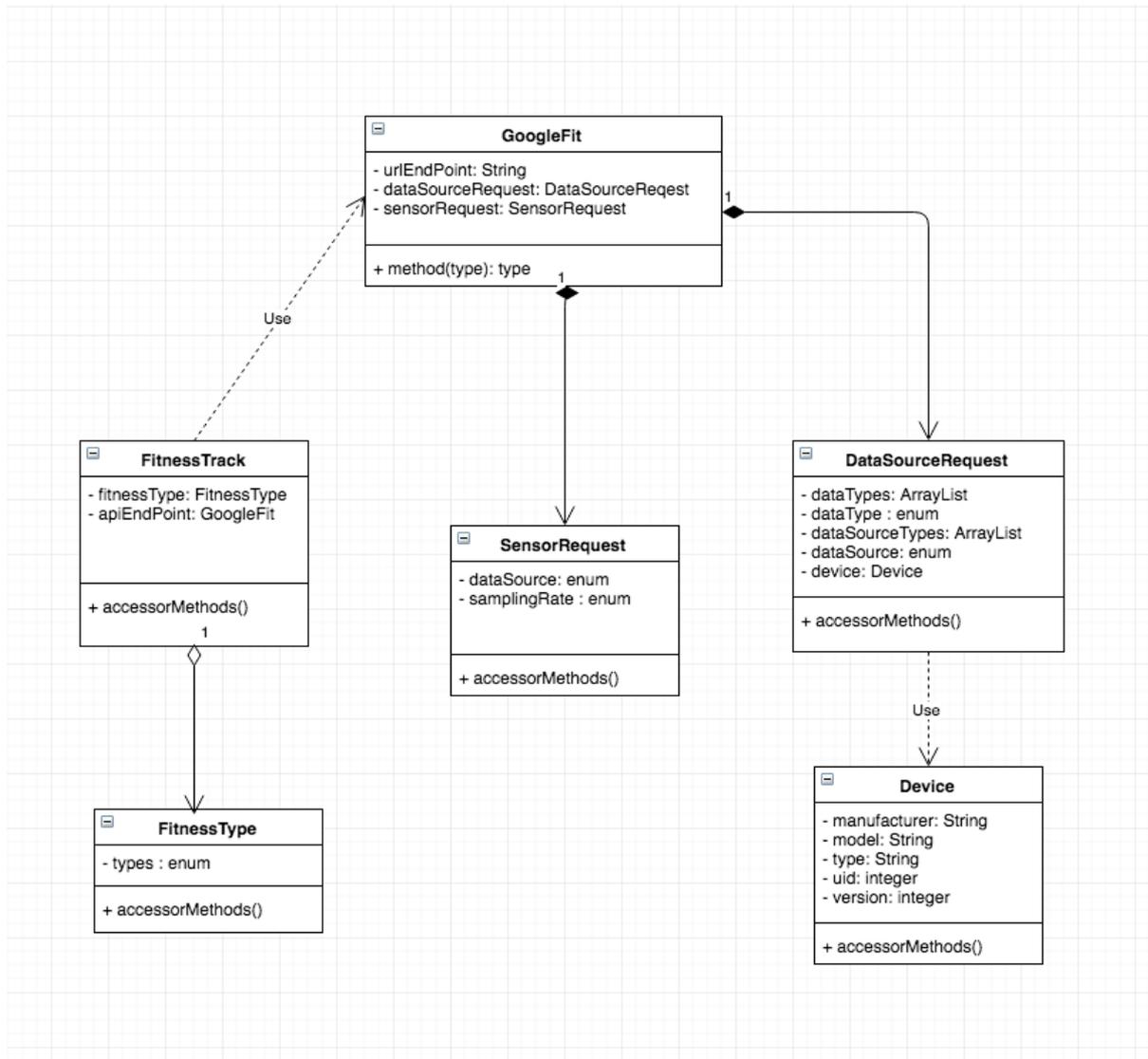
Figure 4



- User class depends on IdentityAuth to retrieve user account authorisation such as Google/Facebook
- IdentityAuth depends on Facebook, Twitter, Google and GuestAuth classes to retrieve profile information to authenticate the user

3.2 Fitness tracking class diagram

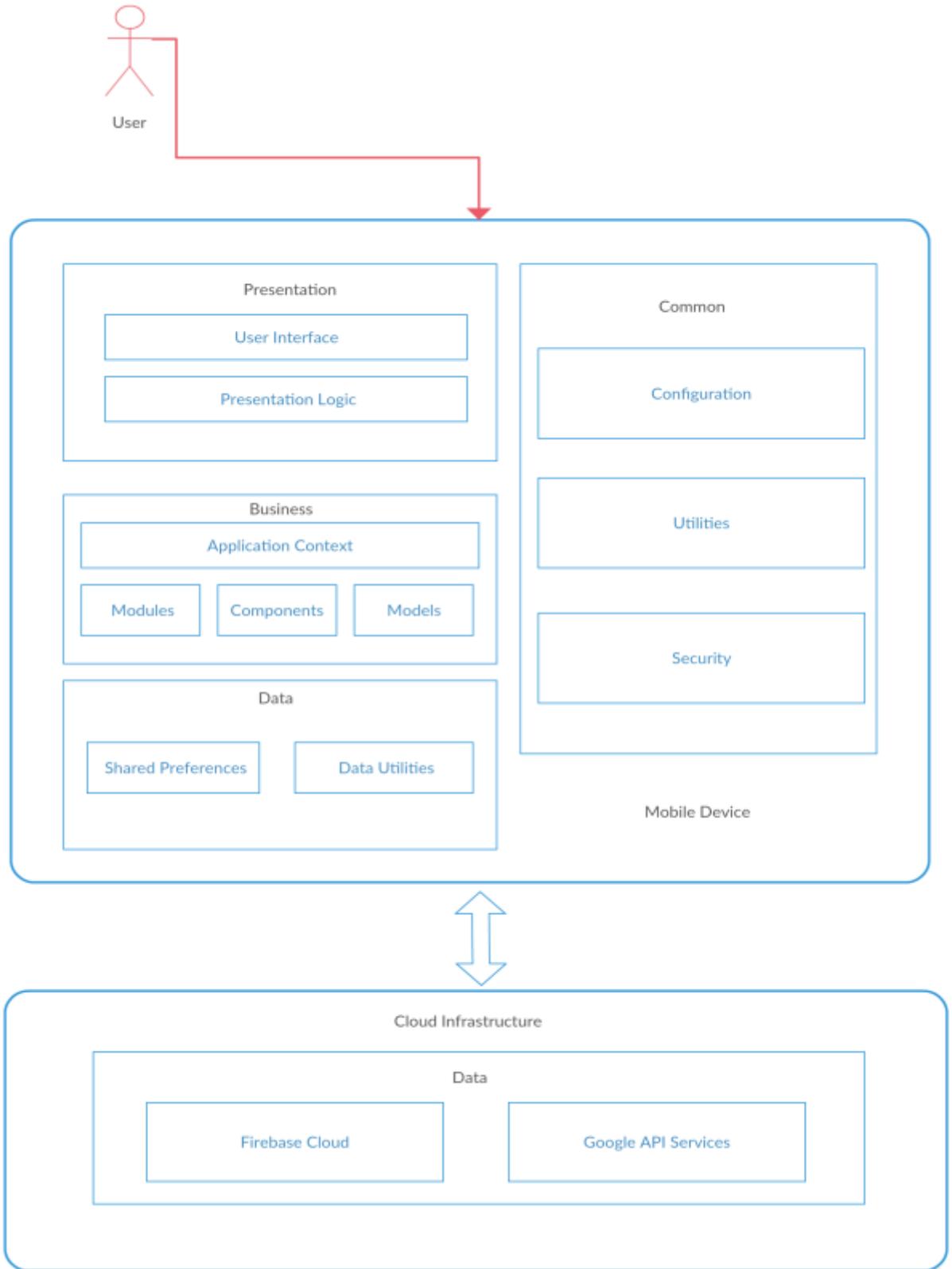
Figure 5



- FitnessTrack class depends on GoogleFit for the REST URL end-point and requests.
- FitnessTrack uses FitnessType class to get access to fixed constant data types. These types can be used to describe a fitness type such as RUNNING, WALKING and CYCLING
- GoogleFit class is a composite of SensorRequest and DataSourceRequest and it is a “strong-type” relationship meaning that they cannot exist independently.
- DataSourceRequest uses a dependency relationship where it depends on the implementation of the Device class

3.3 Architecture diagram

Figure 6 – App Architecture Diagram



- Presentation layer contains the user interface and contains the presentation logic that interacts with the UI
 - Business layer contains the business logic that is broken down into application context, modules, components and modules
 - Data layer contains the user's shared preferences on the mobile device and any utilities from the user's device such as Samsung touch wiz for example.
 - Common contains configurations, utilities and security features of the user's device
- Cloud Infrastructure container contains the Firebase Cloud and Google API Services components

4. Implementation

The object of this section is to describe and evaluate the technologies and design patterns used in the implementation of the mobile application for Android.

4.1 Overview

It was decided to take the approach to build a native Android application using Java with Android Studio IDE. The reason for the decision to take the native Android approach is because it's faster in terms of performance than other mobile app hybrid frameworks such as HTML5 and React Native. Another reason would be that a native app can access a wide range of hardware on a device such as location, camera, accelerometer and much more. While a hybrid app does not have the same ability to interact with the hardware.

Therefore, the app will take advantage of the native features provided by Google's Firebase platform such as authentication, real-time database and storage.

4.2 Technologies

4.2.1 Android Studio IDE

Android Studio is the official Android integrated development environment that is supported by Google and built by JetBrains team (the creators of IntelliJ). It offers a range of tools and features from refactoring, instant run and intelligent code editor.

There are alternatives to Android Studio such as Eclipse but it is not officially supported by Google anymore. It is recommended to use the official IDE platform to receive the most recent Android SDK components.

4.2.2 Firebase

Firebase is a development SDK framework provided by Google which allows developers to grow their apps without writing a single line of server-side code. It was the ideal backend solution as the application can be extended to iPhone devices using the same backend infrastructure, thanks to Firebase's portability.

Firebase offers a range of development tools and the app utilised the following:

- **Authentication**

Authentication services such as email/password and social login all in one single SDK

- **Realtime Database**

Provided app data in realtime for storing and syncing fitness data from our users.

- **Cloud Storage**

It served as a content management for storing media such as photos from our users.

4.2.3 Google Fit API

Google Fit API is deeply integrated with the Google Play Services. It provides several APIs and the app utilised the sensors API. This allows us to access sensor data from the Android device and wearables. The figure below describes the architecture of the Google Fit API services.

5. Design Patterns

It was necessary to utilise design patterns for the complexity of the application. After conducting research in design patterns for the Android platform it was decided to favour the Model View Presenter passive view pattern which is a derivative from the MVC (Model View Controller) pattern.

5.1 Model View Controller

In MVC, the controller is responsible for determining which view is to be displayed. Therefore, every action in the view is executed via a call to the controller. The view simply renders and is regarded as stateless.

Pro

- There is separation of concerns, making the code more maintainable

Con

Android is tightly coupled together and the View will end up containing business logic which reduces your unit test coverage

5.2 Model View Presenter – Passive View

Model View Presenter is a popular design pattern within the Android community as it allows us to separate activities, fragments and UI widgets away from the business logic. In passive view, the View is dumb as possible allowing for it to contain zero logic. The presenter acts as the mediator (middle man) by sending the presentation logic to the views and updating the models of any changes.

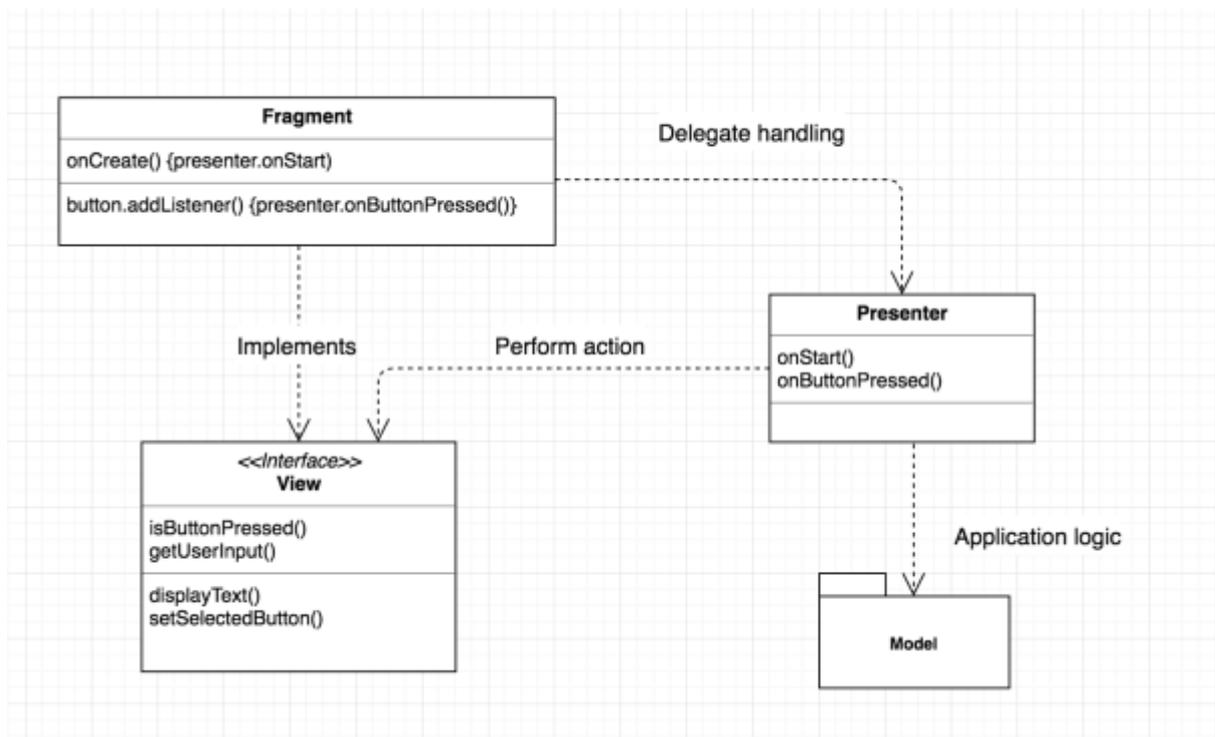
Pro

- Increased testing coverage as there is separation of the View and Model

Con

- In the model, there is more work required as you will be covering the data binding yourself

Figure 7 – MVP Class Diagram



- Activity/Fragments contains only references to Android view objects
- View Interface is implemented by the Activity/Fragment
- Presenter performs the business logic
- Model is the container of the application logic

The decision to use the MVP passive view design pattern allows us to decouple a lot of the business logic away from activities/fragments. As a result, it has made unit testing a lot easier. It provides the ability to test the presenter logic without any overhead from the network and UI.

6. Graphical User Interface (GUI)

6.1 Login

Figure 8 – Old login design

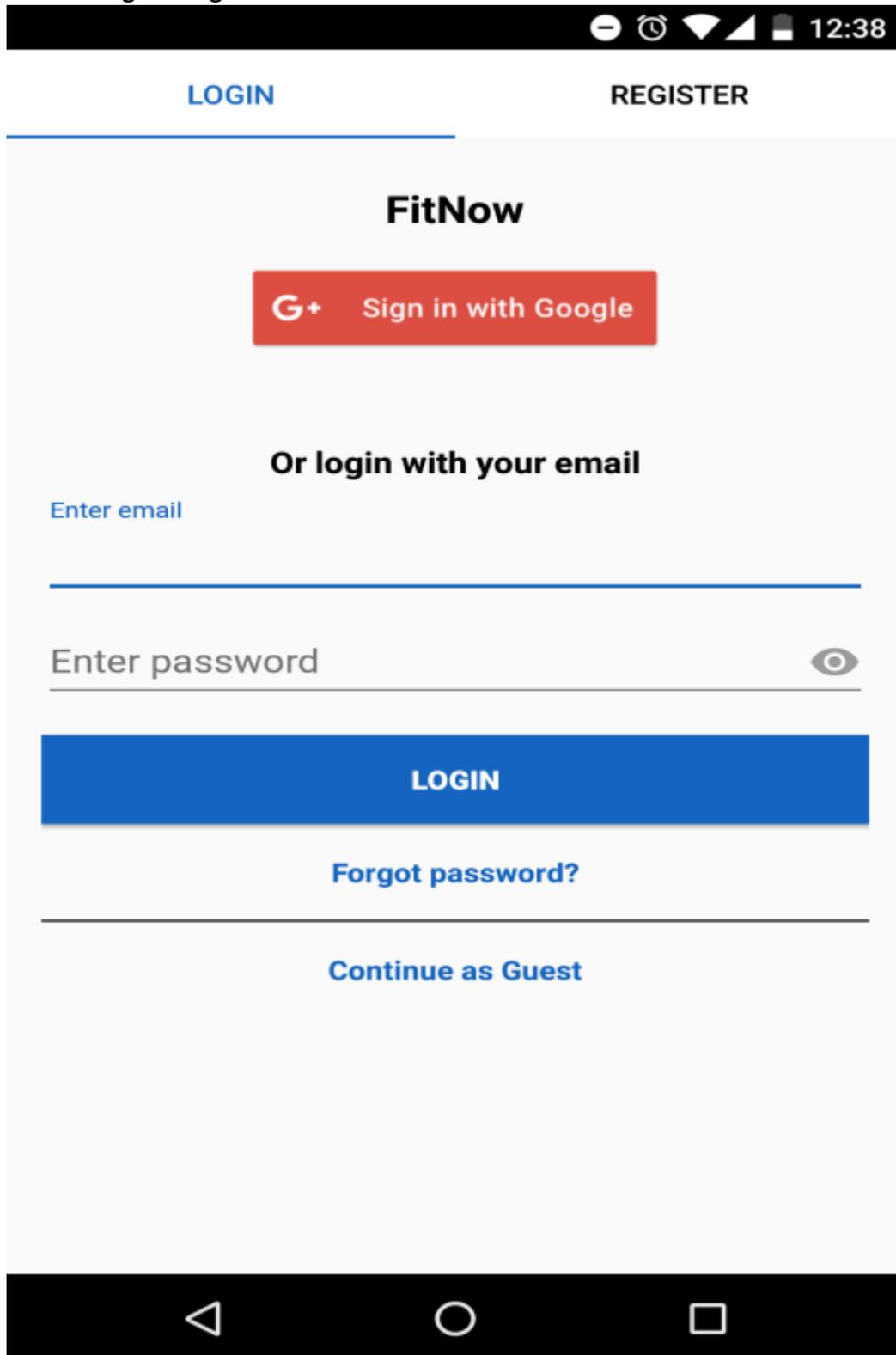
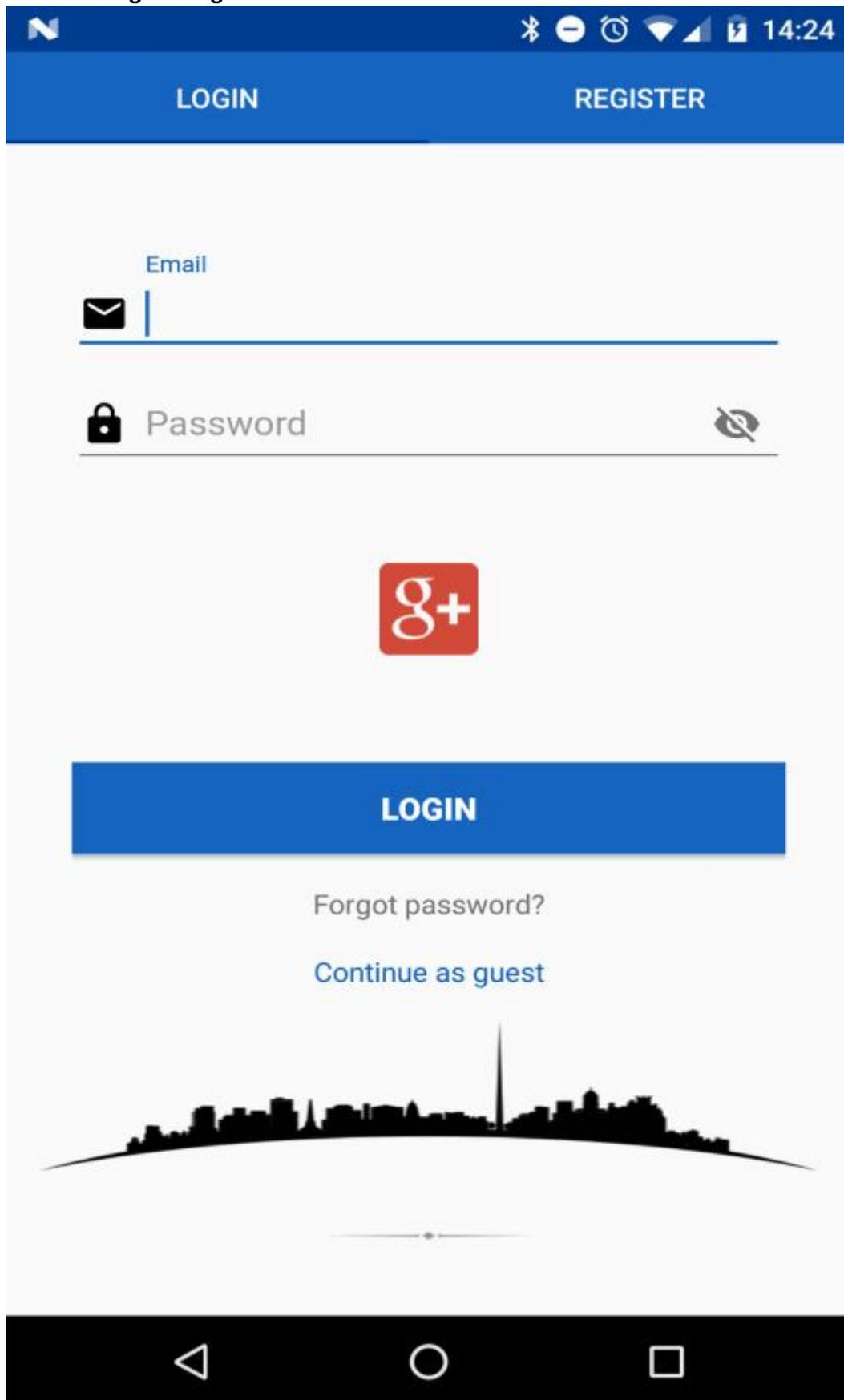


Figure 9 – New login design



The designs above were used to describe the login and registration use cases.

The first attempt in figure 8 didn't go well as planned in terms of user experience. There was too much text information and it disregarded the main functionality of the screen. Which was to login with ease.

The new design in figure 9 attempted to clean up the screen and put more attention to call to action elements such as entering email and password. As a result, it allowed for a consistent user experience in contrast to the first attempt where too much textual information was provided.

