Department of Computer Science

Individual Project - CS3IP16

A Multi-User Cross-Platform Fitness Tracking Mobile Application

Student Name: Rory Fielding

Student Number: 22002003

Supervisor: Dr Lily Sun

29th April 2019

ABSTRACT

Fitness tracking applications aim to provide users with three main functionalities. These include the ability to plan exercise and manage calories efficiently in order to provide oversight into developing a user's overall health. These applications also aim to provide the functionality of tracking and analysing activity data, allowing the user to compare results against previous performances and gauge improvements to their fitness over a period of time. Fitness tracking applications aim to provide the functionality of entertainment, creating a fun and enjoyable tool to support a user's fitness goals. This report will detail the research undertaken to understand the features required to develop a fitness tracking application and also address why utilising React Native for a cross-platform framework was selected for this implementation. This will also detail how through utilising React Native, Redux and Firebase, a multi-user cross-platform mobile application can be built in order to handle fitness tracking functionalities. This report will demonstrate the success of this implementation, by creating a complete running tracking application that allows the user to manage their calorie intake, provide location tracking functionality in combination with calculation calorie expenditure, as well as providing the ability to analyse running statistics.

Dedication and acknowledgements

I would like to thank Dr. Lily Sun at the University of Reading for her continual support and guidance throughout the duration of this project.

I would also like to acknowledge Drew Rathbone, Head of Software development at Flexy for his advice in implementing Redux.

Furthermore I would like to give thanks to Robert Good-Stephenson, Tony Maynard and Sian Baker for their help in testing the final application.

CONTENTS

GLOSSARY OF TERMS & ABBREVIATIONS

API - Application Programming Interface

App - Application

BMI - Body Mass Index

CPU - Central Processing Unit

DOM - Document Object Model

EU - European Union

GPS - Global Position System

GPU - Graphics Processing Unit

IDE - Integrated Development Environment

JSON - JavaScript Object Notation

kCal - Kilocalories

MET - Metabolic Equivalent of a Task

OS - Operating System

PA - Physical Activity Intensity

PbD - Privacy by Design

SDK - Software Development Kit

TEE - Total Energy Expenditure

UAT - User Acceptance Testing

UI - User Interface

UID - Unique Identifier

URL - Uniform Resource Locator

LIST OF TABLES, FIGURES & EQUATIONS

1. INTRODUCTION

1.1 Project Background & Motivation

The first evidence of fitness trackers can drawn from 1780, where a Swiss horologist created a pedometer to measure steps and distance as an individual walks. In 1973, the first handheld mobile phone was created at Motorola by an engineer named Martin Cooper. Following this technical innovation, in 1985 Jiro Kato determined that 10,000 steps was the optimal daily step count to maintain a healthy body of an average Japanese man, at which point the first step counter on a mobile phone was marketed to consumers [1]. Wearable fitness tracking devices, such as wireless heart rate monitors that integrated with commercial-grade fitness equipment found in gyms, became available to consumers in the early 2000s, however it wasn't until 2006 that Garmin introduced GPS technology in order to track an individual's exercise route [2].

The concept of fitness tracking is rapidly advancing and with the advent of smart phone technology over the course of the last two decades, ever-increasing numbers of consumers are able to access and utilise mobile phone applications in order to enhance their own fitness. The purpose of this project is to build upon tools already created so a fitness tracker that incorporates new research and technology can be developed. This will aim to provide a modern useful tool for individuals to work with so they can improve their own health and fitness, positioning itself as a benefit to society.

1.2 Research Problem

The problem this project is addressing is how to build a multi-user cross-platform fitness tracking mobile application that communicates to a backend database. This requires initially developing an understanding of what features make a fitness tracking application, along with developing a complex system that is built for scale in terms of allowing multiple users to interact with the system at the same time, which will function on the latest versions of both Android and iOS mobile devices.