6.3 Registration

Figure 10 – Old registration design

LOGIN REGISTER

Create an account

Enter email

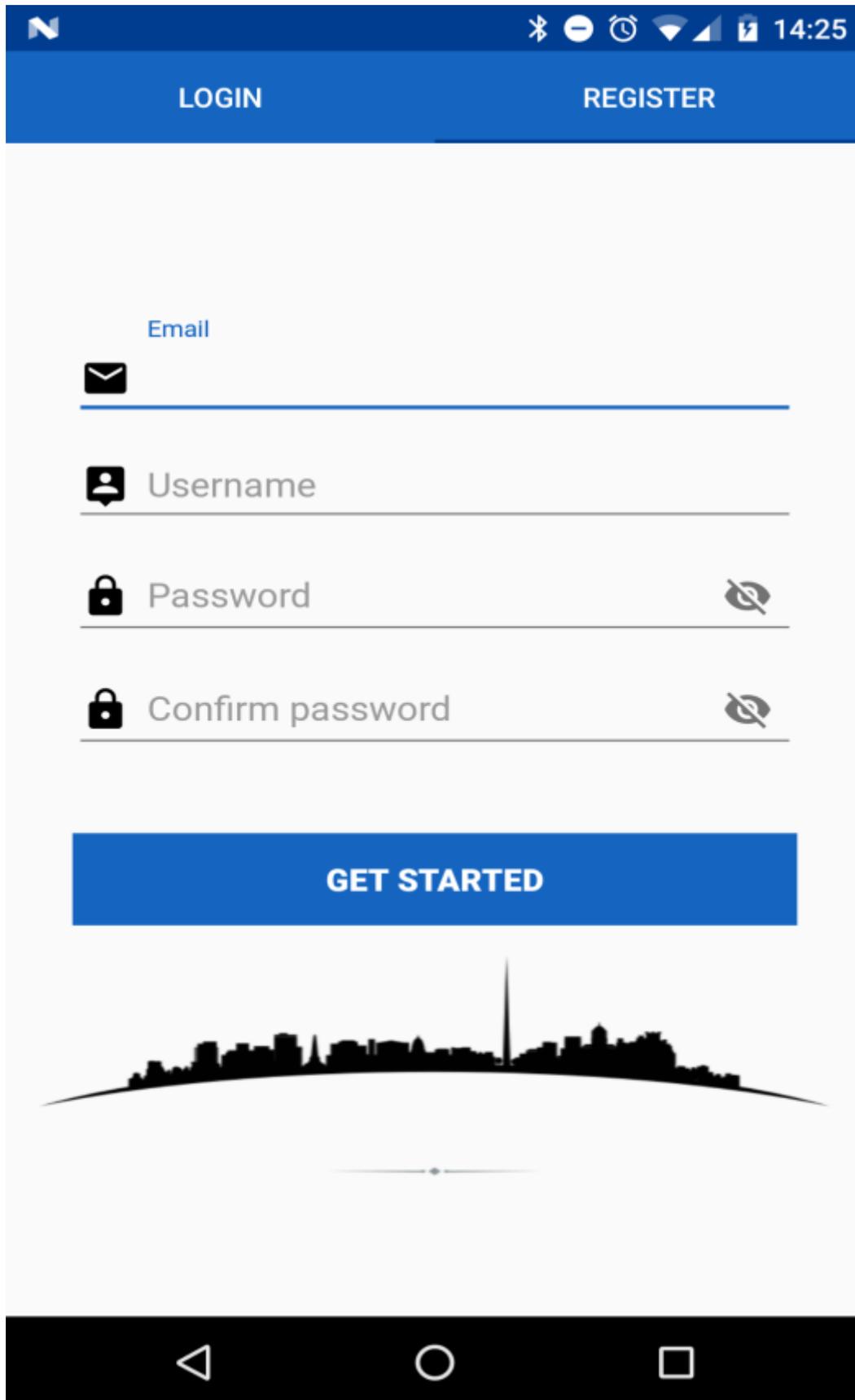
Enter name

Enter password 

Confirm password 

CONTINUE

Figure 11 – New registration design

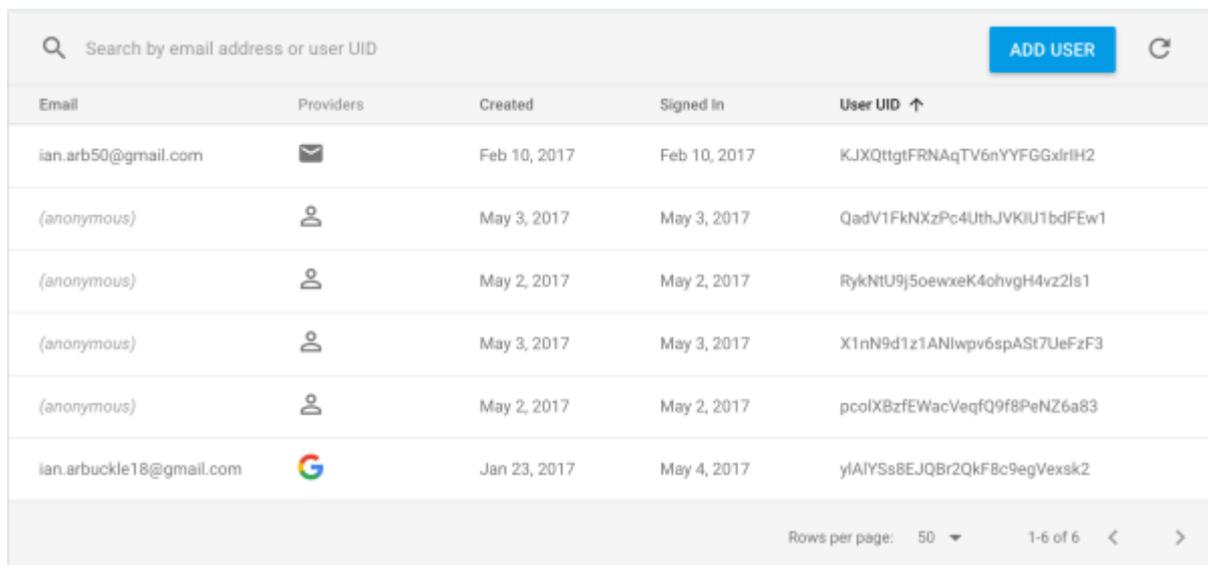


This design allows the user to create a standard account for identity authentication without using a social network account. The sharing features will not be available unless the user decides to add a social account later.

There is one small modification made in the designs above. In figure 11 we've made changes to the colours of the tabs and removed the textual information to put immediate focus on entering the details.

The registration process is handled by Firebase Authentication and the below diagram shows the dashboard of successfully authenticated users stored in the app.

Figure 12 – Authentication database dashboard



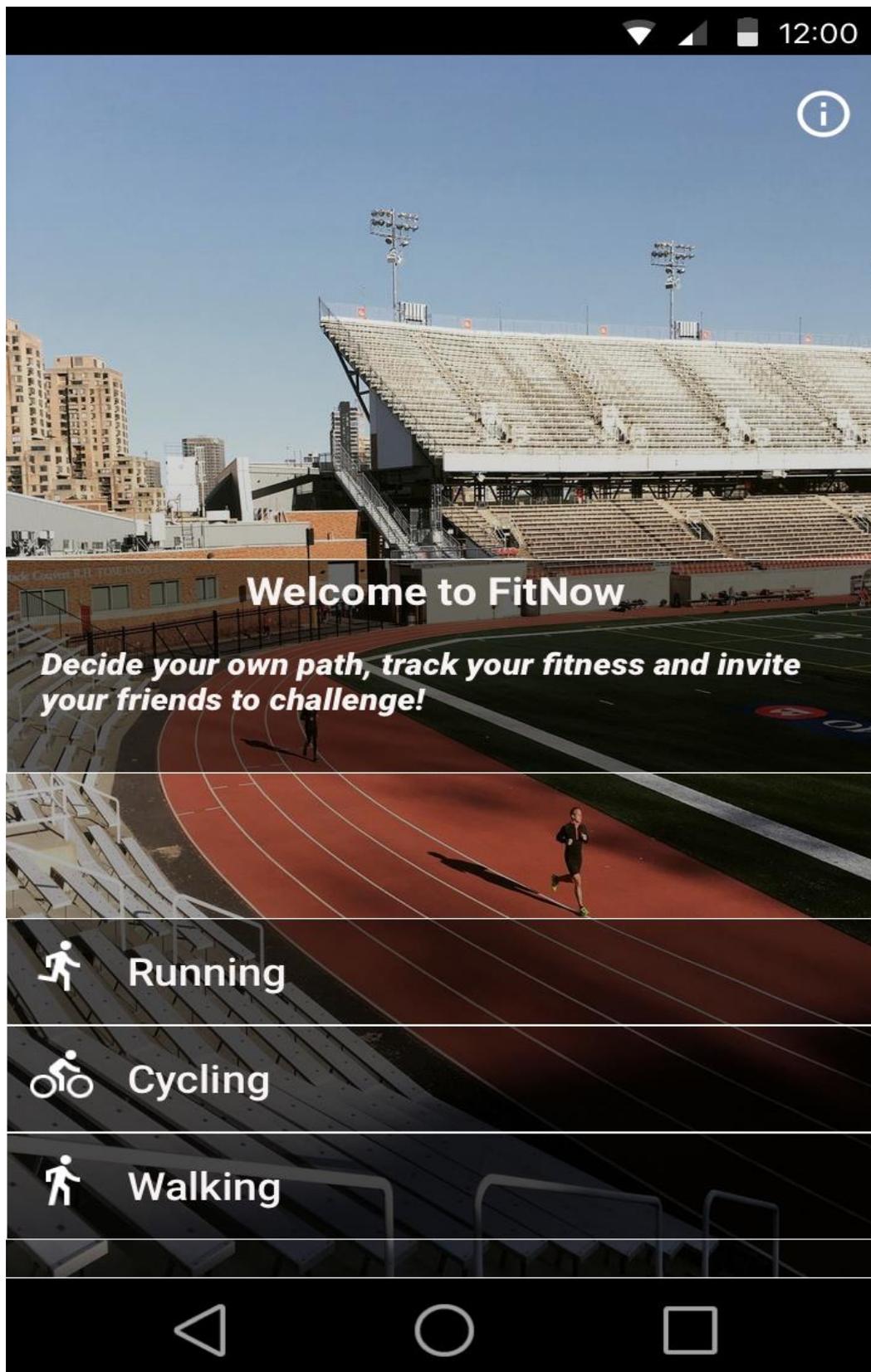
Email	Providers	Created	Signed in	User UID ↑
ian.arb50@gmail.com		Feb 10, 2017	Feb 10, 2017	KJXQttgtFRNAqTV6nYYFGGxlrH2
(anonymous)		May 3, 2017	May 3, 2017	QadV1FkNXzPc4UthJVkIU1bdFEw1
(anonymous)		May 2, 2017	May 2, 2017	RykNtU9j5oewxeK4ohvgH4vz2ls1
(anonymous)		May 3, 2017	May 3, 2017	X1nN9d1z1ANlwpv6spASt7UeFzF3
(anonymous)		May 2, 2017	May 2, 2017	pcoIXBzfEWacVeqfQ9f8PeNZ6a83
ian.arbuckle18@gmail.com		Jan 23, 2017	May 4, 2017	yAIYsS8EJQBr2QkF8c9egVexsk2

It takes the following input:

- Username as a unique identifier to recognise the user (User UID)
- Name to display
- Email
- Password to authorise the user

6.4 Home screen

Figure 13 – Home Screen



The user is presented the home screen when they are successfully authenticated. They can specify their own activity exercise from 3 specific categories; running, cycling and walking.

The importance of designing a home screen is to provide a welcome text with a small description that triggers a call to action. This is evident as the description asks the user to invite their friends to try the app.

There have been no significant updates to the home screen designs as it has been kept simple and clean.

6.5 Recording exercise screens

Figure 14 – Old start design

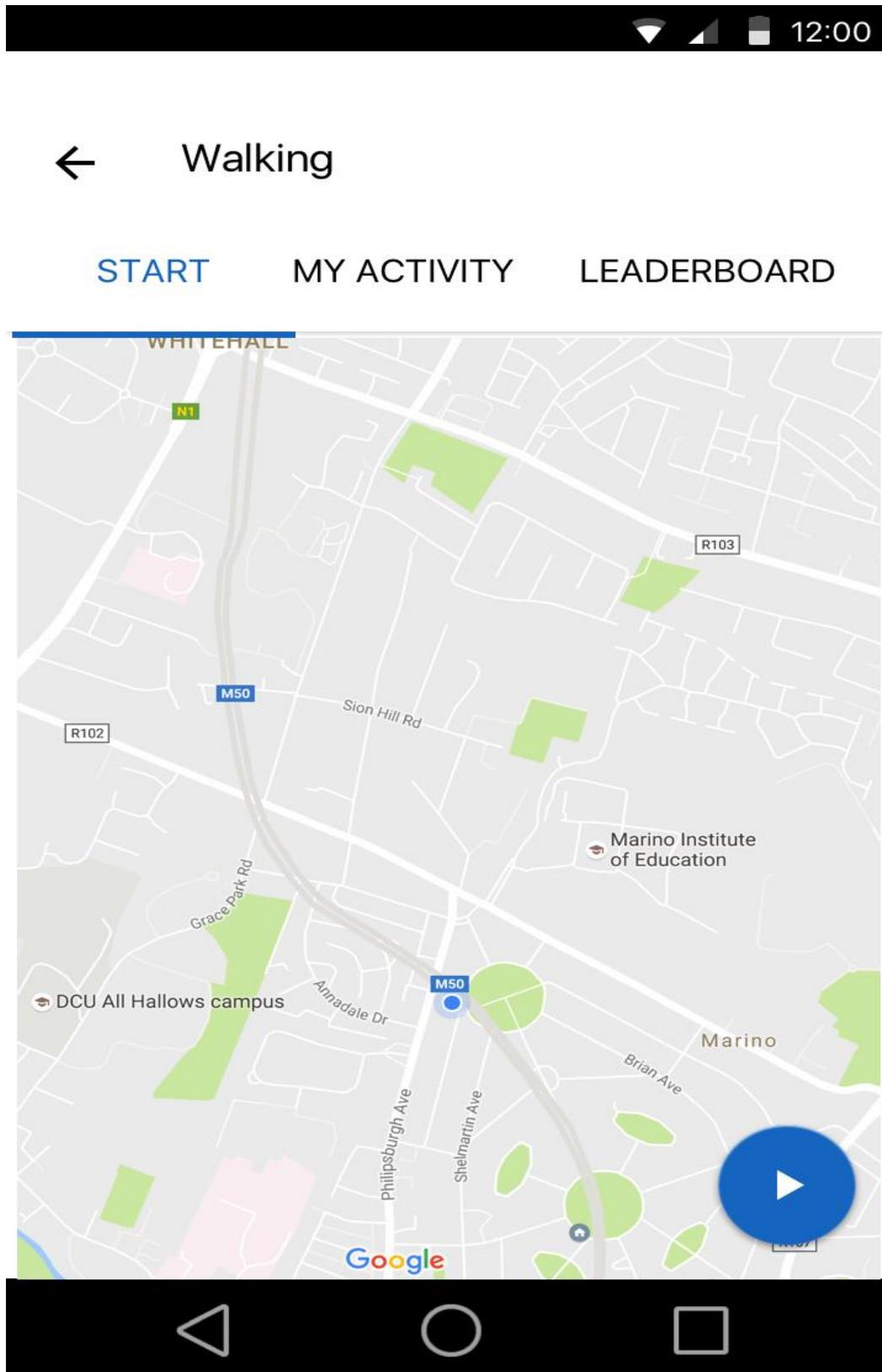
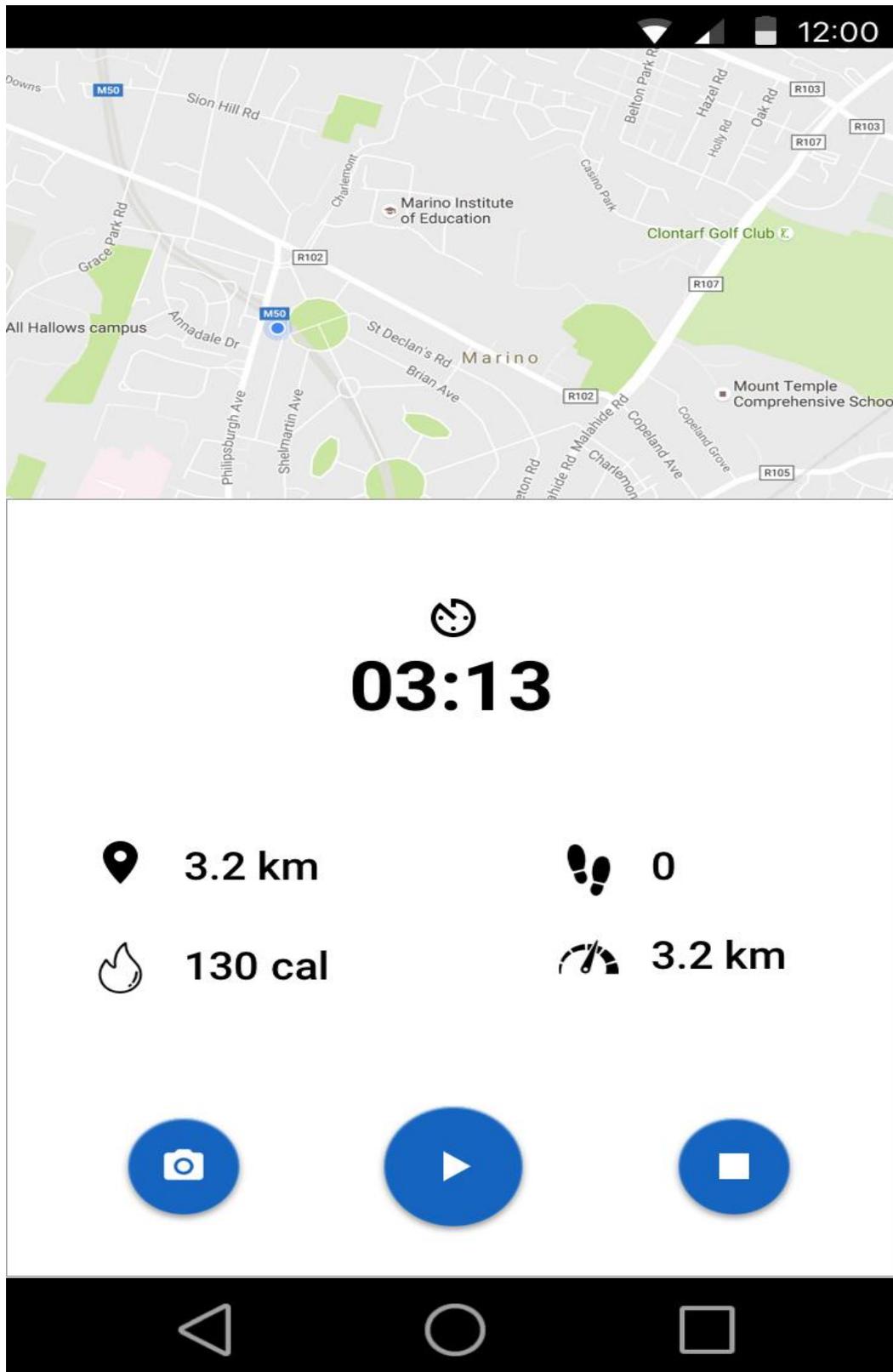


Figure 15 – Recording activity design



In figure 15, the user's exercise is recorded in real-time using their device's sensors to detect their progress. They can pause the session to take a photo of their surroundings and stopping the exercise to end their activity.

It has been decided to remove figure 14 as it was providing an inconsistent navigation pattern. Therefore, it was decided to replace it with my activity section which will be outlined later.