1.3 Aims and Objectives

Adhering to the Project Initiation Document (PID), the project should satisfy a number of different aims and objectives. The objectives below have been adapted from the PID in order to more accurately align to the scope of the final project.

The first objective of the project is to conduct research evaluating how to calculate an individual's daily calorie needs, as well as how to calculate the amount of calories an individual burns during a running exercise. Following this, market research will be conducted in order to evaluate the top fitness tracking applications on the market and analyse the most important features required to build a fitness tracking application. This will also involve consider analysing the best practices of cross-platform software development and evaluating cross-platforms frameworks that are available. Once research has bee completed, the following initial application design outputs will be created:

- Application screen wireframes
- Use cases for the application

- Flowchart to detail how the user interacts with the application
- High-level logics
- Optimised structure of database tables

Following constructing these outputs, an initial design of the application will have been completed. The design created will then be analysed in order to discover any potential flaws in the system and re-designed appropriately to consider any issues found in the analysis. The front-end prototype of the application can then be developed and once this has been completed, the backend database will be integrated into the system in order to allow the system to more efficiently store and retrieve data.

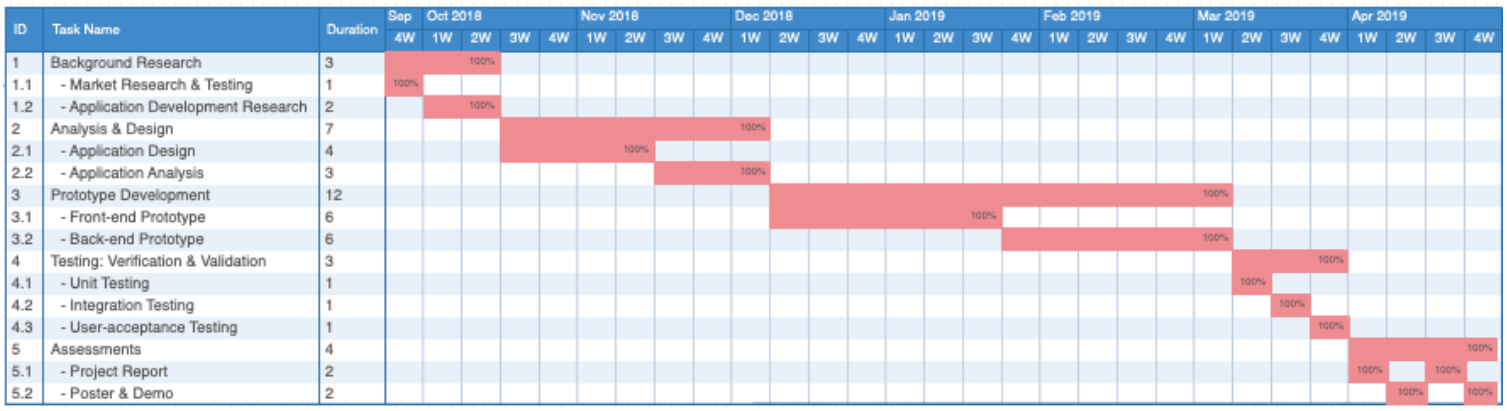
Following completing a prototype of the front and backend systems, test cases listed below will be implemented in order to judge the efficacy of the solution implemented.

- Unit tests
- Integration tests
- User acceptance testing

Given the results of testing, action will be taken in order to improve the application within the time constraints of this project, at which point an evaluation will be conducted to discuss how well the final application created meets the initial project aims.

1.4 Work Plan (Gantt Chart)

In order to incorporate project management tools in the project, a Gantt chart is created which displays a timeline of how the project will run. This allows the project to be completed with greater efficiency, with improved cost and time outcomes [3]. The Gantt chart in Figure 1 details individual project tasks, their durations and the sequencing of these tasks. Furthermore this also provides the overall timeline of the project with the expected completion date.