6.6 Finished activity screens

Figure 16 – Old Results screen

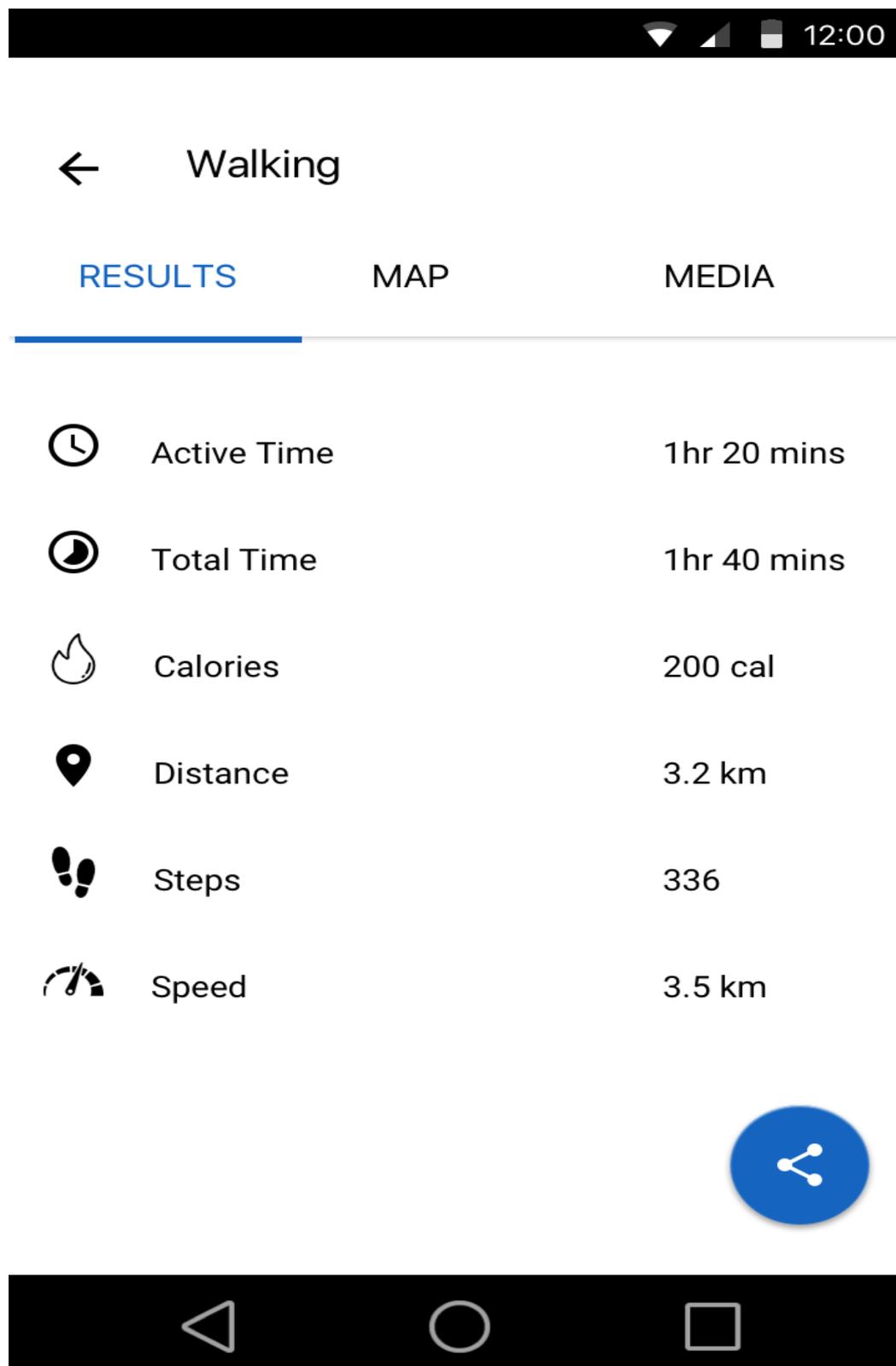


Figure 17 – New Results screen

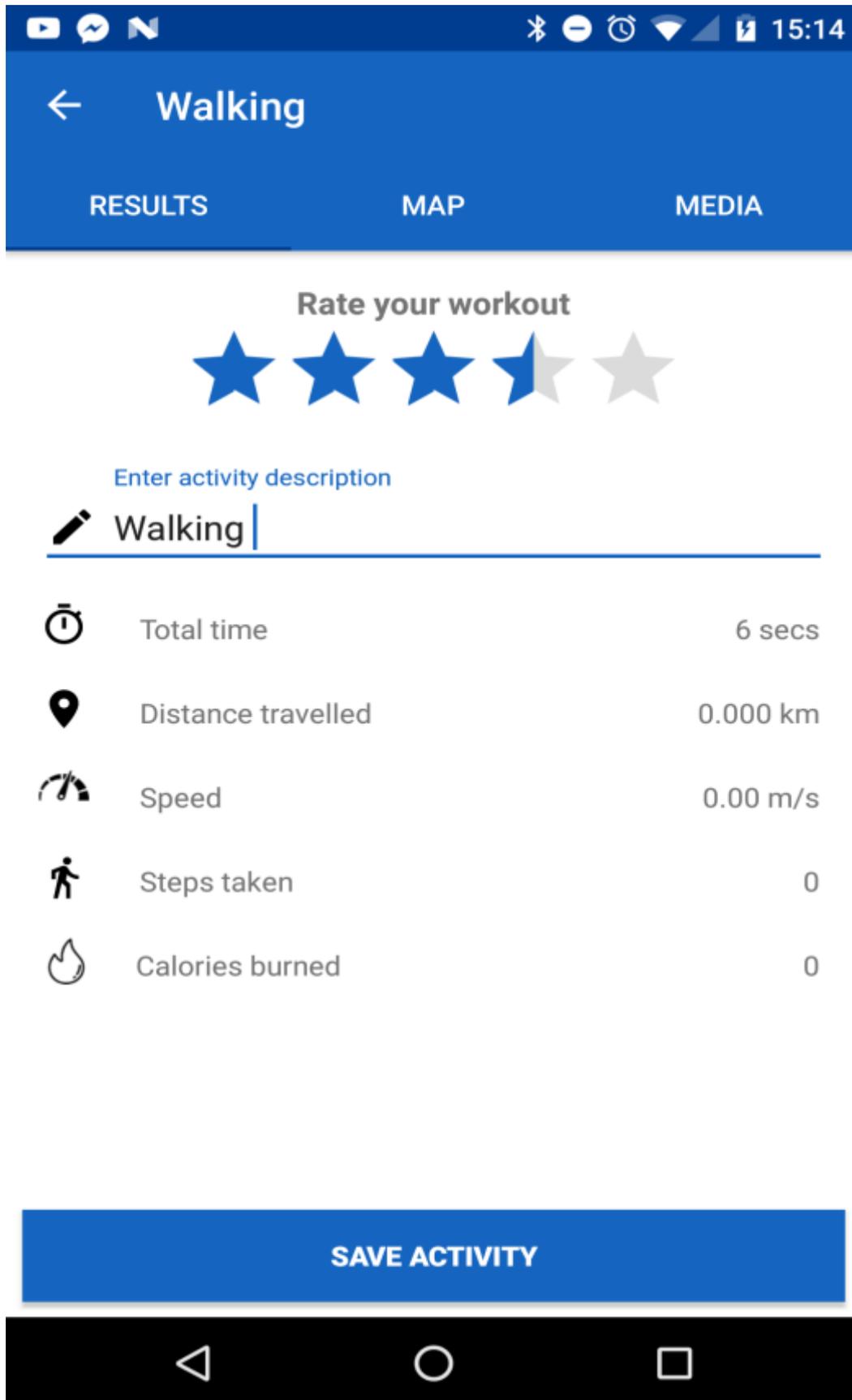


Figure 18 – Old map screen

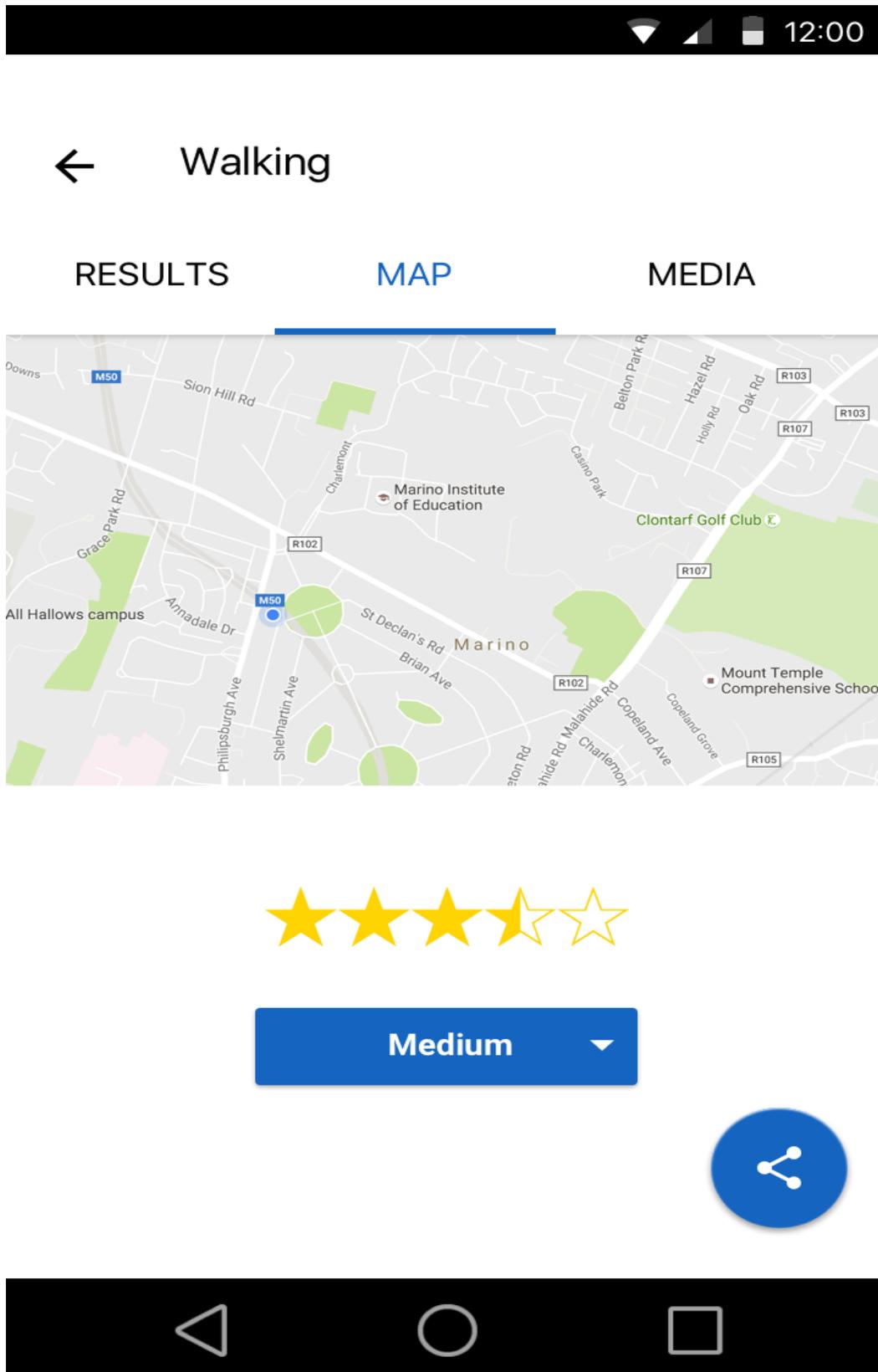


Figure 19 – New map screen

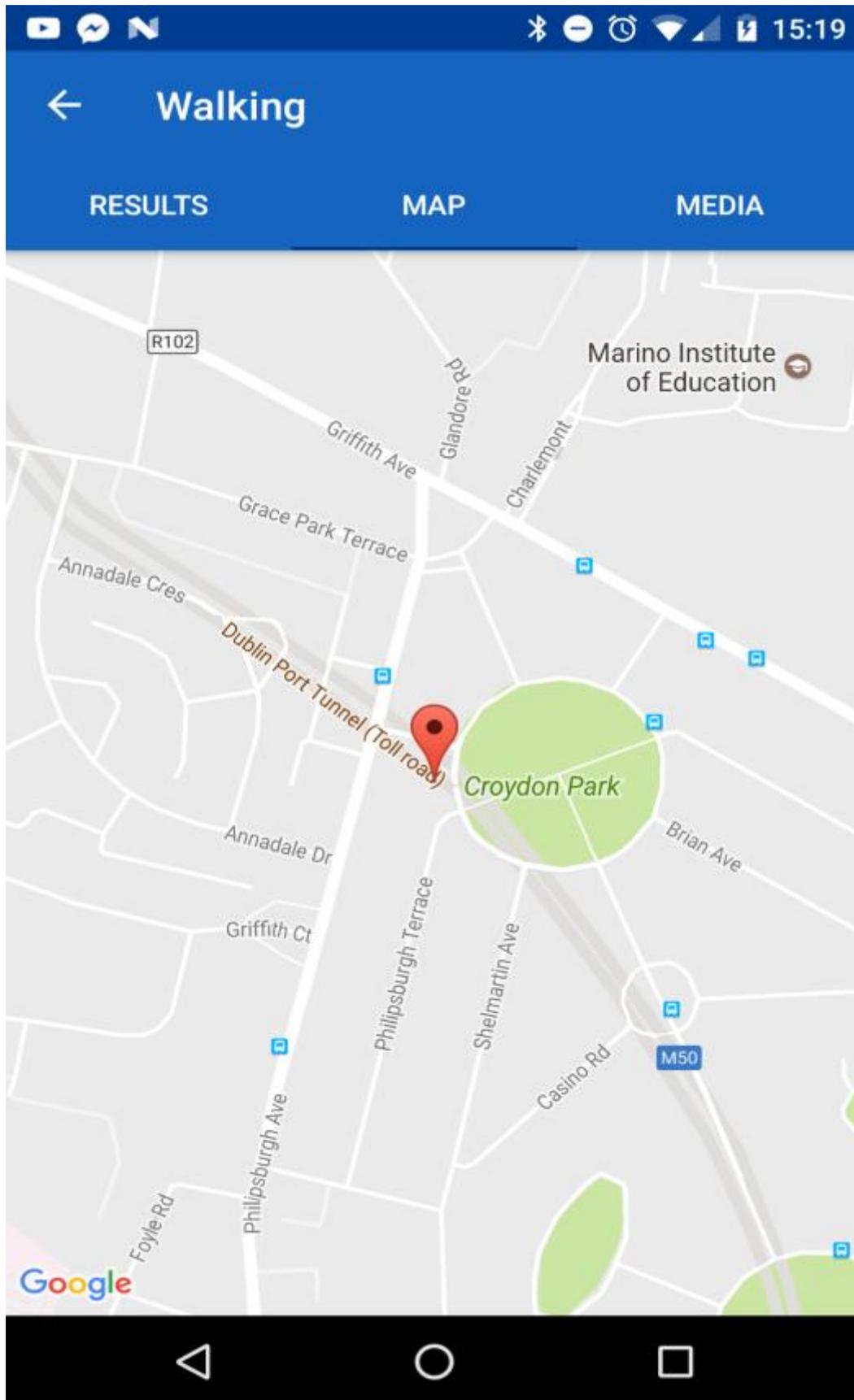
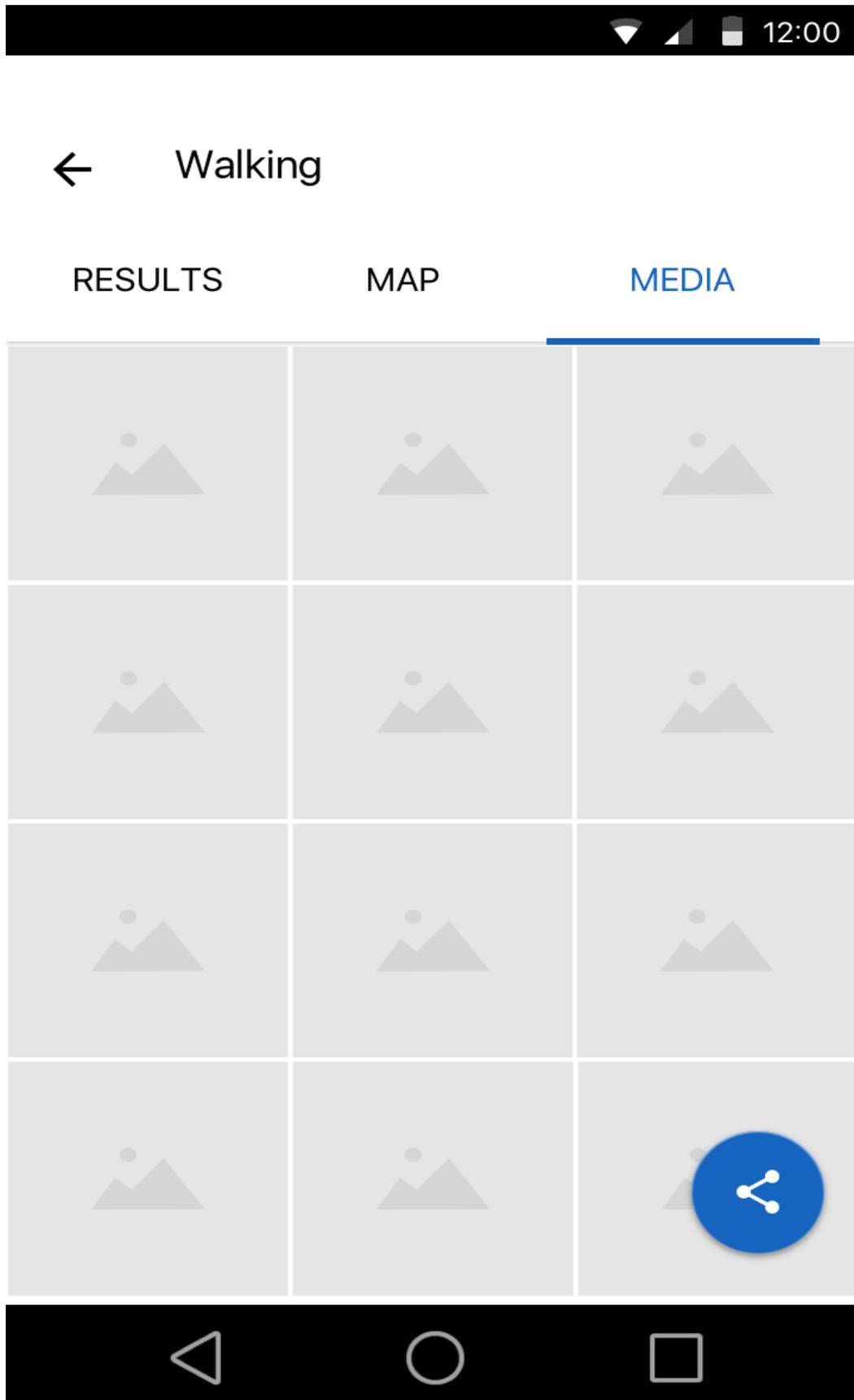


Figure 20 – Media screen



The above screens show the result of their exercise by displaying statistics that will allow users to analyse their performance and rate their experience. They also can view their photos taken during the exercise and share them via their preferred social network.

In figure 16 & 17, there was significant changes to the layout of the results screen. There was decision to place the rating bar and activity description in the results fragment. It allowed for more consistency as the user will be saving the data.

The map layout in figure 18 looked a bit untidy. Therefore, it was decided to place the map inside its own standalone fragment as shown in figure 19.

The layout in figure 20 was achieved using a custom square layout utility class by extending the relative layout class as shown below. It was achieved by overriding the onMeasure method to pass in the widthMeasureSpec twice on the super constructor.

Code Snippet – SquareLayout class

```
public class SquareLayout extends RelativeLayout {

    public SquareLayout(Context context) {
        super(context);
    }

    public SquareLayout(Context context, AttributeSet attrs) {
        super(context, attrs);
    }

    public SquareLayout(Context context, AttributeSet attrs, int defStyleAttr) {
        super(context, attrs, defStyleAttr);
    }

    @RequiresApi(api = Build.VERSION_CODES.LOLLIPOP)
    public SquareLayout(Context context, AttributeSet attrs, int defStyleAttr, int defStyleRes) {
        super(context, attrs, defStyleAttr, defStyleRes);
    }

    @Override
    protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {
        super.onMeasure(widthMeasureSpec, widthMeasureSpec);
    }
}
```

6.7 My activity screen

Figure 21 – Old my activity design

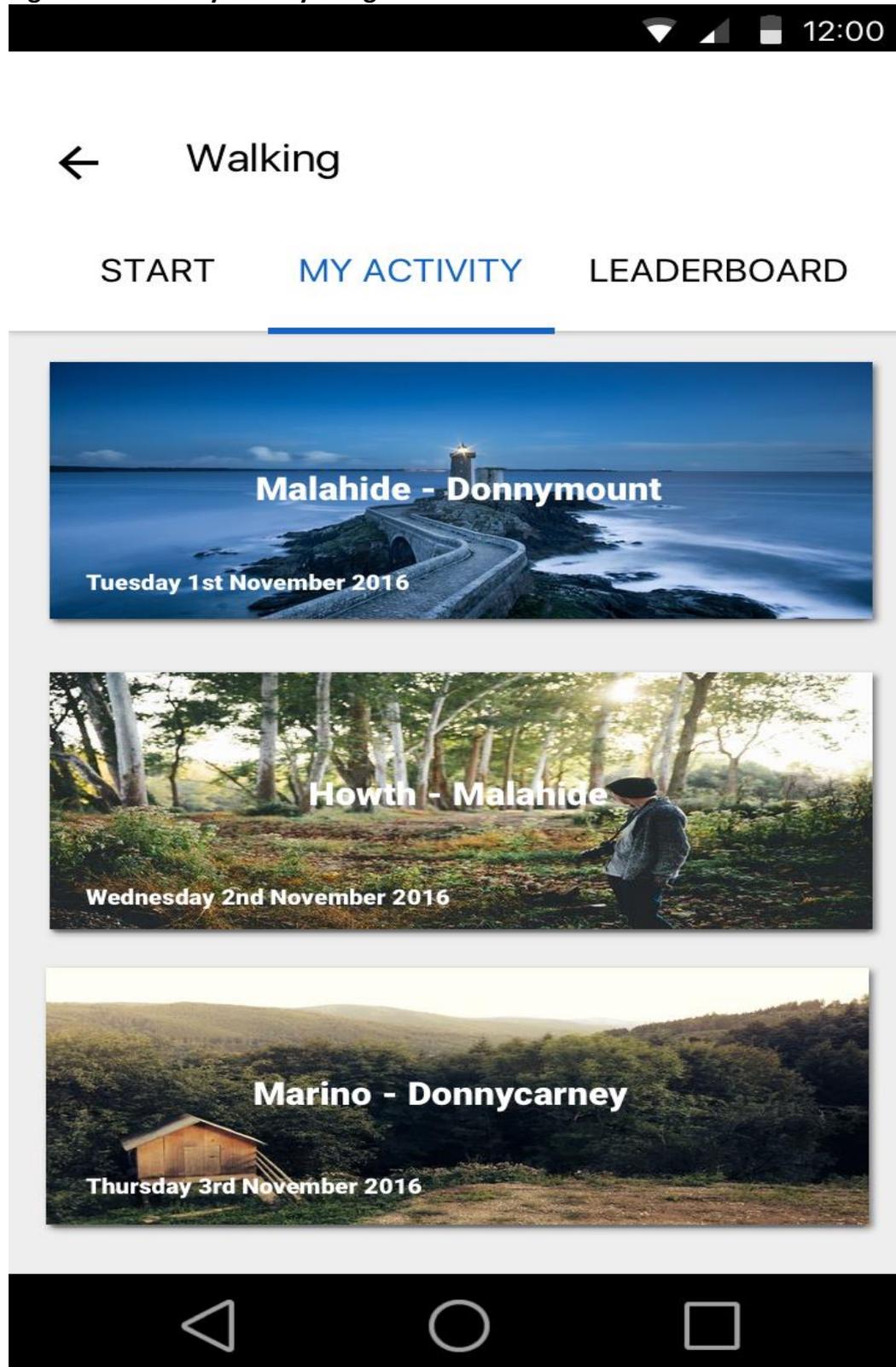
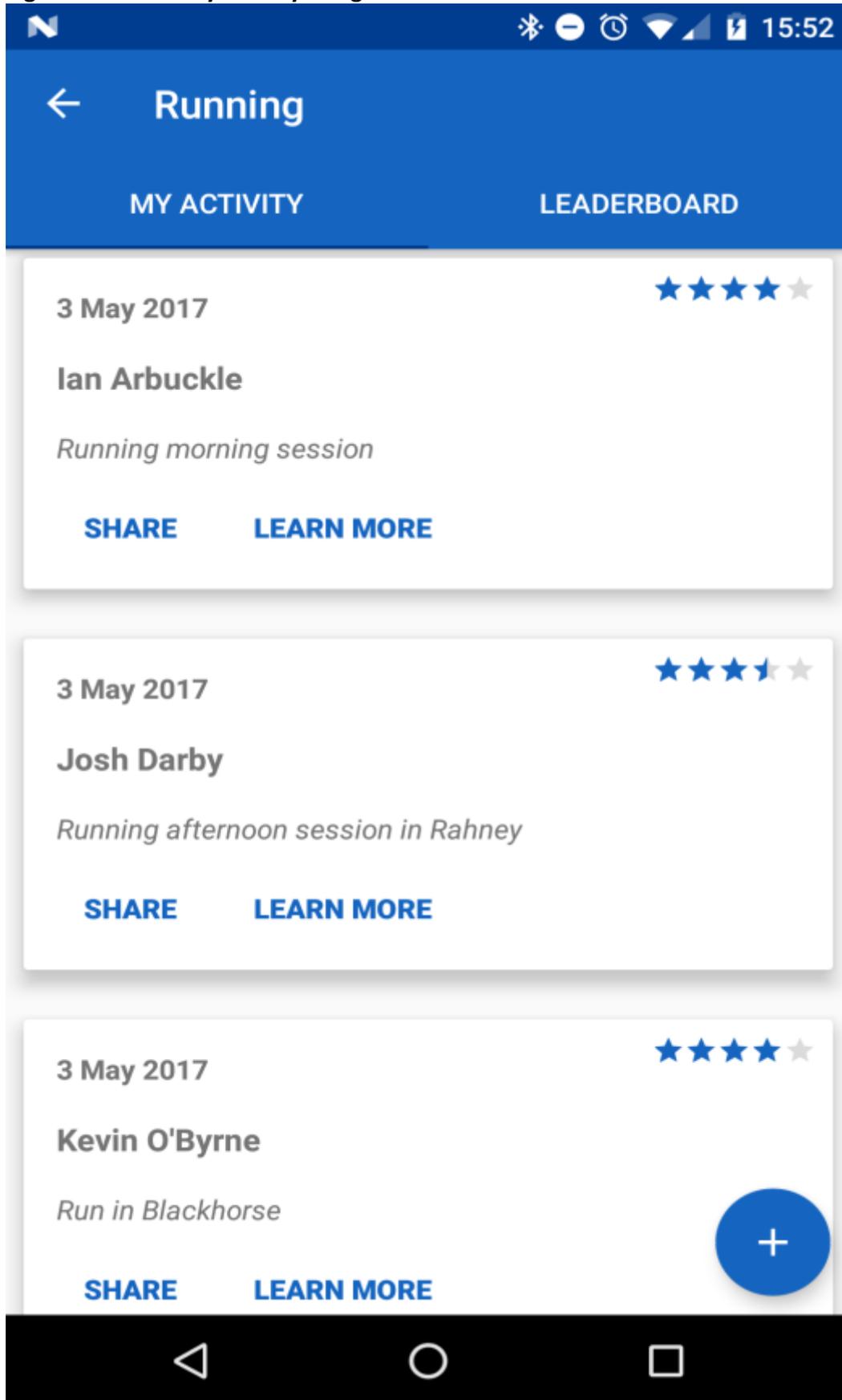


Figure 21 – New my activity design



User can view their past or present exercises by tapping on the card and are presented with the results which was shown in the previous designs.

Figure 21 and figure 22 were achieved by using card layouts with the Android recycler view component. Each card will contain data retrieved from its appropriate node in the database as shown in figure 23 below.

Figure 22 – Firebase database



It was decided to make modifications to the card layout as the images looked stretched from its original size. Therefore, in figure 14, the solution was to introduce a simple and elegant card displaying details about the session. It has call to action for sharing with friends or view more detailed information by tapping "Learn More".

6.8 Leaderboard

Figure 24 – Old leaderboard design

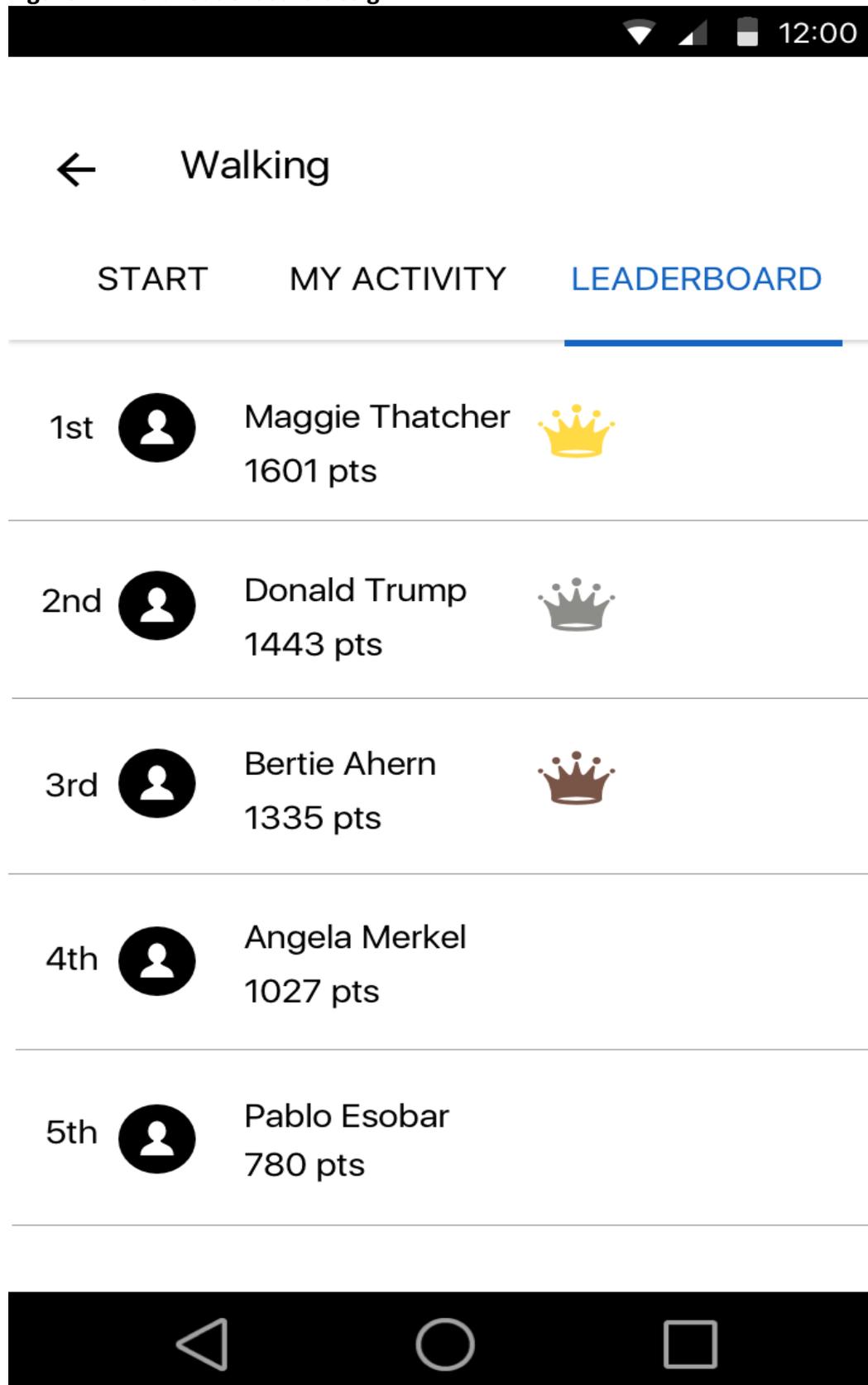
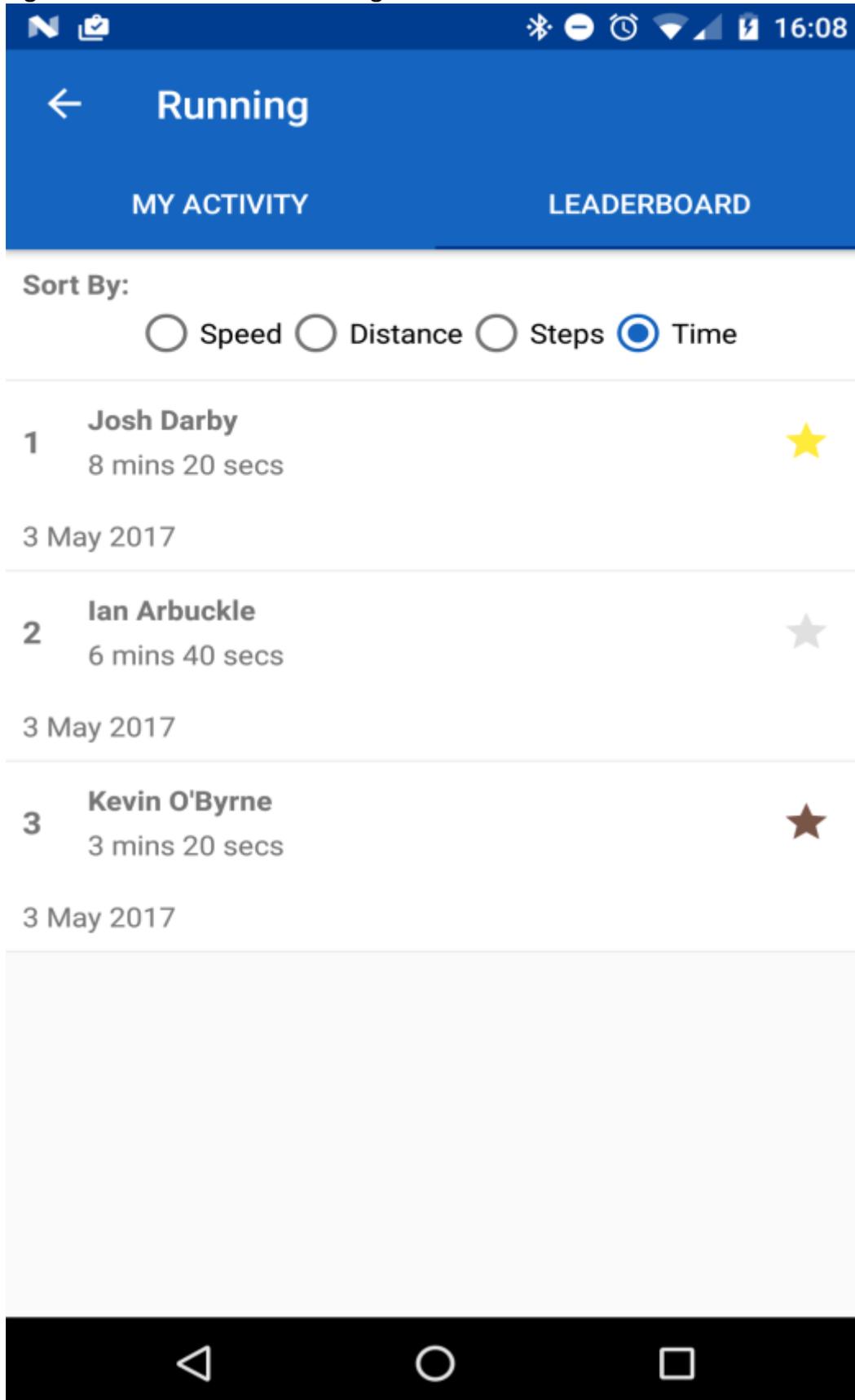


Figure 25 – New leaderboard design



A leaderboard will be displayed as a competition element to encourage users to improve their exercises and learn from other users. It will present the users who has the most points collected and can be filtered by your local area or your list of friends.

The ranking algorithm was achieved by using the Firebase Query object and ordering the data by the specified value. For instance, to sort the table by time, you can specify a query to sort the firebase data by the value of the time field as shown in figure 26.

Figure 26 – Leaderboard ranking using Firebase Query operator

```
@Override
public void setTimeQuery(int layout) {
    Query query = childRef.orderByChild(Constants.FIREBASE_DATABASE_CHILD_TIME);
    adapter = new RunLeadersTimeAdapter(RunWalkModel.class, layout, RunLeadersViewHolder.class, query,
    view.getContext());
    view.setAdapter(adapter);
}
```

However, there are alternative ranking algorithms offered by Google Play Game Services. In the experience of using the Google leaderboard, they do not offer easy customisable user interfaces. As we already implemented our own user interface, it would take a certain amount of work to integrate Google's leaderboard platform. As a result, this didn't suit our requirements for our custom user interface.

In contrast, Firebase introduced their Query operators and this allowed us to simply query the dataset without modifying our user interface. Therefore, it achieved our expected requirements.

7. Testing

This section will describe and evaluate the testing methods employed in testing the application. It was decided to perform manual and automation testing to ensure that the app was ready for production. The benefits of testing provide the confidence that our app executes the expected requirements.

7.1 Manual Testing

This involves testing the features of the app without using automation tools to determine the robustness and performance.

It was decided to write test cases for user stories and acceptance criteria. However, sometimes we assume that acceptance criteria are to identify bugs while it's the opposite. Instead, it allows us to test if the business requirements are met.

Below are executed test cases for user stories:

7.1.1 Test Cases

Project Name: FitNow

Test Case ID: FN-114

Test Designed by: Ian Arbuckle

Test Priority: High

Test Designed date: 27/2/2017

Module Name: Recording workout session

Test Executed by: Ian Arbuckle

Test Title: Verification of recording workout session

Description: Test the recording workout session

Pre-conditions: User retrieves sensor data that was recorded

Steps	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)
1	Choose activity from home screen. Start activity by tapping fab (float action button)	N/A	User is displayed activity start screen	User was displayed start screen with call to action	Pass
2	Starting activity session	Countdown time	User is shown countdown time and activity dashboard is displayed	User shown the activity dashboard after successful countdown	Pass
3	Recording sensor data	Speed, calories, distance and steps	User should retrieve sensor data while executing activity	User was displayed the number of steps while walking	Pass
4	Pause recording session	Time	User should be shown that time is paused	User is shown the time is paused	Pass
5	Stop Session	N/A	User should be brought to result dashboard	User is brought to result dashboard	Pass

Post conditions:

User's session is active and sensor data is being collected. The user is also able to use the controls to pause or stop the session.

Project Name: FitNow

Test Case ID: FN-112

Test Designed by: Ian Arbuckle

Test Priority: High

Test Designed date: 2/5/2017

Module Name: Leaderboard

Test Executed by: Ian Arbuckle

Test Title: Verification of leaderboard statistics

Description: Test the accuracy of data provided by the leaderboard

Pre-conditions: User views the statistics displayed on leaderboard

Steps	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass/Fail)
1	Choose activity from home screen. Swipe right or tap the "Leaderboard" tab	N/A	User is displayed leaderboard screen	User was displayed leaderboard screen	Pass
2	User is displayed radio buttons for filtering leaderboard	N/A	User is shown filter options for leaderboard	User was displayed appropriate filter options	Pass
3	User filters leaderboard for appropriate data	Time, Steps, Distance and Speed	User is shown appropriate data for filtered data	User was displayed appropriate data for filtered data	Pass

Post conditions:

User's session is active and sensor data is being collected. The user is also able to use the controls to pause or stop the session.

Although, manual testing has not been overall successful as there was not enough sufficient time to execute manual test cases. Due to time constraints of the project deadlines it was not possible to execute stress and performance testing to determine the robustness of the app.

If sufficient time was allocated to execute these tests, we would be able to determine if the app was ready for live production to the app store.

7.2 Automation testing

This section will evaluate and describe the automation testing methods employed in testing the application.

7.2.1 Unit testing

FitNow has utilised the MVP design pattern which enables us to test the business logic using POJOs (plain old java classes) in the mockito library for Android.

Figure 27 - Code snippets of unit test cases below

```
@RunWith(MockitoJUnitRunner.class)
public class RegisterPresenterImplTest {

    private RegisterPresenterImpl presenter;

    @Mock
    AuthenticationHelper authenticationHelper;

    @Mock
    RegisterView view;

    @Before
    public void setup() throws Exception {
        MockitoAnnotations.initMocks(this);
        presenter = new RegisterPresenterImpl(authenticationHelper);
        presenter.setView(view);
    }

    @Test
    public void testRegisterValidUser() throws Exception {
        presenter.registerAccount("email", "test");
        verify(authenticationHelper).logOutUser();
        verify(authenticationHelper).registerUser(anyString(), anyString(), any(RequestListener.class));
    }

    @Test
    public void testAttemptToRegisterIfUserEmailIsEmpty() throws Exception {
        presenter.registerAccount(" ", "test");
        verify(view).hideProgress();
        verify(view).showInvalidEmailMessage();
    }

    @Test
    public void testIfPasswordsMatchAreEmpty() throws Exception {
        presenter.validatePassword(" ", " ");
        verify(view).hideProgress();
        verify(view).showPasswordEmptyMessage();
    }

    @Test
    public void testIfPasswordsMatchAreDifferent() throws Exception {
        presenter.validatePassword("bacon", "tuna");
        verify(view).showErrorMessage();
    }
}
```

```
verify(view).hideProgress();  
}
```

```
@Test  
public void testIfPasswordsMatch() throws Exception {  
    presenter.validatePassword("bacon", "bacon");  
    verify(view).showProgress();  
    verify(view).registerOnPasswordMatch();  
}
```

The above unit test, figure 26, uses the Mockito open source library framework to mock certain objects. It utilises the use of annotations to remove boiler plate code and this test uses the following:

@RunWith – This initialises mocks annotated with **@Mock**

@Mock – This marks the field as a mock and reduces repetitive code

@Before – This tells the method to run before the **@Test** method so that objects are initialised

@Test – This tells JUnit that the method can be run as a test case

Registration test cases:

- Test if there is a valid registration
 - Mockito mocks the presenter object
 - Executes a call to the presenter method “registerUser”
 - Verifies if authentication helper class can log out the user
 - Verifies if authentication helper class can register a mock user
- Test for attempted registration without empty email string
 - Mockito mocks the presenter object
 - Executes a call to the presenter method “registerAccount”
 - Verifies if hide progress and error dialog method is called
- Test if passwords match are empty
 - Mockito mocks the presenter object
 - Executes a call to the presenter method “validatePassword”
 - Passes in two mock empty strings into presenter method
 - Verifies if hide progress and error dialog method is called

7.3 Continuous Integration

Continuous integration “is a development practice that requires developers to integrate code into a shared repository several times a day (Thoughtworks.com, 2017).” This emphasises the importance of using CI which allows us to detect any potential issues that we may miss.

We setup the CI environment with Travis which is built into GitHub. Firstly, we needed to write a Travis .yml script with certain rulesets and environments. Then directly import it into our Android app root directory so that Travis can trigger a build.

Figure 28 - Code snippet – Travis Script

```
language: android
jdk:
  - oraclejdk7
android:
  components:
    - tools
    - build-tools-23.0.3
    - android-25
    - platform-tools
    - extra-google-google_play_services
    - extra-google-m2repository
    - extra-android-m2repository
    - addon-google_apis-google-19

notifications:
  email:
    - ian.arbuckle18@gmail.com

before_install:
  - pip install --user codecov

before_script:
  - chmod +x gradlew

script:
  - chmod +x ./gradlew

after_success:
  - codecov
```

The above .yml script, figure 28, is broken down into the following:

- Language

This describes the environment you want the CI build to execute in. It supports many language environments such as Java, C, C++, Python, Ruby and so on.

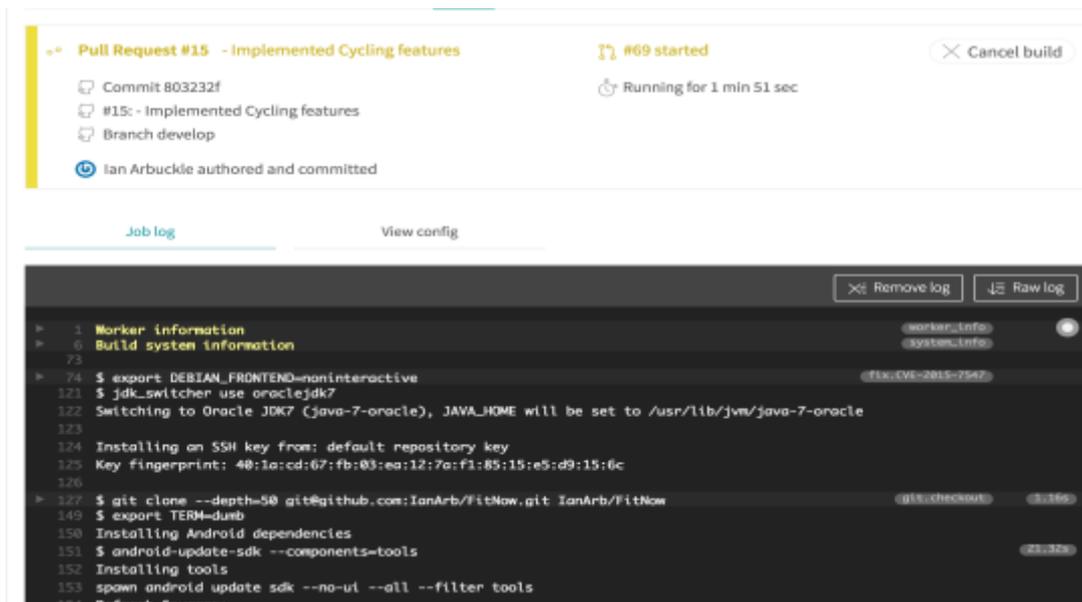
- Components

These describe the Android components you which to test on the CI host environment

- Before_install, before_script, script and after_success

These elements describe the conditions when a CI build is triggered. For instance, I needed to grant permissions for the build to read scripts that it does not have access to.

Figure 29 – CI build being triggered



Every time a push or pull request is made, a CI build is automatically triggered. Figure 29 above shows a CI build in process for a pull request to develop branch. Once the build has passed, you can merge the changes into the required branch.

From experience in a Jenkins environment it was difficult to integrate a GitHub repository with Jenkins. The reason is because there are more configurations required to setup a Jenkins CI environment. However, if the configurations and environment are integrated correctly, it can be a very powerful CI tool in contrast to Travis.

Although, it was decided to try Travis as the CI hosting environment as it is automatically generated for you once you port your GitHub repository. This saved time configuring the environment and allocated more resources into development. However, for future projects, Jenkins would be the most preferred CI tool as it provides a large range of plugin tools for the Android environment.

7.4 Continuous Code Quality

Continuous code quality allows us to gain confidence in the quality of our code coverage. We used tools to analyse our code coverage to spot potential bad coding practices that could lead to potential bugs or defects.

We took advantage of using an open source Java source code analyser, PMD. In our Android application, we used a PMD tool called findbugs which analyses our code source. We activated this tool by running a custom gradle script and then simply executing a command in the Unix terminal.

PMD can detect the following flaws:

- Possible bugs

An example of possible bugs could be where you compare strings with == operator while you should always use equals() method.

- Dead code

This can be unused code such as unused variable name or imports that are not required anymore.

- Overcomplicated expressions

For instance, you may have an if statement with several conditions that can be easily refactored into a class or a switch case.

- Suboptimal code

This consists of wasteful String/Butter usage.

- Classes with high measurements

This would be a case of measuring the complexity of your code by using Big O notations and PMD will recommend reducing the complexity.

- Duplicate code

PMD can detect if code was copy/pasted as it reduces the maintainability of the code.

Figure 30 - Code snippet - Gradle script for pmd findbugs

```
task findbugs(type: FindBugs) {
    group = "analysis"
    ignoreFailures = false
    effort = "max"
    reportLevel = "high"
    excludeFilter = new File("${project.rootDir}/config/findbugs/findbugs-exclude-filter.xml")
    includeFilter = new File("${project.rootDir}/config/findbugs/findbugs-include-filter.xml")
    classes = files("$project.buildDir/intermediates/classes")

    source 'src'
    include '**/*.java'
    exclude '**/gen/**'

    reports {
        xml.enabled = false
        html.enabled = true
        xml {
            destination "$project.buildDir/reports/findbugs/findbugs.xml"
        }
        html {
            destination "$project.buildDir/analysis/reports/findbugs.html"
        }
    }

    classpath = files()
}
```

- The gradle script figure 29 above, is written in Groovy
- Each task is defined and searches through the project directory for the xml files to execute

Figure 31 – Terminal Output for PMD

```
Total time: 15.807 secs
Ian-Arbuckle:FitNow ianarbuckle$ ./gradlew pmd findbugs
The Task.leftShift(Closure) method has been deprecated and is scheduled to be removed in Gradle 5.0. Please use Task.doLast(Action) instead.
    at quality_63qve6wn2u4o5q8qd6lrvca7r.run(/Users/ianarbuckle/AndroidStudioProjects/FitNow/config/quality.gradle:16)
:app:pmd FAILED

FAILURE: Build failed with an exception.

* What went wrong:
Execution failed for task ':app:pmd'.
> 6 PMD rule violations were found. See the report at: file:///Users/ianarbuckle/AndroidStudioProjects/FitNow/app/build/analysis/reports/pmd.html

* Try:
Run with --stacktrace option to get the stack trace. Run with --info or --debug option to get more log output.

BUILD FAILED

Total time: 8.474 secs
Ian-Arbuckle:FitNow ianarbuckle$
```

Activate the tool by using the built in unix terminal in the IDE. As a result, it prints a HTML document for you to review potential bugs.