ID	Task Name	Duration
1	Background Research	3
1.1	- Market Research & Testing	1
1.2	- Application Development Research	2
2	Analysis & Design	7
2.1	- Application Design	4
2.2	- Application Analysis	3
3	Prototype Development	12
3.1	- Front-end Prototype	6
3.2	- Back-end Prototype	6
4	Testing: Verification & Validation	3
4.1	- Unit Testing	1
4.2	- Integration Testing	1
4.3	- User-acceptance Testing	1
5	Assessments	4
5.1	- Project Report	2
5.2	- Poster & Demo	2

Figure 1 - Work Plan (Gantt Chart)

1.5 Organisation of the Report

In this report, the way in which an individual’s daily calorie needs, along with calorie expenditure during a running activity will be analysed in order to create a comprehensive calorie tracking system. Cross-platform frameworks will then be evaluated in order to choose the most appropriate model for developing a fitness tracking application. The areas considered will involve availability, cost-effectiveness and code re-usability in order to deliver an application which is both built for scale and provides close to native performance. The top fitness tracking applications available in the market will also be analysed in order to create a list of features which describe the essence of what makes a fitness tracking application. Following research, an appropriate design will be created to allow the implementation of a product to form following best design practices for cross-platform development. Following each stage of the software design process, including implementation and testing, an evaluation will take place in order to reflect on the work completed and how this contributes to solution approach. The social, legal, health & safety and ethical issues present in creating a fitness tracking application that deals with location tracking will be discussed in order to ensure that the product created remains compliant with current laws. The created product will also be evaluated against the fitness tracking applications researched at the beginning of this project in order to determine how the developed application builds upon currently available systems. Furthermore the successes and limitations of the application developed will be discussed, considering how the final application meets the original project objectives and outputs, along with considering extra features that can be implemented in terms of future work.

2. LITERATURE REVIEW

2.1 Theoretical and Background Concepts of Fitness Tracking

Calculating and tracking calories is an important facet of weight loss and maintaining an individual's health. This is a fundamental measure used in fitness tracking applications in order to quantify the impact of exercise. The amount of energy in an item of food or drink is measured in calories. [4] Ingels et al. (2017) found that consistent calorie tracking over a period of 12 months leads to significant weight loss and overall health improvement [5]. Furthermore, Gradari et al. (2016) described a "hormetic-like biphasic dose-response that exercise protocols induce, highlighting that exercise can make you smarter, happier and cause your brain to generate more neural connections depending on the intensity of the training programme" [6].

In order to calculate a user's recommended daily calorie needs, their gender, age, height, weight and physical activity level must first be measured [7]. A user's daily calorie needs are then calculated by using Total Energy Expenditure (TEE) [8] which provides two different equations for men and women, given in eq. (1) and eq. (2). Weight is measured in kilograms and height is measured in metres. The value for Physical Activity Intensity (PA) is given depending on the physical activity an individual undertakes on a daily basis, provided in Table 1.

Equation (1) - TEE (Men)

$$TEE(Men) = 864 - 9.72 * Age + PA * (14.2 * Weight + 503 * height)$$

Equation (2) - TEE (Women)

$$TEE(Women) = 387 - 7.31 * Age + PA * (10.9 * Weight + 660.8 * height)$$

Table 1 - Physical Activity Intensity Values For Men & Women

Physical Activity Intensity (PA)	Men	Women
Sedentary	1.0	1.0
Low Active	1.12	1.14
Active	1.27	1.27
Very Active	1.54	1.45

An individual is classified into the given PA category considering the following conditions:

- Sedentary: An individual only has light physical activity associated with typical day-to-day life.

- Low Active: An individual adds 30 minutes per day of walking to their exercise routine at the speed of 4 miles per hour.
- Active: An individual adds an hour of moderate daily exercise to their routine.
- Very Active: An individual undergoes vigorous daily exercise.

Conversely, in order to calculate the amount of energy expenditure (calories/minute) an individual experiences, MET (Metabolic Equivalent of a task) tables from the Compendium of Physical Activities must be considered [9]. The MET is the objective measure of the ratio of the rate at which a person expends energy, relative to the mass of that person, while performing some specific physical activity compared to a reference, set by convention at 3.5ml of oxygen per kilogram per minute, which is roughly equivalent to the energy expended while sitting quietly. The MET tables provide values for running at different speeds (Table 2), which when combined with eq. (3), the number of calories burned during a running exercise can be calculated.

Equation (3) - Energy Expenditure

$$EnergyExpenditure = 0.0175 * MET * Weight(kg)$$

Table 2 - MET Values for Running

MET Value	Speed (mph)
8.0	5
9.0	5.2
10.0	6
11.0	6.7
11.5	7
12.5	7.5
13.5	8
14.0	8.6
15.0	9
16.0	10
18.0	10.9
15.0	Running stairs

2.2 Market Research of Existing Applications

In order to begin designing a new fitness tracking application, it is essential to understand what technologies are currently available in the market and how these can potentially be improved upon. For the purposes of this project, the applications considered when conducting market research and analysis were the cross-referenced top five results which appeared when searching for the term "fitness tracker" on the Apple iOS App Store and Google Play Store on the 24th September 2018. These results yielded the following fitness tracking applications:

- Strava
- MyFitnessPal
- Pacer Pedometer & Step Tracker (MyPacer)
- Runtastic Running & Fitness
- Endomondo Sports Tracker

Through analysing each of these applications by comparing and contrasting their strengths and weaknesses, a cohesive understanding of the features currently available in the fitness tracking market can be developed.

Strava

Strava [10] is a running and cycling tracking application with an emphasis on social media integration. This application allows the user to create and log in to their user account on the platform using either Facebook or Google+. The application uses the phone's location data in combination with Google Maps in order to provide GPS tracking of the user. This plots their activity on a map and provides the user with different statistics about the activity undertaken. On a brief overview, these statistics include distance, time and pace while running. Users are able to like and comment on other user's fitness sessions and these can be shared to Facebook & Twitter. More in-depth details are also provided regarding the activity undertaken, such as speed, max speed, and kCal burned. Graphs are also formed from this information and displayed to the user.

Once the user has started up the application and logged in they will see the home screen. There are 5 tabs at the bottom of the screen in order to allow the user to navigate through the application. The home screen displays a social feed to the user and the explore tab which allows the user to search for challenges and clubs. The record tab allows the user to record their activity by tracking their location data and listing this information on the view of a map. The profile tab allows the user to set their own personal goals, view their activities, track statistics and will allow the user to enter information regarding their gear. This will provide distance metrics on how long the user has used different equipment. The profile also has features such as routes, which allow the user to create and undertake different routes utilising the application.

When pressing the record tab, the user will be asked to select a sport. The user can select the activity they will be doing and a GPS signal will be acquired for the user. The recording screen of the application tracks the activity throughout its duration. The user is able to press the location button to see their route on a map, or the user can press the stop button which will stop the timer and display the user with a new screen. At this point the activity can either be resumed or finished.

Resuming the application takes the user back to recording, pressing the finish button then takes the user to a new screen, where the user can save their activity. When saving a user's activity, the user can record further detailed information about their workout, such as writing notes. While in an ongoing activity, the user can select the settings menu. This allows the user to prevent the phone from auto locking via 'Display Settings', allowing the application to run continuously while the user is undergoing an activity. The user can also set audio cues to provide the user with audio feedback at half mile or mile intervals. Lastly there is an auto-pause feature, which stops the timers running if the application detects that the user is at a traffic light or has stopped moving for a period of time.

In terms of the technical aspects of the application, Strava is built using the React Native framework using the ES6 variant of JavaScript. This incorporates using the Google Maps API with a React Native package in order to provide the application with location data on the user. The programming languages used in development include Ruby, Ruby-on-Rails, Scala and JavaScript/ CoffeeScript.