- Run the following command in terminal to execute the plugin
 - `./gradlew pmd findbugs`
- While the process is running it will generate the pmd file and findbugs script defined in the xml
- It will generate an HTML file output to review potential bugs found in the source code

Automation testing was a big factor in ensuring that our code coverage has met high coding standards and practices. It successfully allowed us to ensure that our source code was doing what it's expected. In contrast to manual testing, it was significantly faster and this was assisted by third party tools such as Mockito, Travis CI and Sonar.

8. Conclusion

It can be concluded that it was an enjoyable and challenging project. There were significant challenges in optimising the accuracy of the sensors on a mobile device. Unfortunately, there was not much that could be achieved in optimising the accuracy of the live tracking sensors. To overcome these challenges, it was decided to focus on manipulating the dataset into the leaderboard and dashboard.

If the budget was provided, it would be interesting to obtain an Android Wear device to compare the built-in sensors with a mobile device. It also would be worth investigating other alternatives for fitness tracking to overcome the accuracy of the live sensors.

The follow pros and cons of the app for the users as follows:

Pros

- User can personalise their fitness using photos
- A competitive environment using a leaderboard gamification feature
- Displaying statistical data to allow users to evaluate and improve their fitness

Cons

- Engaging users to keep using the application in the long-term
- Some users may not wish to use social media to share their experience (privacy concerns)
- Competition in the fitness market may prove be difficult to get the app exposure in the market

Finally, the project has improved my software development skills in the mobile development environment. It has also improved my awareness of the importance of continuous delivery such as including new features, configurations and bug fixes.

9. Further Development or research

App

- The application can be integrated into Android and Apple wear devices to allow users to easily track their exercises.
- The application can be extended to iPhone devices using the same backend infrastructure, thanks to Firebase's portability.
- The application can present an opportunity to generate money using paid features and with the help of the push notification and analytics engine from Firebase infrastructure.

Sponsorship and marketing

- It would worth exploring the possibilities to gain sponsorships from firms that specialise in health and offer packages as part of a premium feature in the app

Use Firebase analytics tool to target ads on the app to generate revenue

10. Bibliography

Szczerba, R.J. (2014) ***Six fitness problems Fitbit won't solve for you.***

Available at: <http://www.forbes.com/sites/robertszczerba/2014/01/15/six-fitness-problems-fitbit-wont-solve-for-you/#3fc668d43205> (Accessed: 2 December 2016).

Exercise motivation: How to get it, how to keep it (2013)

Available at: <http://www.webmd.com/fitness-exercise/features/exercise-motivation#1> (Accessed: 2 December 2016).

Accessibility - usability - material design guidelines (no date)

Available at: <https://material.google.com/usability/accessibility.html#accessibility-principles> (Accessed: 2 December 2016)

Thoughtworks.com. (2017). Continuous integration | ThoughtWorks. [online]

Available at: <https://www.thoughtworks.com/continuous-integration>

[Accessed 25 Apr. 2017].

Cloudspm.com. (2017). CloudSPM. [online]

Available at: http://www.cloudspm.com/applications_gamification.html [Accessed 5 May 2017].

CITE A WEBSITE - CITE THIS FOR ME

Vironit.com. (2017). Cite a Website - Cite This For Me. [online]

Available at: <http://vironit.com/what-mobile-fitness-apps-do-users-really-need/>

[Accessed 6 May 2017].

Appendix

Project Proposal

1. Objectives

The objective of the Android app is to allow the user to track and improve their fitness by using gamification methods. Gamification is a concept of applying game mechanics and techniques to engage and motivate users to achieve their goals. It allows us to tap into the basic desires and needs of the end user which resolves around the idea of status and achievement.

The mobile application will integrate a leaderboard to allow users to compete in a competitive environment. The user will have the option to share their statistics with friends via social media (Facebook and/or Twitter) to prompt for a challenge to beat their record.

FitNow will allow the user to choose their own specific category of exercise combinations:

- Walking
- Running
- Cycling

FitNow will consist of the following features:

- FitNow will use a leaderboard as a gamification technique to create a competitive environment
- Users will have the opportunity to learn from others by viewing their competitor's statistical data
- The user will be able to view their fitness progress such as distance travelled, calories burned, active time to name but just a few.
- Social media integration to allow users to share or invite friends to challenge them

Socialise their own fitness by adding photos of their surroundings and rating their experience by level of difficulty.

I believe the scope of the objectives outlined is flexible and more features can be added. The aim is implement the gamification element to motivate users with social and fitness features.

2. Background

The background of FitNow is to integrate a social and gamification element to improve our fitness. We all have issues finding ways to motivate ourselves to keep a healthy lifestyle and the problem is that we don't use technology effectively to solve this problem.

To solve the problem, we can achieve this by using gamification techniques that will help us to engage and motivate users through the idea of status and achievement.

Our friends use social media to connect and share content. The use of social media can be a powerful tool of motivation which is evident through status updates of our friends sharing their fitness milestones. Social media elements will allow FitNow to share fitness content and connect in an engaging activity.

3. Technical Approach

Research

- Research online databases that will allow me to implement the real-time competition element of the app (Live network database)
- Explore technologies that will help me track the user's fitness and health
- Further my knowledge of the Android framework through book and online resources
- Investigate and explore statistical dashboards to display fitness metrics

Literature Review

- Pick up a book or course on MVP (Model View Presenter) to increase my existing knowledge of the architecture design pattern
 - MVP is a derivation of the MVC (Model View Controller) architecture pattern and is popular for building user interfaces.
 - In MVP, the presenter takes the responsibility of the "middle man" and all business logic is pushed to the presenter.
- Use online resources such as Android documentation, YouTube tutorials and pluralsight.

Implementation

- Real time network database for storing information online and offline
- Tracking health and fitness of the user
- Social media integration

Dashboard for fitness and health metric4

4. Special Resources Required

Hardware

- Android device – Minimum SDK, Kitkat
- Android wear device (future implementation)
- MAC OSX

Software

- Android Studio IDE

5. Project Plan

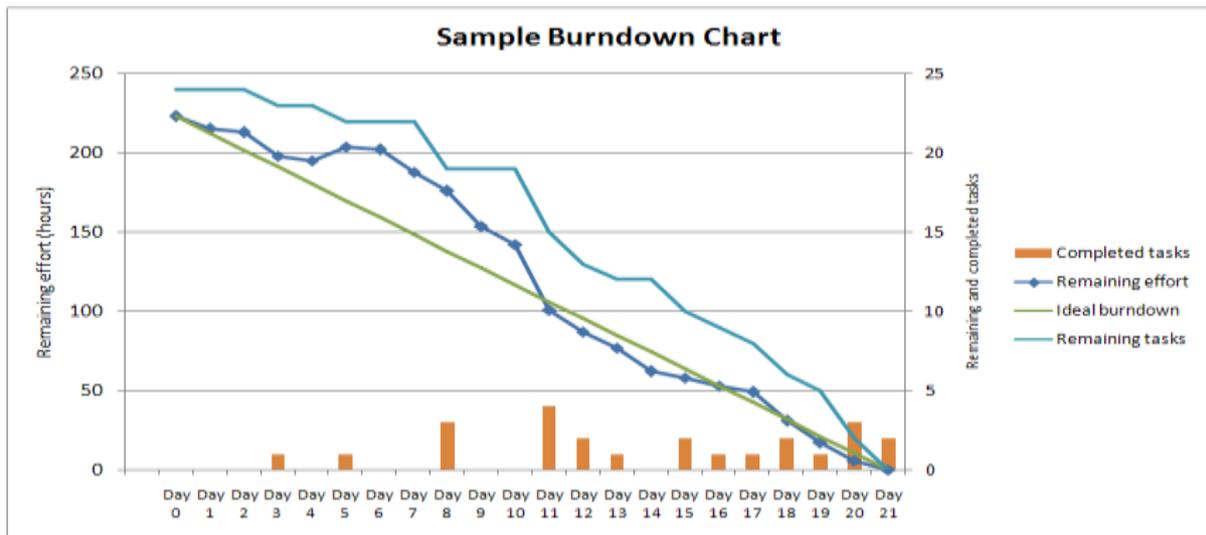
The project plan will be implementing agile methodology methods. Agile is an iterative approach to software delivery that builds software incrementally from the beginning of the project, instead of trying to deliver it all at once at the end with the waterfall model approach.

Agile features

- Backlog (Stories/Features)
 - In Agile, a backlog is a list of items that needs to be completed
- Sprints
 - Each sprint contains shippable increments that will be completed
- Grooming Session
 - It is keeping the backlog is kept clean and orderly and ensuring that the backlog is ready for the next sprint

To track each sprint, I will create burn down charts to determine whether the sprint is being successful. A burn down chart is graphical representation of work left to do versus time. The outstanding backlog is often on the vertical axis, with time along the horizontal. That is, it is a run chart of outstanding work. It is useful for predicting when all the work will be completed.

Example of a burn down chart:



I will be using Trello which is a web app project management tool that allows me to visually organise my project. I will provide access to my supervisor to view the Trello board.

Website - <https://trello.com/>

6. Technical Details

Languages

- Java
 - It is the recommended language used to develop native Android applications. I've used Java since my first year in college and during my work placement in MTT (Mobile Travel Technologies).
- XML
 - It is used to organise UI widgets and layouts for the Android user interface
- JSON
 - This will be used to parse information from the backend and I've have experience using JSON at my work placement

Libraries

- 1. Android SDK**
 - 1.1.** The package library for the Android platform
- 2. Gradle**
 - 2.1.** It is the build system to automate tasks in Android
- 3. Firebase**
 - 3.1.** A cloud backend service owned by Google
- 4. Google Fit, Charts and Maps APIs**
 - 4.1.** APIs available to Android developers to integrate into their apps

8. Evaluation

To evaluate the Android application with an end user the following methods will be implemented:

End user testing

1. Manual Testing
 - a. Test the features of the app without using automation tools
 - b) User Acceptance Testing
 - a. At the end of sprint completion there will be a user acceptance testing for testing the features of the application

Automation testing

1. Unit testing
 - b. Test the business logic in plain old Java classes using Android mockito library
 - c) Instrumentation tests
 - a. Use Android espresso test library to test UI components
 - d) Continuous Integration

Use continuous integration tools such as Jenkins or Travis to automate test

Project Plan (Requirements Specifications)

1. Introduction

1. Purpose

The purpose of this document is to set out the requirements for the development of FitNow Android application.

The intended customers are those who wish to improve their fitness and overall health by using gamification methods.

2. Project Scope

The scope of the project is to develop a fitness mobile application using gamification techniques on the Android platform. Gamification is a concept of applying game mechanics and techniques to engage and motivate users to achieve their goals. It allows us to tap into the basic desires and needs of the end user which resolves around the idea of status and achievement.

FitNow System features:

FitNow will allow the user to choose their own specific category of exercise combinations:

- Walking
- Running
- Cycling

FitNow will consist of the following features:

- FitNow will use a leaderboard as a gamification technique to create a competitive and healthy environment
- Users will have the opportunity to learn from others by viewing their competitor's statistical data
- The user will be able to view their fitness progress such as distance travelled, calories burned, active time to name but just a few.
- Social media integration to allow users to share or invite friends to challenge them
- Socialise their own fitness by adding photos of their surroundings and rating their experience by level of difficult

2. User Requirements Definition

The user requirements of FitNow is to integrate a social and gamification element to improve our fitness. We all have issues finding ways to motivate ourselves to keep a healthy lifestyle and the problem is that we don't use technology effectively to solve this problem. In order to solve the problem, we can achieve this by using gamification techniques that will help us to engage and motivate users through the idea of status and achievement.

Our friends use social media to connect and share content. The use of social media can be a powerful tool of motivation which is evident through status updates of our friends sharing their fitness milestones. Social media elements will allow FitNow to share fitness content and connect in an engaging activity.

Clients expectations of a fitness application:

- Ability to motivate themselves by challenging and learning from other end users
- Socialise their fitness experience by sharing with friends of photos, statistics and map of distance path travelled
- Learning from users on what equipment, level of difficulty, and distance travelled to improve their own fitness by viewing their statistics via the leaderboard
- Invite friends to use the app to encourage competition

3. Requirements Specification

FitNow will be first implemented as an Android application for the smartphone and wear devices for a future implementation. The app will aim to satisfy experienced Android users through a user-friendly interface. Although there will be users who are not experienced Android users and the application will cater via a helpful guide on how to use the application.

1. Functional requirements

The functional requirements of FitNow will be ranked by order of highest ranking.

1. User registration via Social Network (Facebook, Twitter, Google+) to collect personal data
2. Selecting specific fitness category – Walking, Running or Cycling
3. Recording exercise activity by using GPS location tracking and detecting health for calories burnt, speed per k/m etc.
4. Viewing dashboard for finished exercise – Fitness health statistics, google map statistics and sharing media for photos
5. Viewing leaderboard to view opponents and send app invites to other users to use the app

Viewing your fitness activity – Show past fitness exercises to share with friends or analyse the dashboard to view progress

Use Cases

Requirement 1 – User Login & Registration

User must provide login credentials in order to collect personal data for tracking location, fitness and ability to share content via social media.

Description & Priority

In order to store information about the user's exercise record we need to collect social media profiles in order to grant ability to share content and send app invites.

Use Case

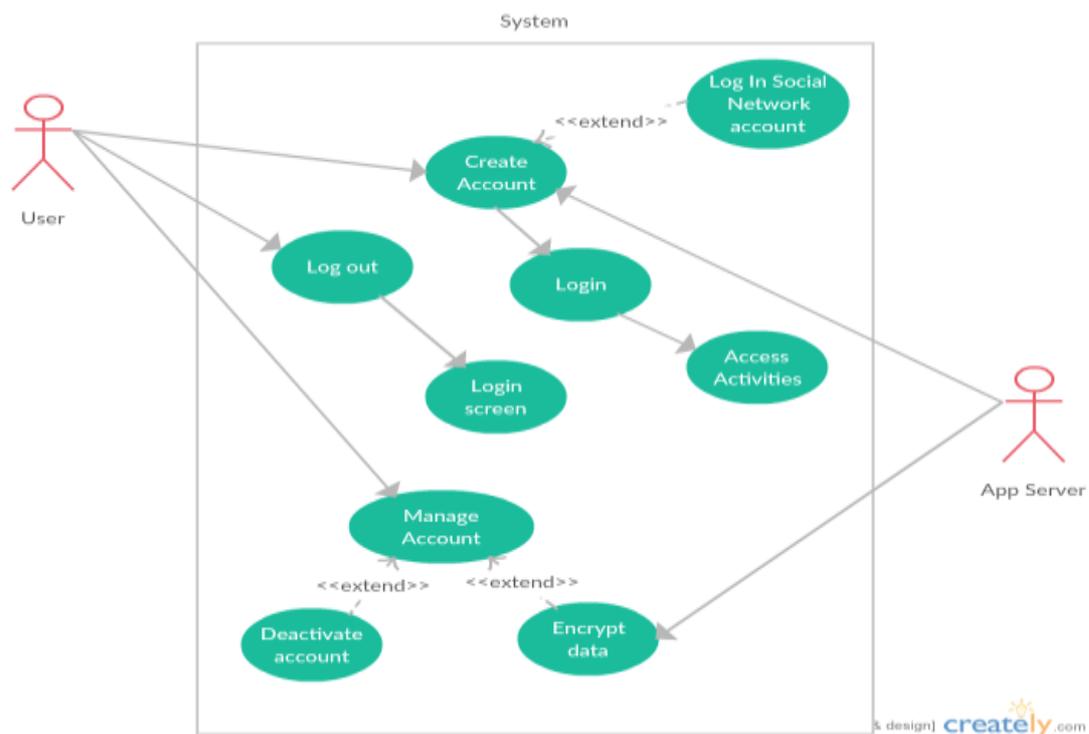
Scope

The scope of this use case is to grant the user access to the application features by logging in with user credentials

Description

This use case describes the registration of an account and managing a user's account.

Use Case Diagram



Flow Description

Precondition

The application is started by the user and is presented with a splash screen

Activation

The use case starts when a user creates an account with their social media credentials

Main Flow

1. App presents user option to create account with social media credentials
2. The user initialises connection with their preferred social platform
3. The app server collects the information to initialise a session with the user
4. The user proceeds to login
5. The user is able to access activities
6. The app presents the user the option to manage account settings
7. The user modifies settings
8. The app encrypts the data provided by the user

Alternate flow

1. The user has an account with the app and doesn't need to register
2. The app identifies the user's credentials and session
3. The user is brought directly to the main flow 5

Alternate flow

1. The user deactivates its account
2. The app presents user with warning dialog that all data will be lost
3. The user dismisses the dialog and is presented to main flow 1

Exceptional flow

1. The user attempts to connect account with social media credentials
2. The app rejects the credentials provided

3. The user is displayed a helpful dialog to either try again or choose a different social platform
4. The app continues at main flow 2

Termination

The app presents the user to the home screen to proceed with its activities

Post condition

The app awaits user interaction

Requirement 2 – Selecting specific exercise category

Description & priority

User must select the specific category the wish to start exercise; walking, running or cycling.

This allows the app to setup the platform for the specific exercise.

Use Case

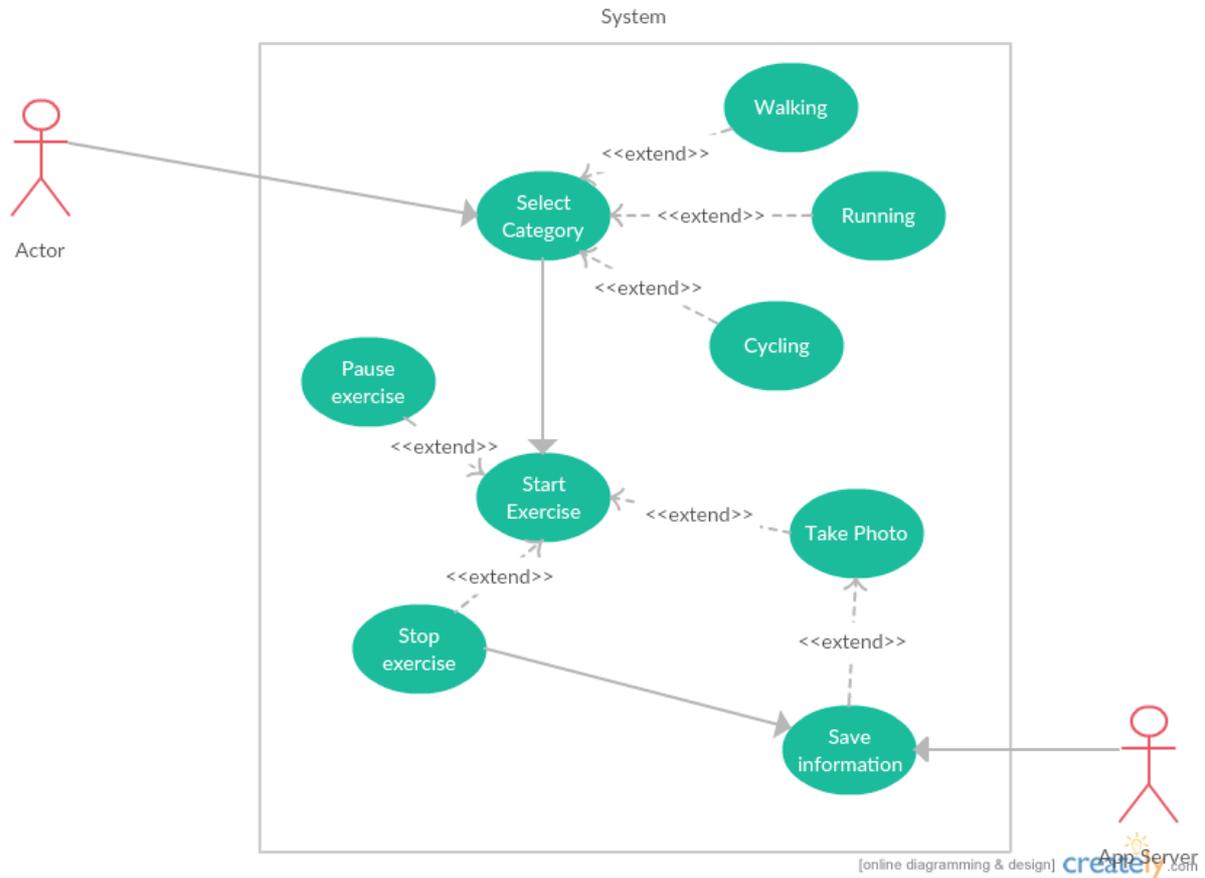
Scope

The scope of this use case is setup the exercise platform based on the user's specific category.

Description

This use case describes the procedure to selecting a category and the user being presented with their platform to start their exercise.

Use Case Diagram



Flow Description

Precondition

The app presents the user to the home screen to select specific category

Activation

The use case starts when the user selects their category

Main flow

1. The user selects their specific category
2. The app initialises the platform for the user
3. The user attempts to start their exercise
4. The app starts the exercise
5. The user attempts to pause the exercise

6. The app pauses the exercise
7. The user takes a photo
8. The app saves the photo
9. The user resumes the exercise
10. The user attempts to stop exercise
11. The app stops the exercise and saves information during the session

Alternative flow

1. The user attempts to take a photo during exercise
2. The app will pause the session while photo is being executed
3. The user will resume their exercise

Exception flow

1. The user answers another feature on the device i.e. phone call
2. The app will pause the exercise in the background
3. The user reopens the app from the background task
4. The exercise is in a pause state
5. The user reactivates the exercise
6. The app goes to main flow 9

Termination

The app presents the user to the next screen

Post condition

The app saves the exercise information to the app server's database

Requirement 3 – Recording the exercise

Description & priority

The user exercise's is being calculated and tracked by the application. The app will determine the distance travelled on a map and the health progress of the exercise.

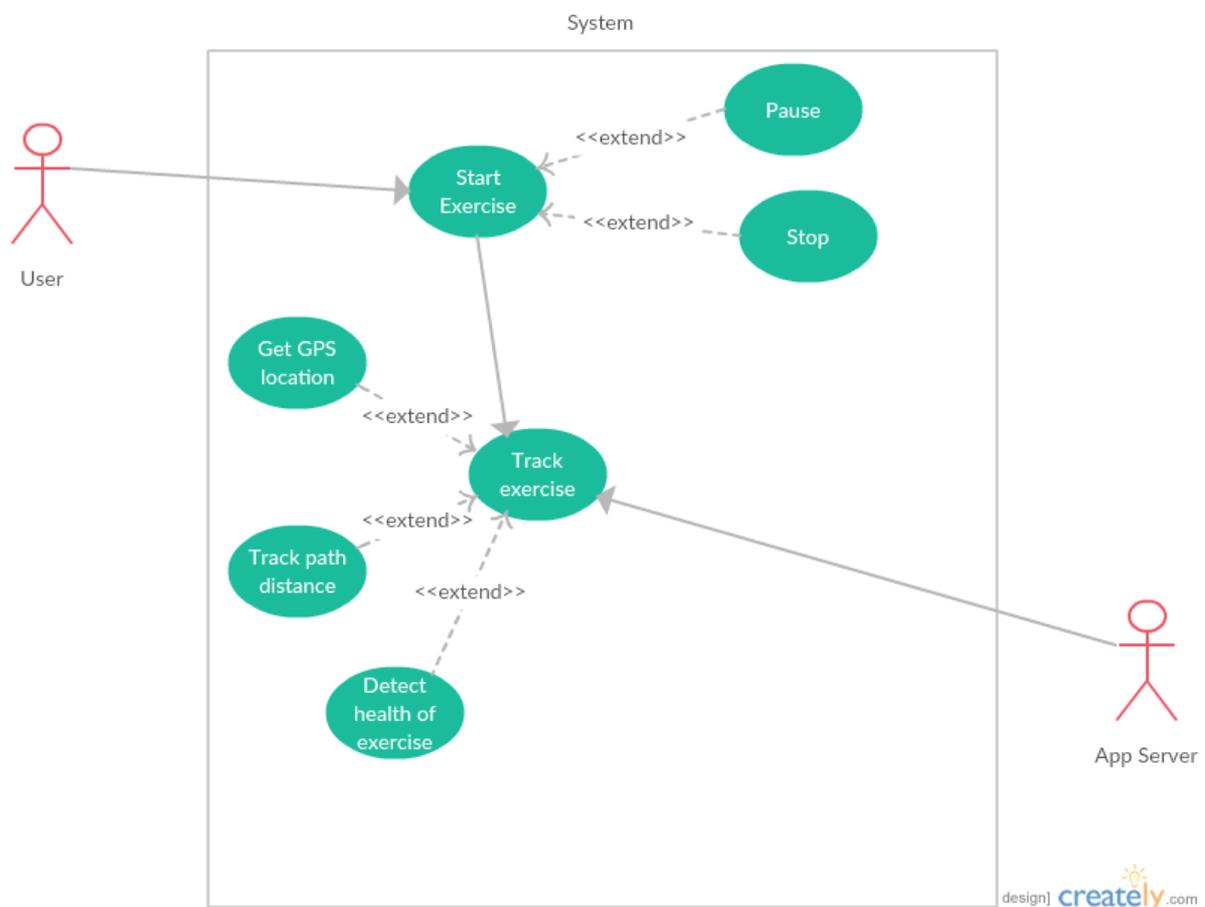
Scope

The scope of this exercise is to track the user's exercise by determining its GPS location and tracking its overall progress on a map and providing fitness statistics.

Description

This use case describes the process of how the user is being tracked by GPS and how the app is determining its fitness progress.

Use Case Diagram



Flow Description

Precondition

The app awaits the user to start the exercise

Activation

The use case starts when the user starts the exercise

Main flow

1. The user starts the exercise
2. The app starts tracking the exercise using GPS location, calculating distance and tracking fitness progress
3. The user pauses the exercise
4. The app pauses the current state
5. The user resumes the exercise
6. The app resumes the exercise
7. The user stops the exercise
8. The app stops the exercise

Alternate flow

1. The user puts the app in the background
2. The app pauses its current state
3. The user navigates to the app's paused state
4. The app displays a helpful dialog box asking the user to stop or resume exercise
5. The user is brought to main flow 6

Exception flow

1. The user has no Wi-Fi or GPS connection
2. The app presents a friendly dialog presenting the user to enable Wi-Fi or GPS connection
3. The user reinitialises connection and is brought to main flow 6

Termination

The app presents the next screen for the user

Post condition

The app records the statistics gather into the server's database.

Requirement 4 – View finished activity statistics

Description & priority

When the user has finished their exercise they are brought to review their session.

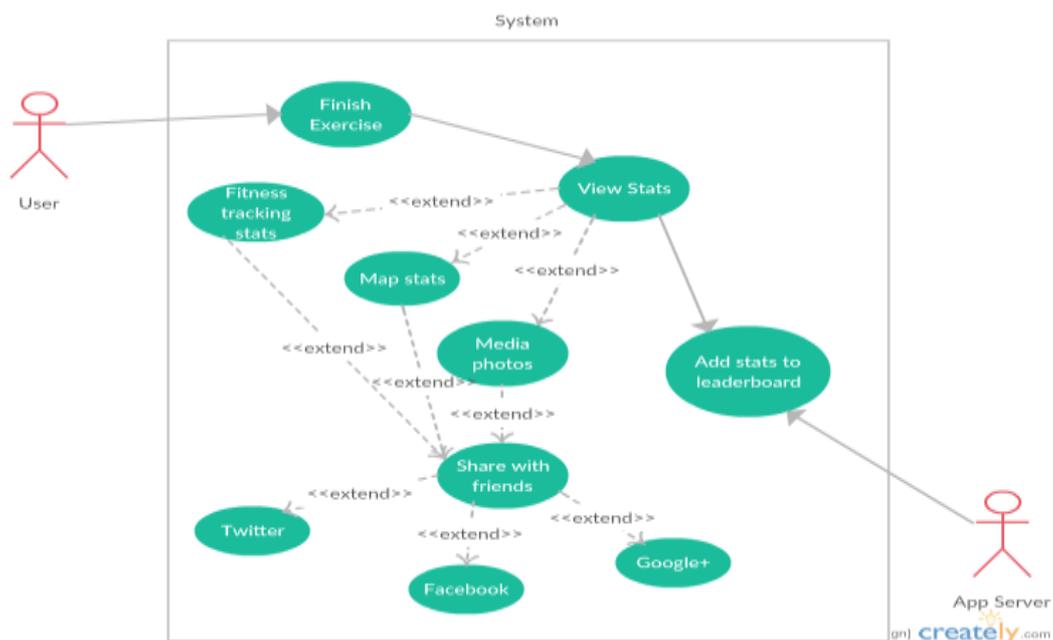
Scope

The scope of this use case is to provide the user to review this current exercise.

Description

The user will be presented with their fitness progress, map activity, and their photos if they captured any. They will have the ability to share this content with their friends.

Use Case Diagram



Flow Description

Precondition

The app waits for the user to finish exercise

Activation

The user finishes their exercise

Main flow

- User finishes exercise
- App sends user to review their session
- App adds statistics to the leaderboard
- User views fitness stats, map activity and photos if captured any
- User shares content via social media
- App sends request to target

Alternate flow

- User exits activity
- User can retrieve its exercise history via the next use case requirement

Exception flow

- The user has no Wi-Fi connection
- The app presents a friendly dialog presenting the user to enable Wi-Fi or GPS connection

Termination

The user exits the activity

Post condition

The persists the data into the app server's backend

Requirement 5 – Viewing My Activity

Description & priority

The app allows the user to view its past and current exercises to either share with its friends via social media or analyse the data to improve their overall fitness.

Scope

The scope of this use case is to provide the user the ability to view its current and past exercises.

Description

This use case describes the ability to view current and past exercises for analysing and sharing with friends

Use Case Diagram



Flow Description

Precondition

The app awaits user interaction

Activation

The user selects an exercise to view

Main flow

1. User selects to view activity
2. The app presents exercises history (past and present)
3. User views a specific exercise history
4. The app presents the screen
5. The user receives map, fitness statistics and media
6. The user shares an exercise via social media
7. The app sends the intent to the target
8. The user sends an app invite
9. The app provides the app invite intent to the target

Alternate flow

1. App can't find any current or past exercises
2. User is displayed helpful information prompting them to start an exercise
3. App records new exercise and brings user to main flow 1

Exception flow

1. User attempts to share content without WiFi connection
2. App displays helpful message that the user is not connected to WiFi
3. User initialises WiFi connection
4. App brings user to main flow 6

Post condition

The app awaits interaction from the user

Requirement 6 – Leaderboard

Description & priority

The app displays a leaderboard of all its users to promote a healthy competition and allows those to learn from others to improve their own fitness.

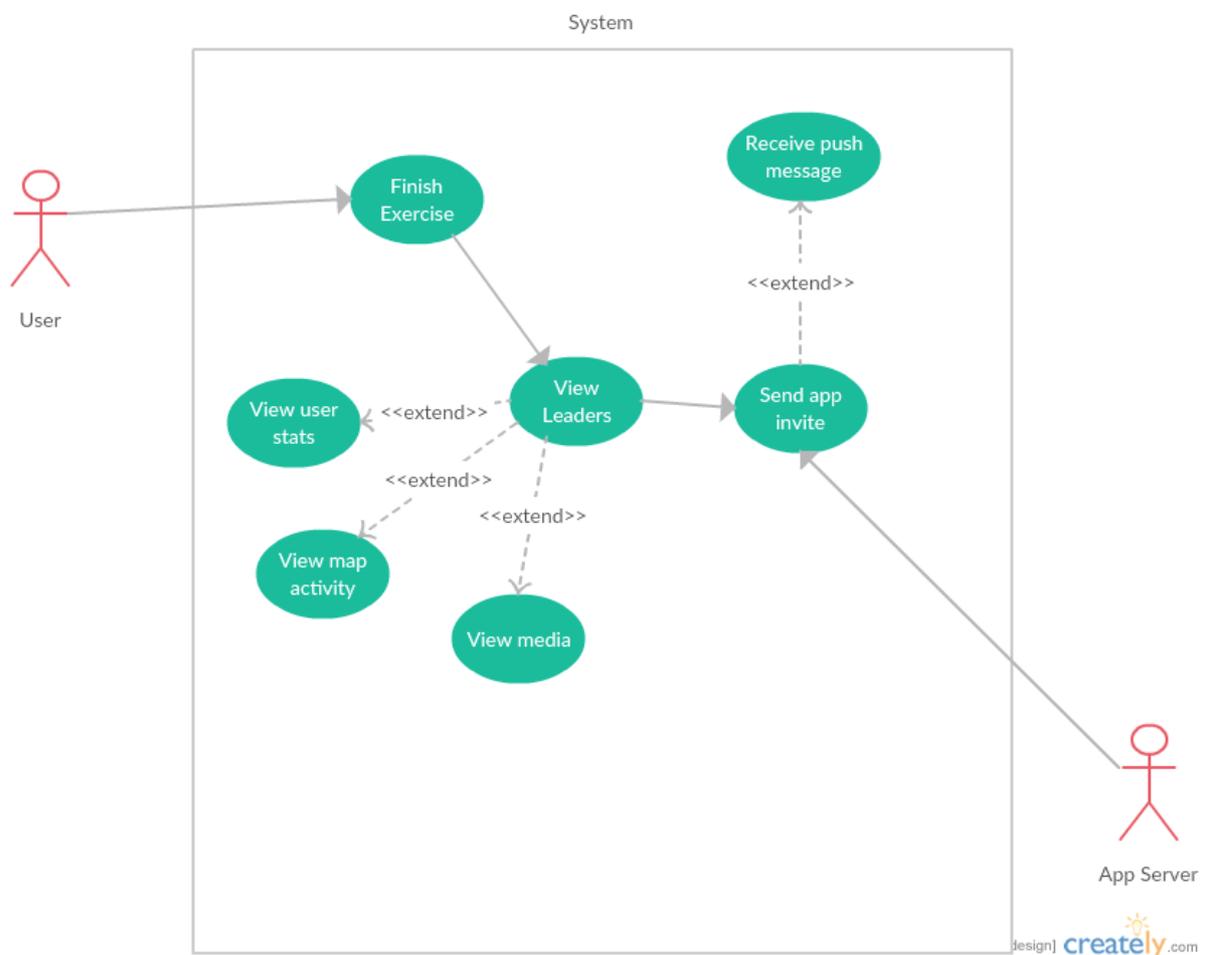
Scope

The scope of this use case is to provide the users to view the leaderboard rankings and view their competitor's data

Description

The use case describes the ability to view competitors on the leaderboard and view their statistics in order to beat their score

Use case diagram



Flow Description

Precondition

The app awaits user interaction

Activation

The finishes its exercise and views their score on the leaderboard

Main flow

- App displays the leader rankings
- User views the leader rankings
- User sends app invite to prompt their friends to challenge their score
- App sends app invite to intended target
- User receives message if friend beats their leader rankings

Alternate flow

- User is offline
- App displays leader rankings prior to user's latest connection
- User is brought to main flow 1

Post condition

The app awaits user interaction

Non-functional requirements

Performance

The performance of the app will be thoroughly tested against all the supported Android devices.

Types performance testing that will be implemented:

- Client side testing
 - User responsiveness to interaction
 - User experience testing
- Network testing
 - Test the network on low bandwidth
- Server side
 - Test the server for traffic load

Availability

To ensure availability at all times, it will allow users to use the app while there is no internet connection (offline). In order to test network availability, throttling the network connection for low bandwidth will determine its availability strength.

Network throttling is the intention of slowing of internet service by an ISP. Charles proxy program will allow for testing the network connection.

Robustness

The app will ensure that it will gracefully handle error exceptions using robust code which is a style of programming that prevents termination or unexpected crashes.

Robustness testing will be implemented to detect vulnerabilities of a component under unexpected inputs or in a stressful environment.

Security

The app will hold personal information from its users which it will need a strong encryption protocol to protect identity theft and exploits.

There will be research investigated into Android encryption libraries that use AESCrypt which is an API that performs encryption with no dependencies required.

Portability

Portability is important in Android applications as we need to cater for different support versions.

It will target and support the following Android API versions:

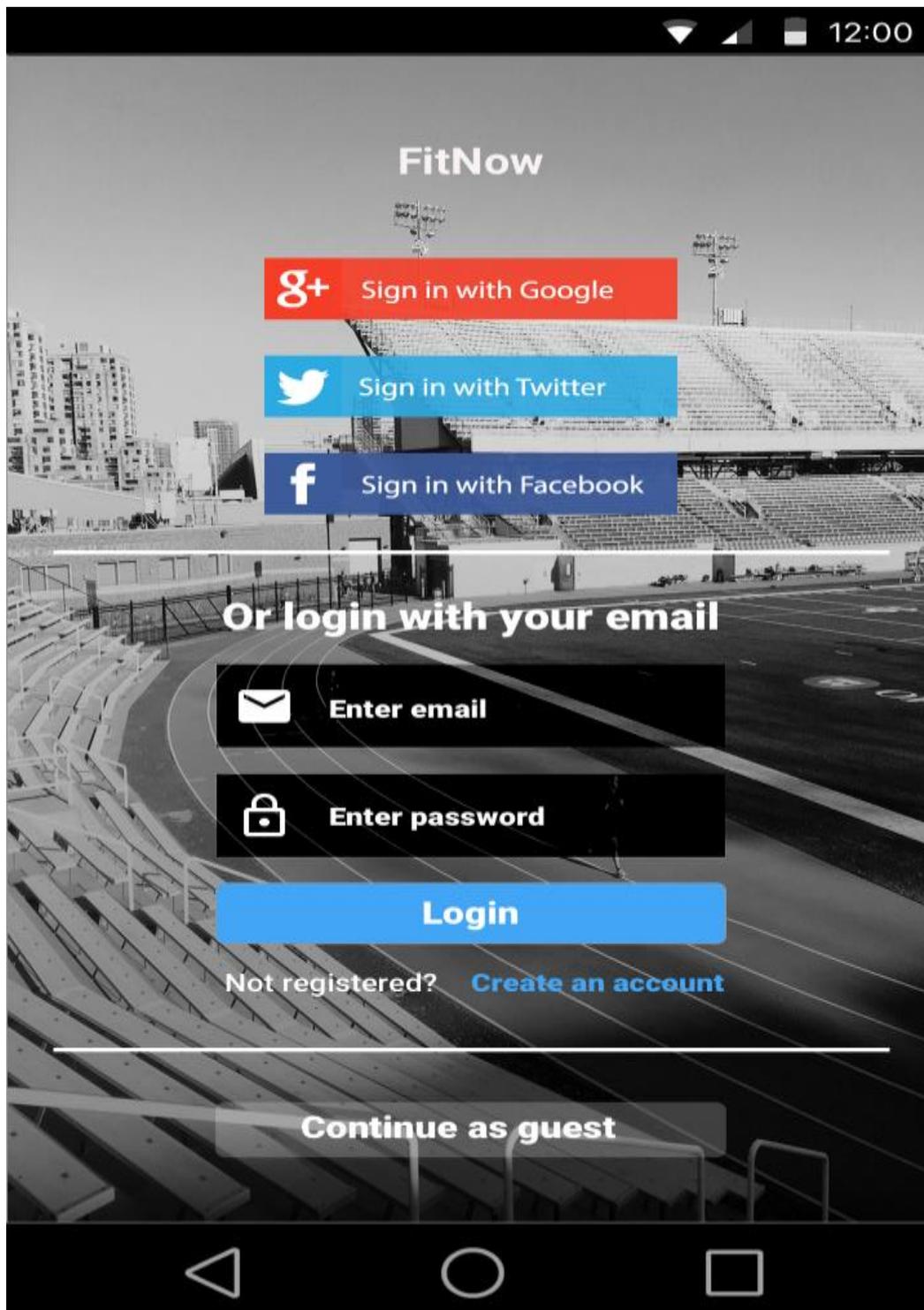
1. Kitkat
2. Lollipop
3. Marshmallow
4. Nougat

There will be automated and manual testing against all supported versions to ensure portability at all times.

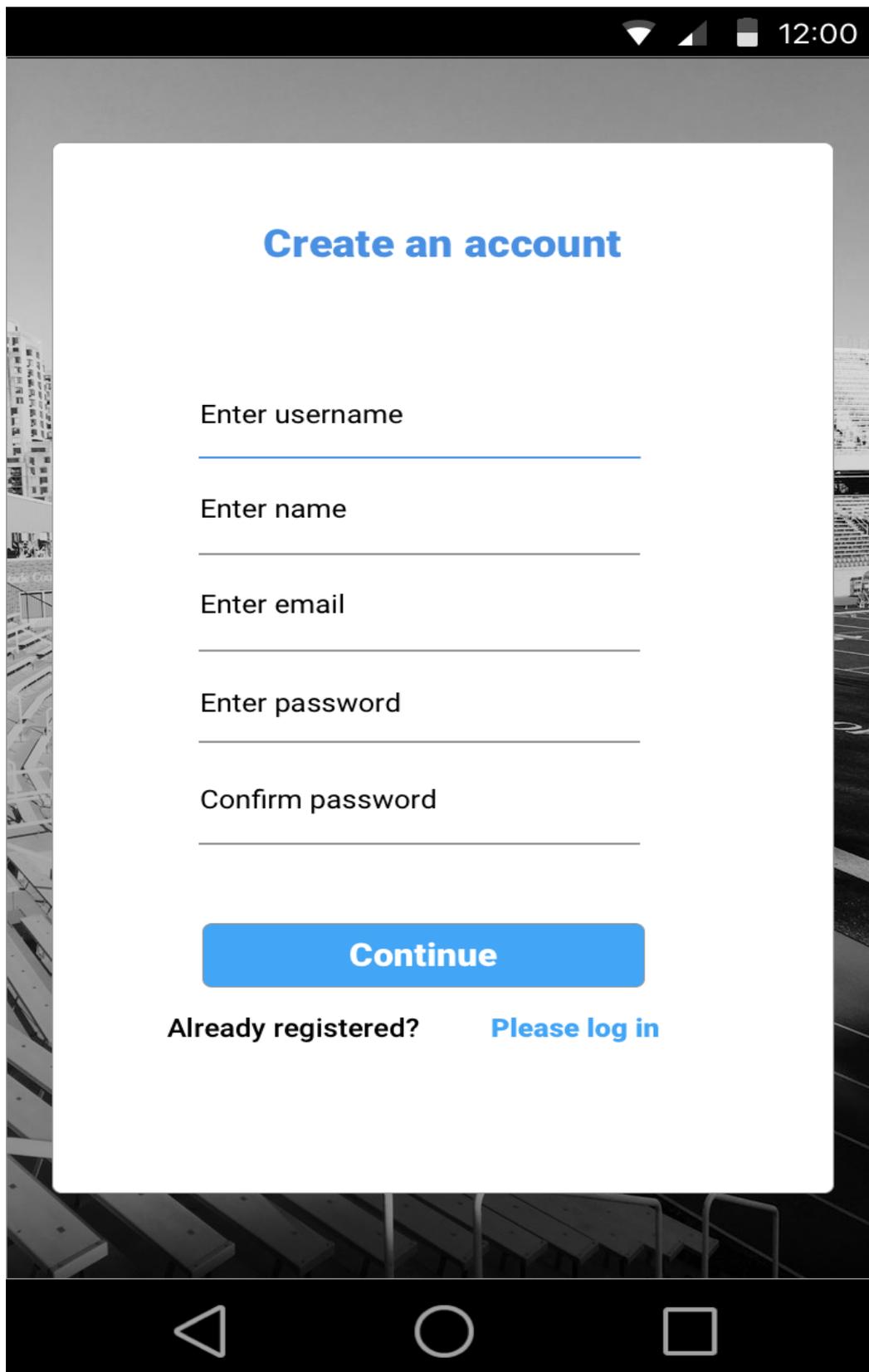
4. Interface Requirements

GUI

Login



The design attempts to following the use case outlined in the functional requirements on taking user's personal information via social networks or registering a standard account.

A mobile application registration screen with a white background and a blue header. The screen is overlaid on a dark background showing a blurred image of a building. At the top right, there is a status bar with icons for Wi-Fi, signal strength, and battery, and the time 12:00. The main content area contains a white card with the title "Create an account" in blue. Below the title are five input fields: "Enter username", "Enter name", "Enter email", "Enter password", and "Confirm password". Each field has a horizontal line below it. At the bottom of the card is a blue button with the text "Continue". Below the button, there are two links: "Already registered?" and "Please log in". At the very bottom of the screen is a black navigation bar with three white icons: a triangle, a circle, and a square.