Overall Strava is a useful fitness tracking application, which allows users to track and analyse their own data. The most noticeable feature of Strava compared to other fitness tracking applications is the ability to create and follow routes, as well as competing against other users on a leaderboard for a given route. Although this is the most innovative feature of the application, research conducted by Williams (2013) [11] highlights that the majority of Strava users care most about tracking their activity data, rather than competing with other users.

MyFitnessPal

MyFitnessPal is a mobile application developed by Under Armour [12]. In contrast to Strava, MyFitnessPal doesn't focus on location tracking by activity and instead focuses on broader calorie tracking, including tracking the food the user consumes as well as the calories burned from different means of exercise.

When the user signs up to the application, the user is prompted to input various personal information, such as their height, weight and a fitness goal they are looking to achieve, such as gaining or losing a certain amount of weight per week. Given the requirements the user specifies, upon creating an account the application will inform the user of their daily net calorie goal for consumption. This provides the user with useful information which can help a user to control and manage their calorie consumption, therefore providing a means of improving an individual's fitness.

Considering MyFitnessPal does not offer a location tracking feature, this application is less entertaining and interactive to use than Strava. MyFitnessPal provides a form for the user to fill out information regarding their recent activity, where the user must input how long the activity lasted, the start time and the calories burned in the activity. This inherently presents a problem for the user, since they have to calculate calories burned for a given activity themselves. This could easily be automated if the number of exercises offered was reduced. For example, if the application focused on a single activity such as running, the number of calories burned could be provided to the user upon completing an activity, rather having the user calculate this information.

The application comprises of 15 individual screens for account creation, in which each screen gathers information about the user's lifestyle in order to provide them with accurate information to achieve a fitness goal. The application is available on both iOS and Android platforms, but is not developed with a cross-platform architecture and instead consists of two separate code bases for each native platform.

Overall MyFitnessPal seems to be a good tracking application for calorie intake, however is somewhat lacking in terms of providing the user with an entertaining experience and calculating the calories burned for a given activity. The main feature of this application is the ability to track the calories a user consumes, by allowing the user to scan barcodes of food products. Upon scanning a product, the number of calories that this product contains is presented to the user and the application tracks this data against their daily calorie intake goal. This presents a useful fitness tracking application, however this could be further improved upon by calculating the amount of a calories a user burns in an activity. B. Laing et al. (2015) [13] conducted a study evaluating the impact of MyFitnessPal on weight loss. This suggests that the application may be useful for individuals who are proactive in self-monitoring calories, however for the average overweight primary care patient there was no significant weight change. One noticeable element of this study reports that user log ins to the application dropped significantly after one month, therefore highlighting the importance of developing an entertaining an engaging application in order to encourage the user the monitor their fitness.

Pacer Pedometer & Step Tracker (MyPacer)

MyPacer is another top fitness tracking application available on both iOS and Android [14]. MyPacer provides features for the user to either create an account or log in to the application using social network integration. When creating an account with MyPacer, the application also requests permissions to access the a user’s motion and fitness activity. MyPacer utilises a motion co-processor in order to count steps. Therefore upon account creation the user’s steps activity statistics are automatically populated for the previous week. This provides the user with instant feedback in terms of analytics and data regarding their recent activity over the last week.

MyPacer does not have the same options as MyFitnessPal when it comes to providing the user with information regarding calorie intake. MyPacer is more concerned with analysing and tracking the user’s steps through motion detection, however the application also has a GPS tracking feature for when the user is undertaking a walking, jogging or running exercise. This works similar to Strava, providing the user with the time, distance travelled, pace, steps and during the course of the activity. MyPacer also calculates the effective calories burned at the end of an exercise, which MyFitnessPal did not consider.

Whilst MyPacer tracks exercise data interactively, there is another feature which also allows the user to input walking, jogging and running activities that they have completed without using the application. If the user provides the distance and time taken for the exercise in a form, MyPacer will automatically calculate the calories that the user has burned over the exercise period. This application also allows the user to track their weight, blood pressure and steps through manual input for further data analysis.