Create an account

Enter username

Enter name

Enter email

Enter password

Confirm password

Continue

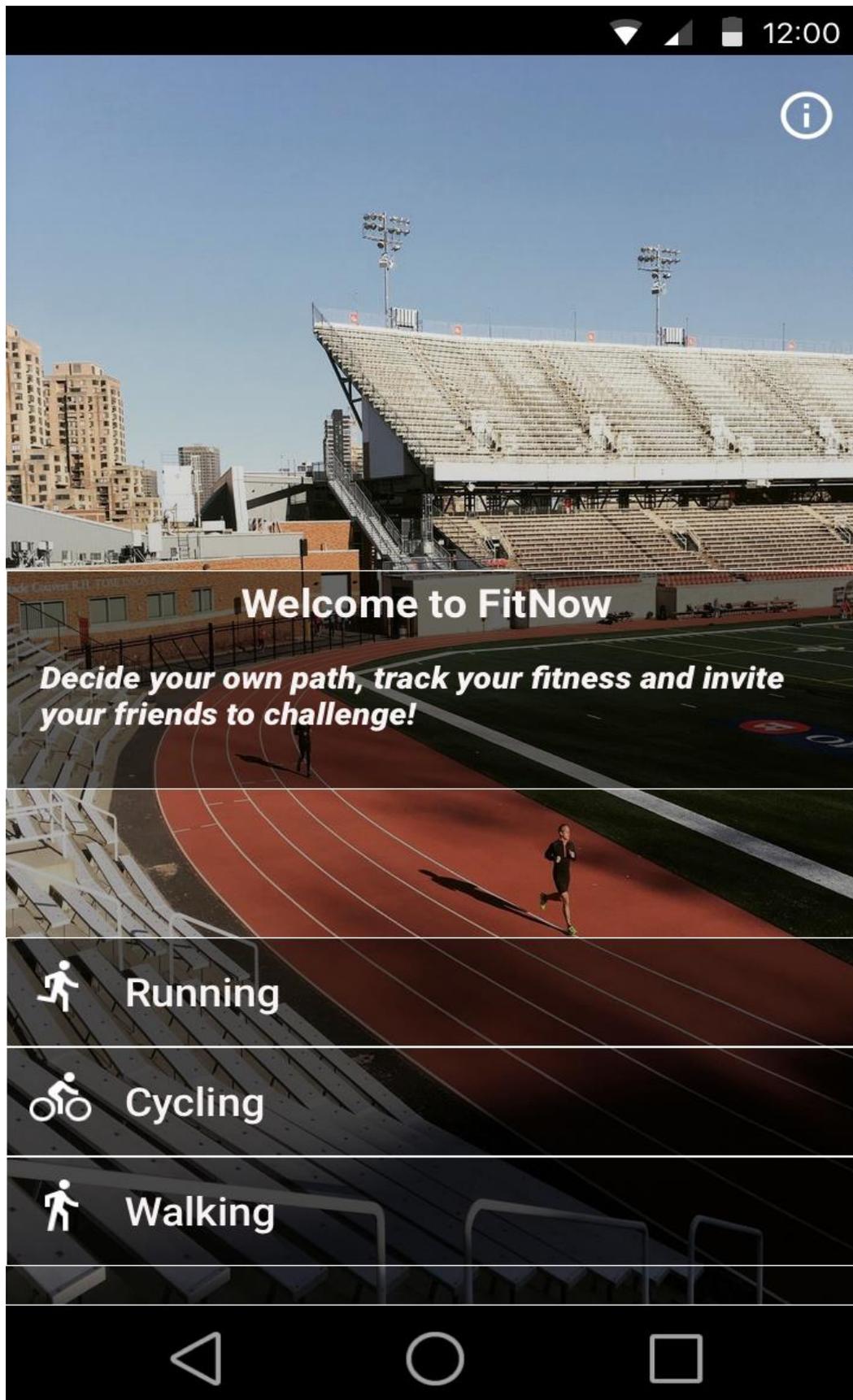
Already registered?

[Please log in](#)

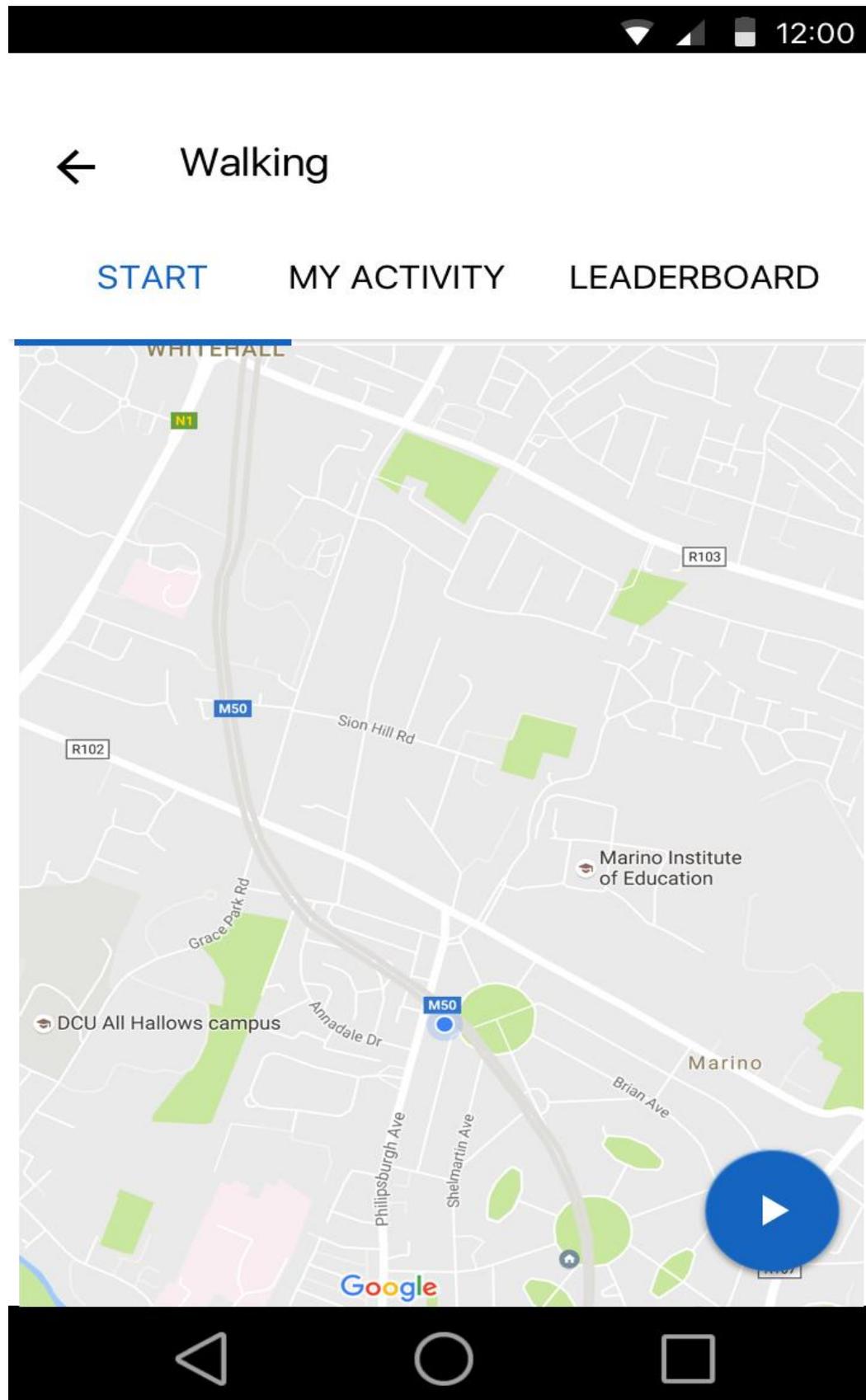
This design allows the user to create a standard account for identity authentication without using a social network account.

It takes the following input:

- Username as a unique identifier to recognise the user
- Name to display on the leaderboard
- Email to register the account and confirm
- Password to authorised the user



The user is presented the home screen if they are successfully authenticated. They have the ability to specify their own activity exercise from 3 specific categories; running, cycling and walking.





03:13

 **3.2 km**

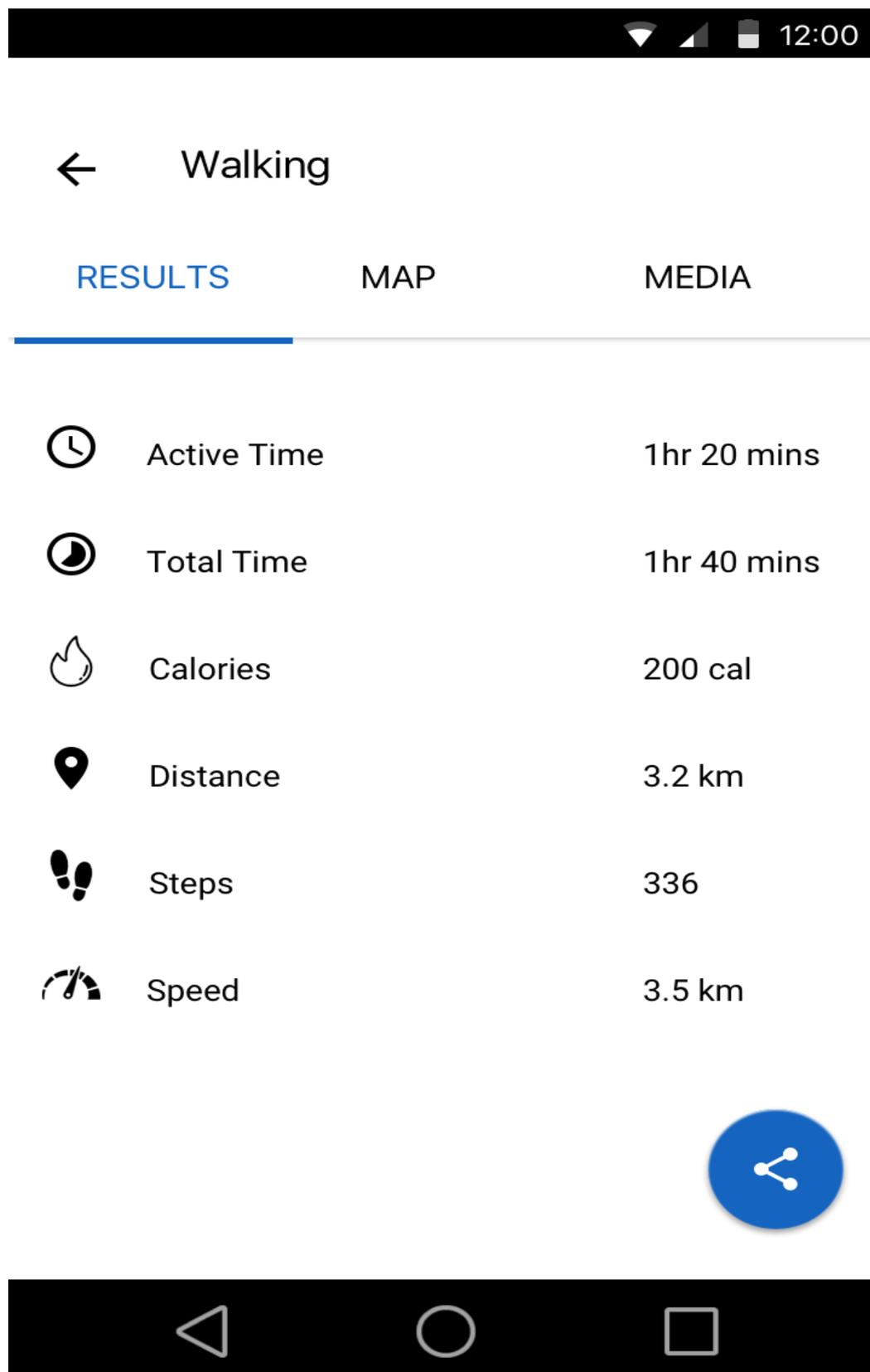
 **0**

 **130 cal**

 **3.2 km**



The user is presented with the following screens from their chosen category. The user's exercise is recorded in real-time using their device's sensors to detect their progress. They have the ability to pause the session to take a photo of their surroundings and also stopping the exercise to end their activity.

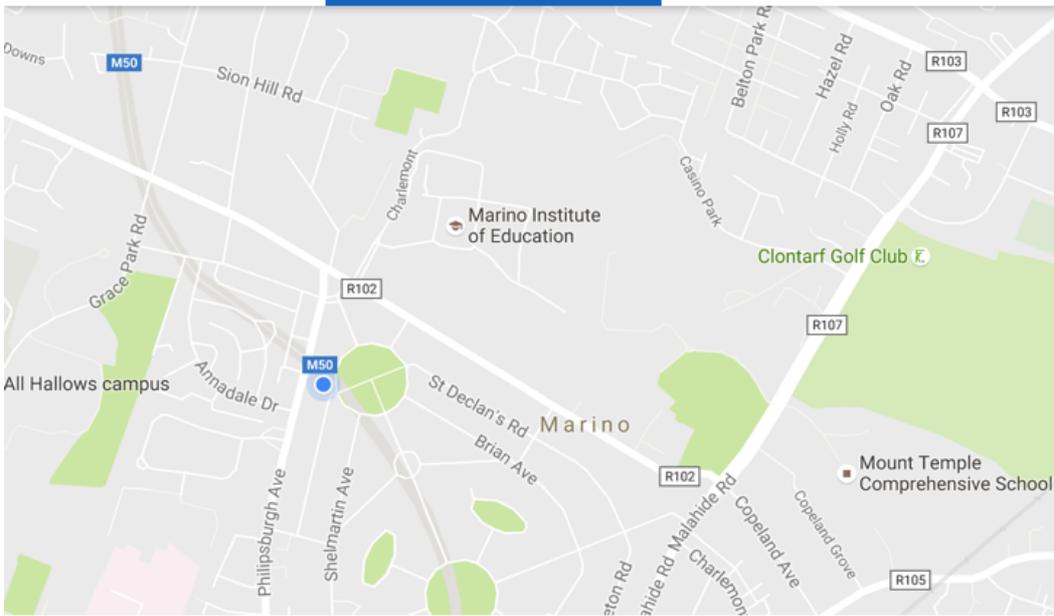


← Walking

RESULTS

MAP

MEDIA



Medium ▼

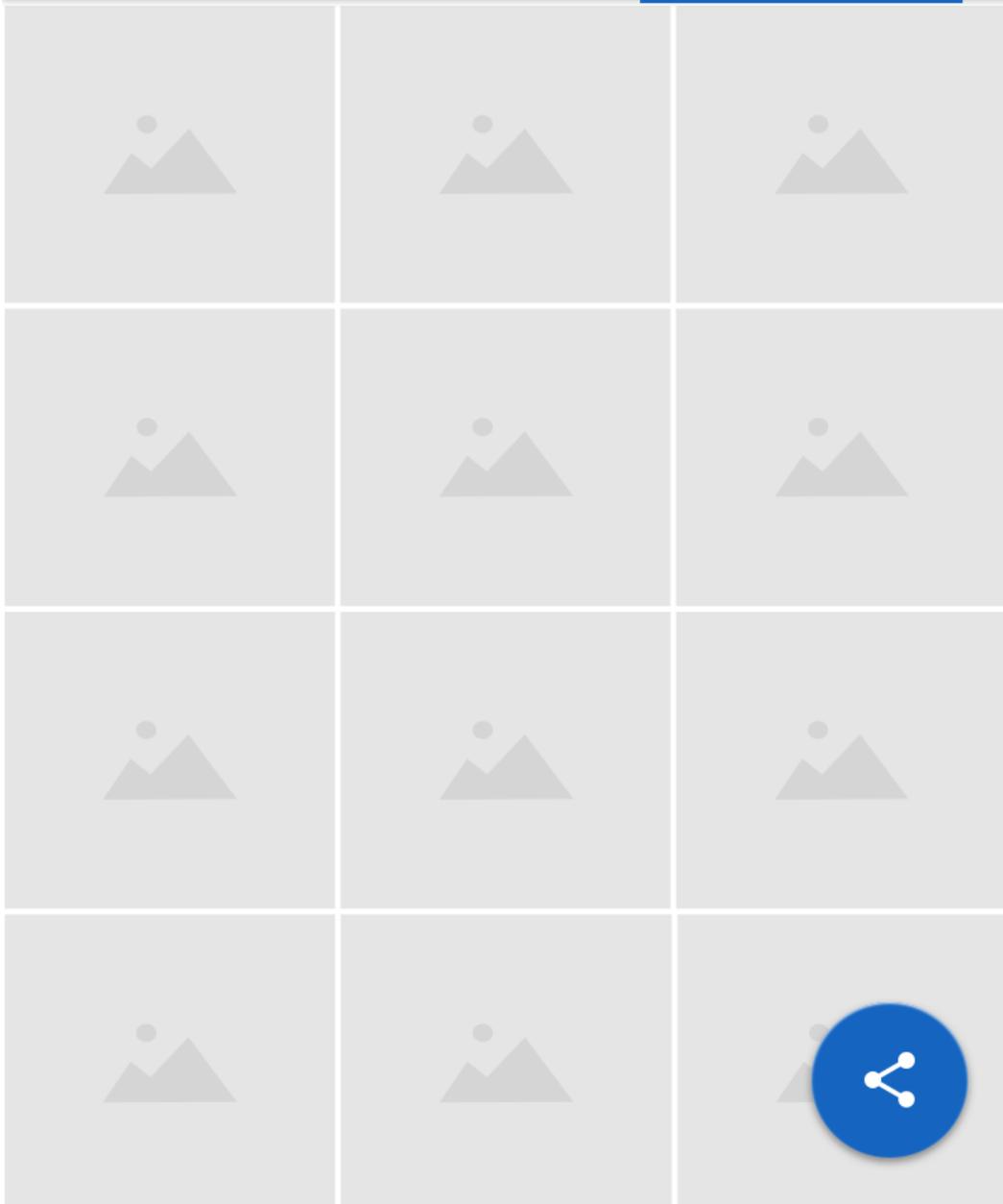


← Walking

RESULTS

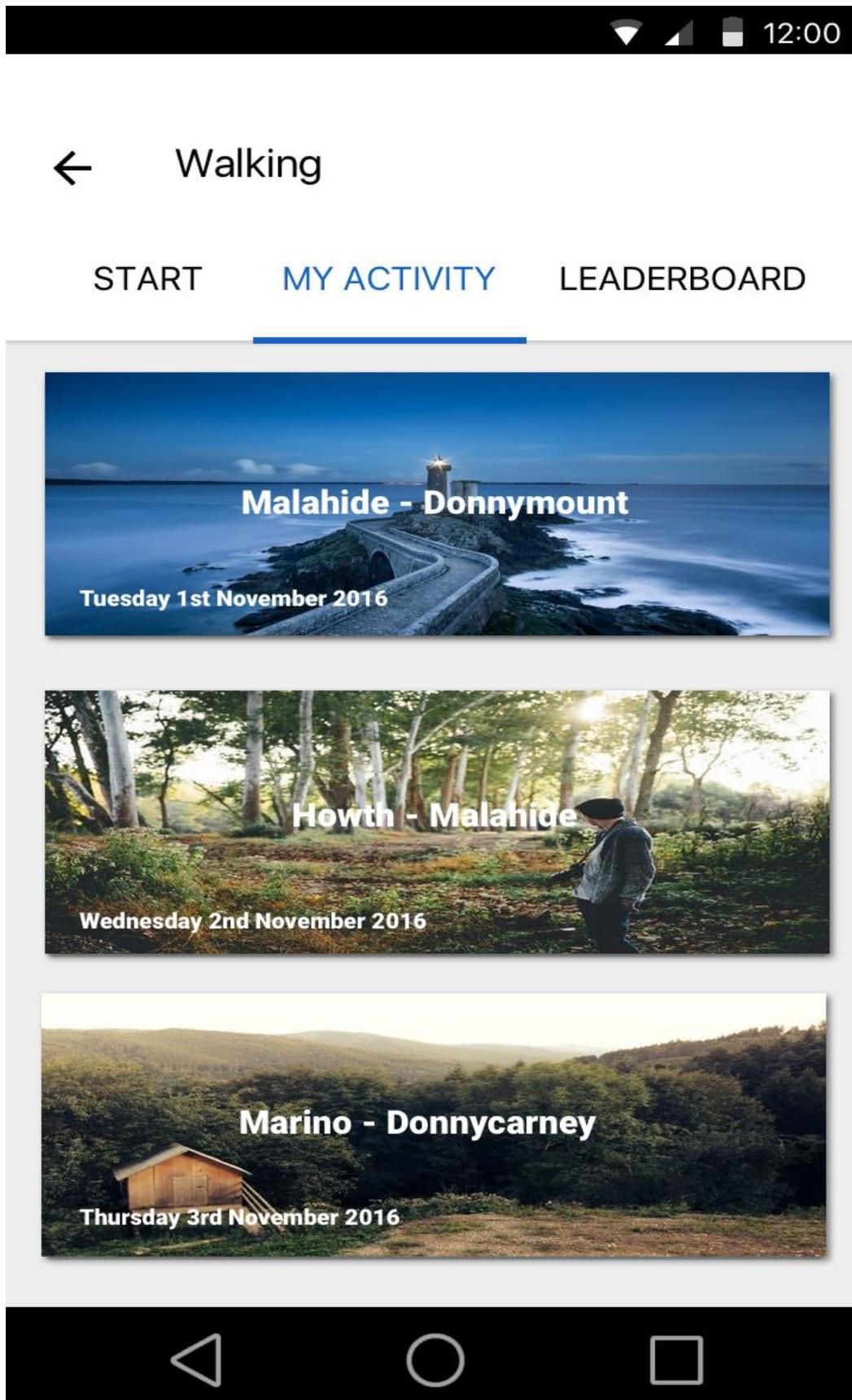
MAP

MEDIA



The above screens show the result of their exercise by displaying statistics that will allow users to analyse their performance and rate their experience. They also can view their photos taken during the exercise and share them via their preferred social network.

My activity screen



User can view their past or present exercises by tapping on the card and are presented with the results which was shown in the previous designs.



← Walking

START

MY ACTIVITY

LEADERBOARD

1st  Maggie Thatcher 
1601 pts

2nd  Donald Trump 
1443 pts

3rd  Bertie Ahern 
1335 pts

4th  Angela Merkel
1027 pts

5th  Pablo Esobar
780 pts



A leaderboard will be displayed as a competition element to encourage users to improve their exercises and learn from other users. It will present the users who has the most points collected and can be filtered by your local area or your list of friends.

Application Programming Interface (API)

The android app will be a native application using the Android Java APIs from the SDK packages. The app will not be exposing any information via an API. Instead, it will use public APIs to build the infrastructure of the app.

However, the app will be using APIs provided by Google to provide the infrastructure.

The following APIs will be used:

- **Google Maps services including Location API**

Google maps will allow the app to display the user's location and ability to display visual information on a map.

- **Google Fit API**

Google provide a service for fitness data via sensors from their Fitness API. This will the app to develop features by reading fitness data from the user.

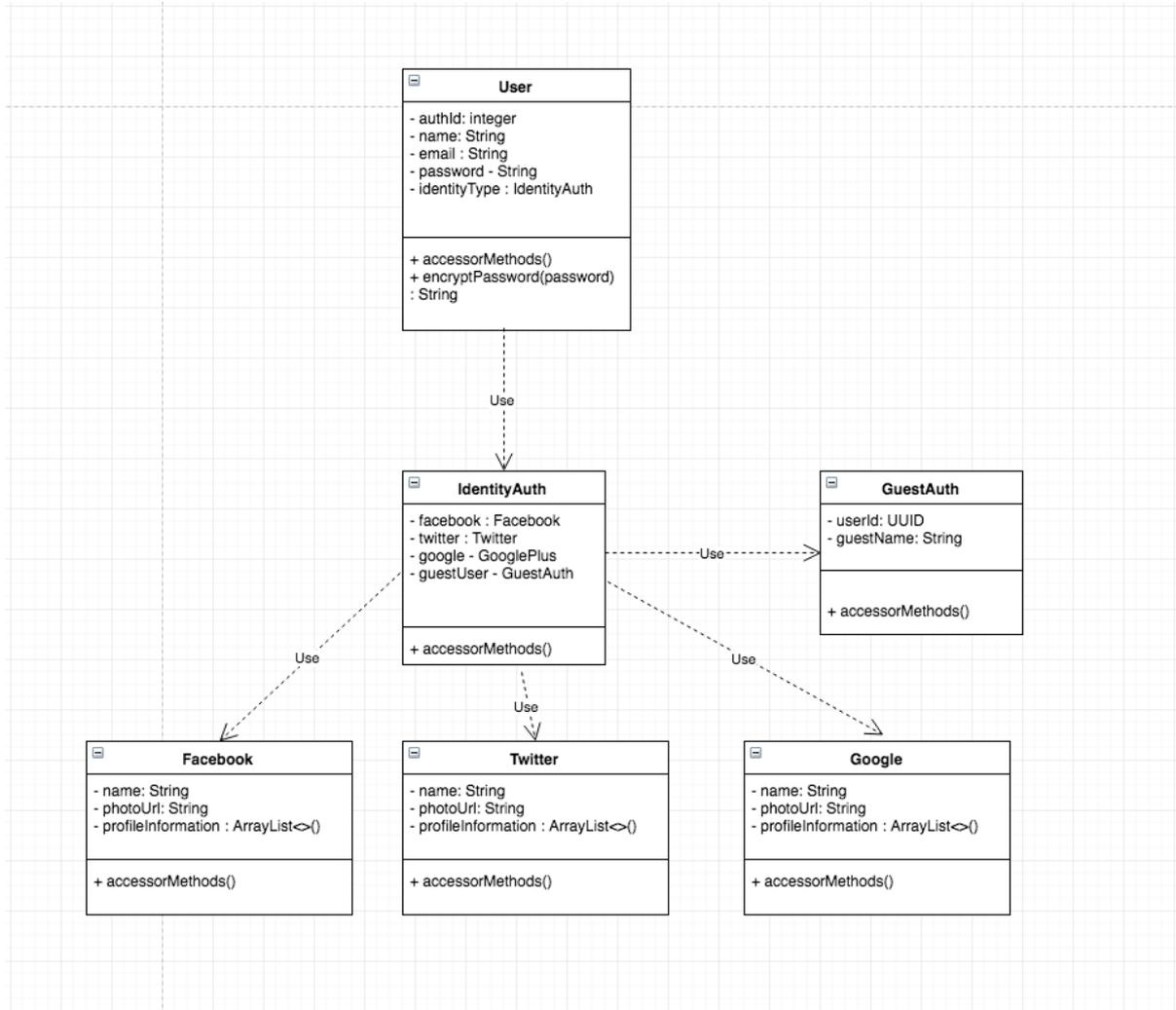
- **Firebase API**

Firebase API is a cloud backend as a service which is a tech start-up firm acquired by Google. It provides all Google APIs packaged into one single SDK. The services include cloud messaging platform, authentication, Realtime database, storage, test lab, crash reporting and much more!

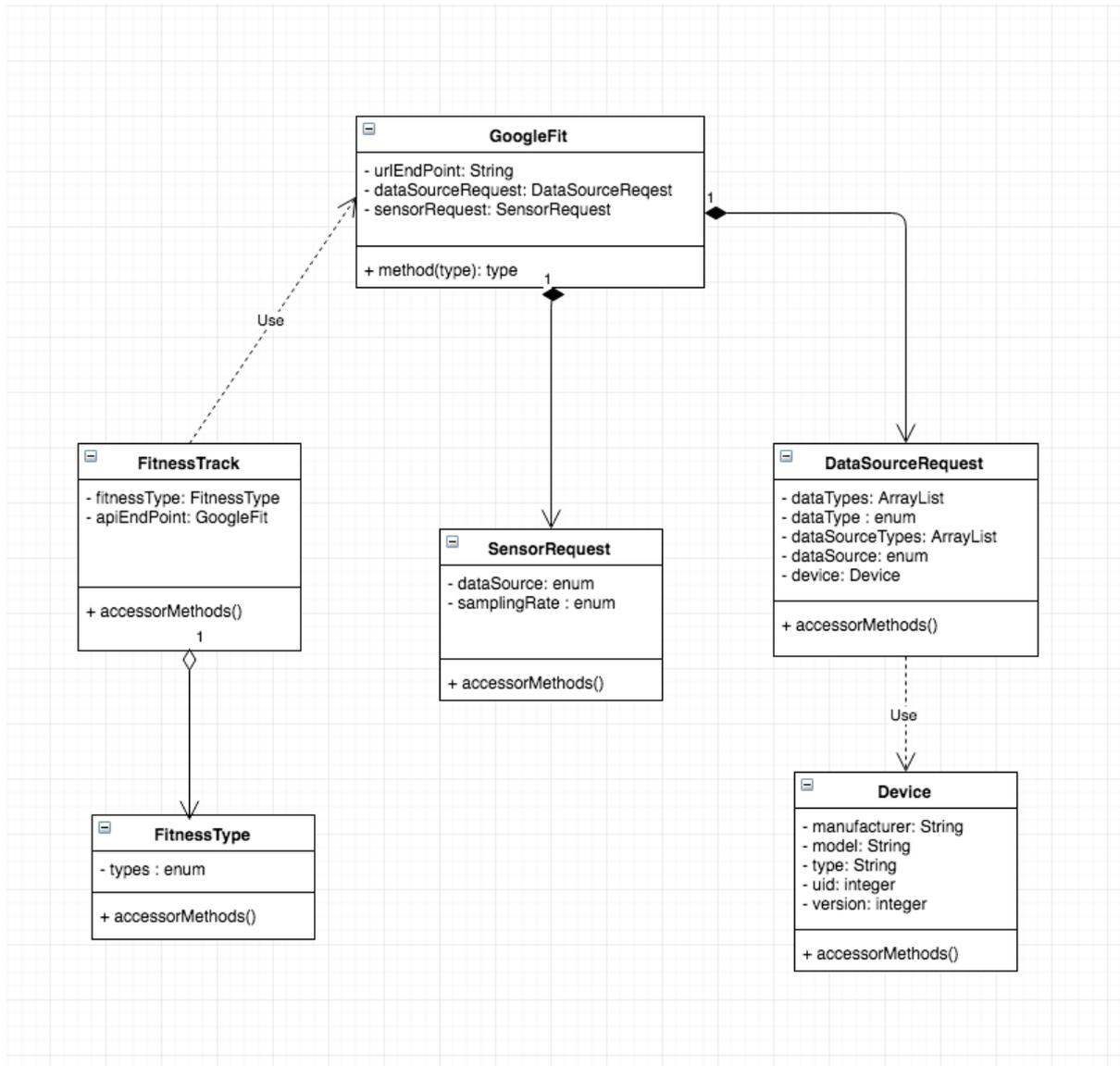
There will be further research into more APIs that can help speed up the development of the project.

5. System Architecture

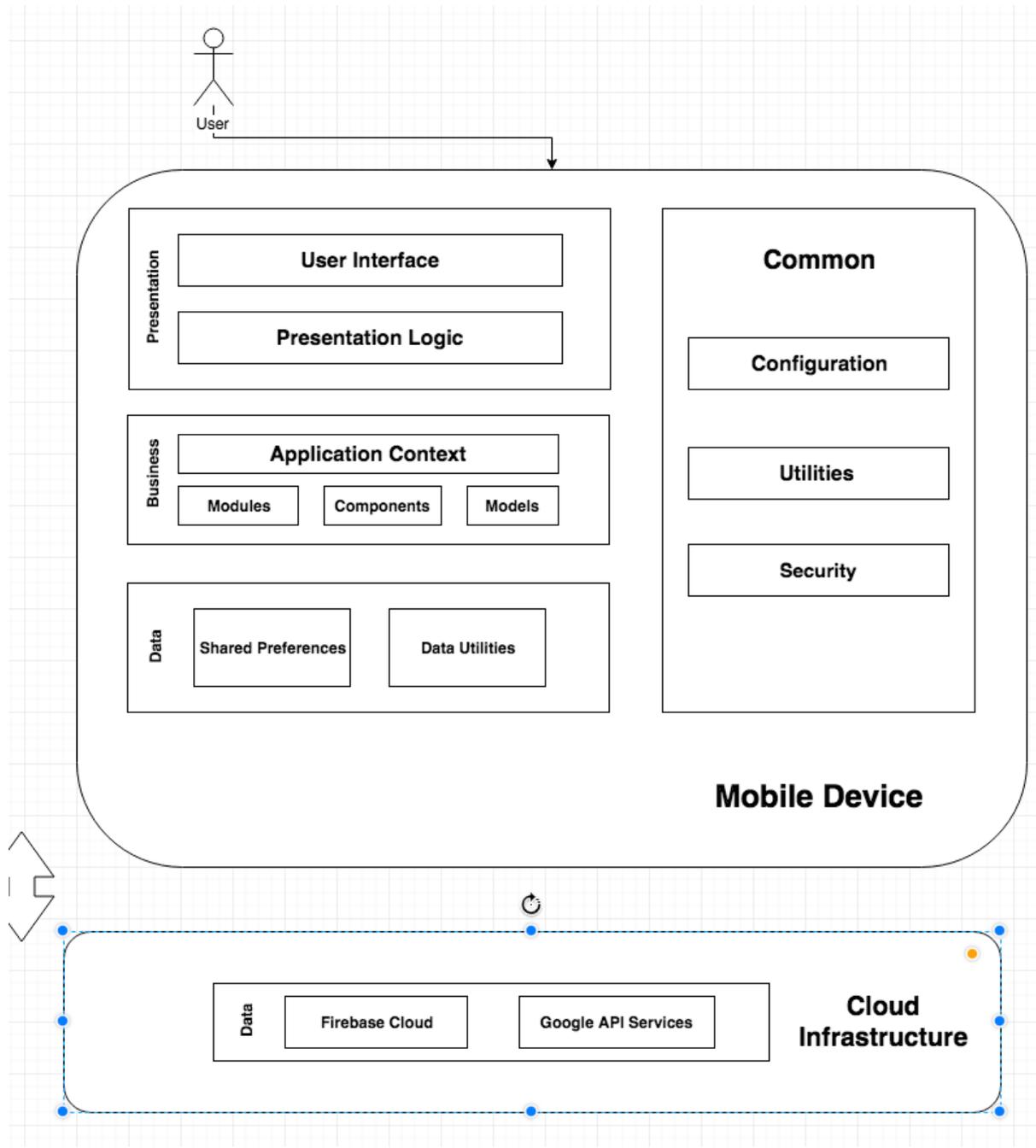
User class diagram



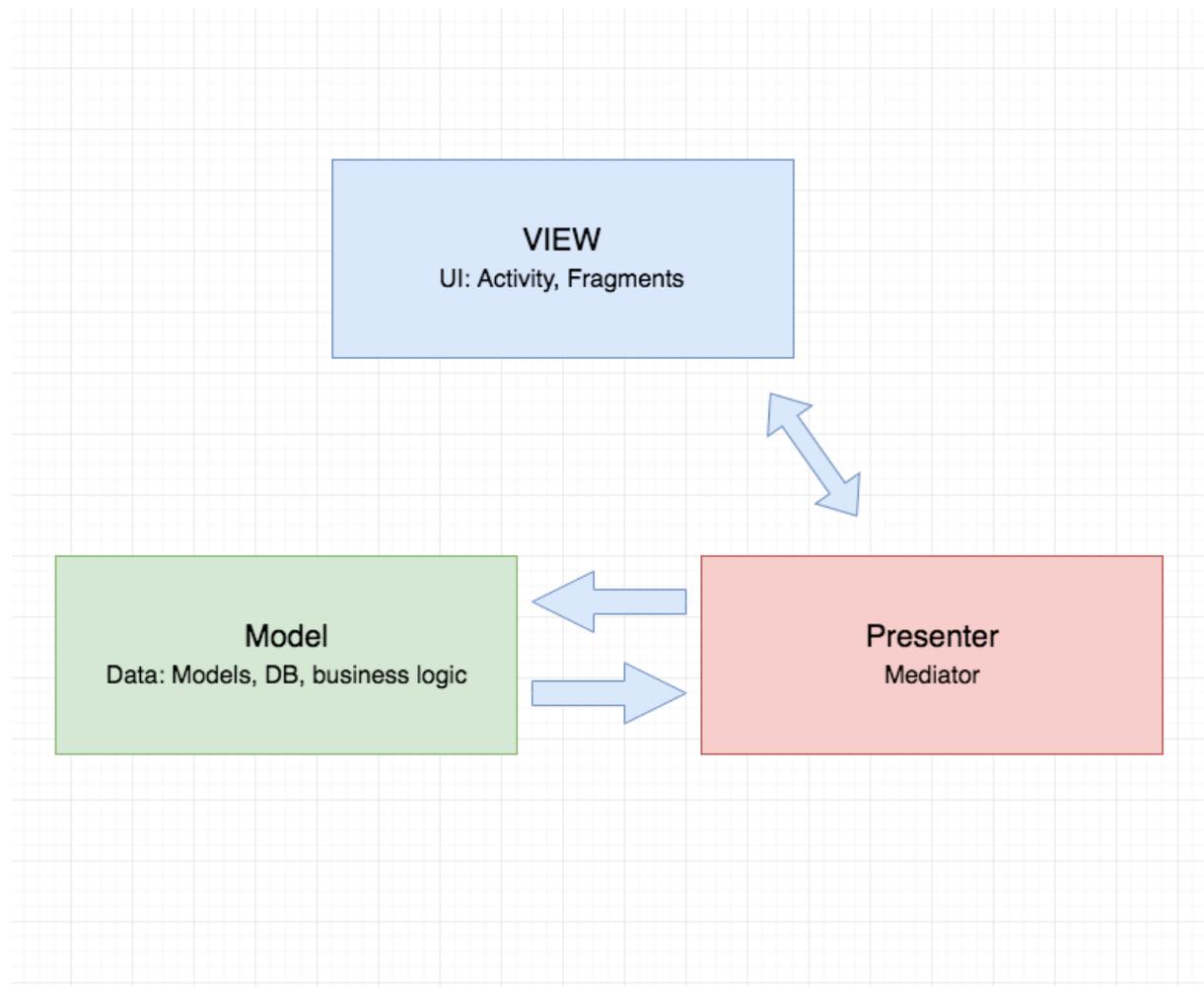
Fitness tracking class diagram



Architecture diagram



Model View Presenter design pattern diagram



6. System Evolution

- The application can be evolved into Android and Apple wear devices to allow users to easily track their exercises.
- The application can be extended to iPhone devices using the same backend infrastructure, thanks to Firebase's portability.

Monthly Journals

Reflective Journal #1 September

Student name: Ian Arbuckle

Programme (e.g., BSc in Computing): BSc in Computing

Month: September

My Achievements

This month, I was able to pitch an idea to three judges and unfortunately the first time I was rejected. My idea didn't have a large enough scope to be able to expand the project and add more features. However, after discussion with the judges afterwards I decided to pitch another idea which had a large enough scope and even better than my original idea. It got accepted and I will continue to do the necessary research in order to plan the project goals. My contributions to the project this month included getting my pitch accepted by the judges and doing necessary research to start the project.

My Reflection

I felt, it worked well to reflect on my original idea which did not have a flexible scope and discuss with the judges on how to pitch another idea that allowed me to plan a larger scope. However, I was not successful in my pitch for my original idea and held a discussion with the judges afterwards to gather feedback on how to improve my new idea.

Intended Changes

Next month, I will plan out my project scope using agile methodology which will involve backlogging stories (features) and designs that I will plan to implement over the coming months. I will also plan to complete the project proposal.

I realised that I need to do research on the area in my project and find tools that will help along the way.

Supervisor Meetings

Date of Meeting: N/A

Items discussed: N/A

Action Items: N/A

Reflective Journal #2 October

Student name: Ian Arbuckle

Programme (e.g., BSc in Computing): BSc in Computing

Month: October

My Achievements

This month, I was able to investigate into technologies that I can incorporate into my project. I've attended a Google Developer Group DevFest event, where I attended presentations on a cloud backend as service technology called Firebase, artificial intelligence public APIs and how to market your app for product design. Firebase in particular has grasped my attention as it allows us to use all the Android APIs into one single SDK. My contributions to the projects included implementing the home screen and successfully integrating the Google Location API for tracking the user's GPS location for the map feature. I've also completed my first draft for the software requirement specification and will attempt to get a review from my supervisor.

My Reflection

I felt, it worked well to investigate into technologies that I can integrate into my project. The Google DevFest event has allowed me to keep up-to-date with the latest tech and increase the potential scope of the project.

However, I was not successful in getting the necessary time to complete my development sprint in time. This was caused by the workload from other modules, with Artificial Intelligence which they've asked us to develop a full chess game component. I didn't realise that my other modules will take up the bulk of my time as the software project alone, is worth 40 credits! To solve this workload problem, I will attempt to finish them as soon as possible.

Intended Changes

Next month, I will try to allocate two full development days to my software project and implement the necessary features in my planned development sprints.

I realised that I need to organise my other modules assignments and allocate time for my project.

Supervisor Meetings

Date of Meeting: 28th October 2016

Items discussed:

- Project Proposal Review
- Software Requirements Specification
- Discuss blockers from other modules

Action Items:

- Create a first draft for software requirements specification

Reflective Journal #3 November

Student name: Ian Arbuckle

Programme (e.g., BSc in Computing): BSc in Computing

Month: November

My Achievements

This month, I was able to integrate the Firebase API into my project which allows to implement a real-time database and access a suite of tools such as analytics, push notification manager engine, identity authentication, storage, hosting and test lab.

My contributions to the projects included implementing the identity authentication with social accounts and also username/password to demonstrate a working prototype for mid-point presentation.

My Reflection

I felt, it worked well to complete the necessary documents before the mid-point presentation and allow me to focus my attention to implementing the prototype.

However, I was not successful in implementing the Google Fit API into the app as I've had issues working with the samples provided by Google. However, this suggests that I need to investigate the documents and find the latest implementations of the API. Although, the Google Location API is fully integrated and will demonstrate that as part of the prototype.

Intended Changes

Next month, I will finish the Firebase identity authentication in time for the mid-point prototype with the finished user interface. I will also investigate into integrating the Google Fit API with the latest implementation and will put it in the prototype if successful.

I realised that I need to track my time during the spring using a burndown chart. My project management of choice is trello and I've found a burndown chart plugin that I will use to track my sprints.

Supervisor Meetings

Date of Meeting: 29/11/2016

Items discussed:

- Google Fit API issues

- Discussing free allocated time for the project

- Prototype expectations

- Technical Report review

Action Items:

- Technical Report feedback

- Implement a working prototype

- Investigate Google Fit API implementations

Reflective Journal #4 December

Student name: Ian Arbuckle

Programme (e.g., BSc in Computing): BSc in Computing

Month: November

My Achievements

This month, I demonstrated a prototype to the panel of lecturers and ensured them the project is work in progress. They were happy with my concept and strategic planning to implement the project's features. There was feedback provided that I should consider adding a badge as a reward element and only implement it if the scope allows it.

My contributions to the projects included demonstrating a prototype to work from for the coming months and evaluating feedback taken from the lecturers.

My Reflection

I felt that my presentation went extremely well from my perspective. Constructive feedback was provided by the lecturers such as researching the competitors that I've discussed about and reducing the content load on the slides.

Intended Changes

Next month, I will start a strategic planning for my future development sprints as soon as my terminal examinations are finished. As the final presentation is a lot closer than I think I will ensure that I utilise burndown charts to analyse my time spent on such features.

Supervisor Meetings

There was no scheduled meeting for this month as there was a lot of focus on completing projects from other modules. I've informed my supervisor of this and agreed to schedule a meeting upon my return in January.

Date of Meeting: N/A

Items discussed:

Action Items:

Reflective Journal #5 January

Student name: Ian Arbuckle

Programme (e.g., BSc in Computing): BSc in Computing

Month: January

My Achievements

This month, I've planned my future development sprints and decide what features will need to be implemented. My implementation of the user story for displaying a timer while the user has been successful. I have also successfully implemented the upload feature for storing user image content which is handled by the Firebase Storage API.

My contributions to the projects included implementing user story for the timer and refactoring the code base for the Location Services permissions model.

My Reflection

I felt that I've made significant progress this month in terms of implementation. I've managed to implement unit tests for Firebase Authentication and will continue to write unit tests for other components.

Intended Changes

Next month, I will create a usability test to ensure that the app is meeting quality user experience standards. I will also ensure that I finish the refactoring of the Location Service and that it will not hamper the development of further features.

Supervisor Meetings

Date of Meeting: 23/01/2017

Items discussed: Mid-point feedback

Action Items: Continue development work of features and testing

I met with my supervisor to revise my performance in the mid-point presentation and how I can improve for my final presentation. We discussed the marking scheme for the final prototype and presentation and where I need to focus on.

Reflective Journal #6 February

Student name: Ian Arbuckle

Programme (e.g., BSc in Computing): BSc in Computing

Month: February

My Achievements

This month, I've planned to implement and refactor the live sensors for walking activities. When the sensor implementation is completed, the user will be displayed their results and have the option to save their session.

My contributions to the projects included implementing and refactoring the user story for live sensor data and displaying session results to the user.

My Reflection

I felt that I've made significant progress this month in terms of implementation. I've managed to implement user story for live sensor data for walking activities and created more unit test coverage.

Intended Changes

Next month, I will focus on the implementation of storing activity session results in a real time database and create the leader board.

Supervisor Meetings

Date of Meeting: 10/03/2017

Items discussed: Update of development

Action Items: Continue development work on new features

I met with my supervisor to provide update on my development progress and we are both aware there are significant development work left to be completed.

Reflective Journal #7 March

Student name: Ian Arbuckle

Programme (e.g., BSc in Computing): BSc in Computing

Month: March

My Achievements

This month, I planned to implement and complete the displaying of session results, sending to realtime database and creating image gallery of photos during the session.

My contributions to the projects included completing the features above and writing unit tests for code coverage.

My Reflection

I felt that I've made significant progress this month in terms of implementation. I've managed to implement user stories for displaying of session results which is stored in the realtime database and completing the image gallery.

Intended Changes

Next month, I will focus on the implementation of implementing the cycling and running live sensor tracking.

Supervisor Meetings

Date of Meeting: 13/4/2017

Items discussed: Final preparation and progress on project

Action Items: Finished features in time for final presentation and deadlines