MyPacer implements social network integration effectively, allowing the user to be able to visualise the exercises that their friends have completed in a social feed. A user can also join groups in order to have competitions for certain activities inside a social circle. There is also an explore feature which allows the user to find upcoming fitness events near their location, encouraging users to undertake fitness activities in a group setting.

Overall MyPacer is a useful fitness tracking application, offering different features to both Strava and MyFitnessPal. The main innovative feature of this application is regarding how the users analytics data populates when an account is created, given that MyPacer connects to the health and motion data on a user's mobile phone. This provides the user with instant feedback on the user's previous activity allowing them to see an immediate improvement in performance when using the application. This application could be improved upon by providing the user with goals that they can set and work towards. Furthermore the user experience could be enhanced throughout the application by providing a more consistent style and design across different screens.

Runtastic Running & Fitness

Runtastic Running & Fitness is another top fitness application, developed in conjunction with the Adidas group [15]. This application is a running GPS tracker with over 90 million users and 8 million push messages sent across 18 languages per week, highlighting their significance as a global player in the fitness tracking technology market.

Runtastic provides a strong integration with wearable technology, initially providing integration support for their own wearable smart watch and fitness bands. Runtastic also supports a vast array of wearables from the the Apple Watch to Wear OS and Samsung's Gear series. Apple Watch and Wear users are also provided with a feature allowing a user to start a run utilising Siri and 'OK Google' voice commands. Running statistics are provided to the user directly from the watch screen, saving the user from looking at their mobile phone whilst running [16].

This application provides GPS tracking while the user completes a running activity in the same way as both Strava and MyPacer. However there are differences in the way the application performs which improves user experience. For example when starting a Runtastic activity, a countdown is provided to the user before the activity tracking starts. This allows the user to begin their activity at a more controlled pace. Runtastic provides goal tracking features for individual workouts or for more long term goals that a user may wish to work towards over the course of a year. The most innovative feature that Runtastic provides is the concept of LIVE Tracking, which allows the user to share their running location data and statistics to their friends live on social media. Runtastic offers further premium features including personalised work out plans, however these will not be considered in order to present a more equal comparison between the applications chosen.

Overall Runtastic is an effective application providing users with a means to track and analyse their running activity data with an entertaining platform. Similarly to Strava, audio settings can be altered in order to provide the user with as much information as wanted at each checkpoint, such as total duration, pace, speed, calories burned and heart rate (if a heart rate monitor is connected) [17]. An auto-pause feature is also provided that stops the activity tracking when a user is not moving, therefore providing more accurate activity statistics. Unlike MyPacer, previous activity data is not automatically populated through the mobile's health and motion data. However other useful features are considered such as goal tracking and Runtastic provides a strong integration with wearable technology.

Endomondo Sports Tracker

The Endomondo Sports Tracker is another GPS fitness tracking application considered while analysing the features of top fitness tracking applications. In February 2015, Endomondo along with MyFitnessPal, were acquired by Under Armour Inc. at which time, Endomondo had 20 million users [18]. Similarly to the other fitness tracking applications considered so far, Endomondo can track numerous fitness attributes such as running routes, distance, duration and calories [19].

Endomondo offers an achievement system in order to keep the user more engaged and provide further means of entertainment alongside fitness tracking capabilities. Once a user completes an activity, if they have set a new personal best in terms of running a certain distance or burning a given amount of calories during the activity, the user is presented with an achievement screen and this information is recorded. This rewards the user with positive feedback for improving their fitness and for creating and completing their own challenges.

Endomondo also attempts to facilitate user engagement with the application by sending the user push notifications as a reminder to work towards any upcoming goals or training sessions that they may have scheduled. This application supports many devices, including iPhone 3GS and later, Android 2.2 and later, Garmin Forerunner, Edge, Soleus and Timex Run Trainer 2.0. Similarly to Runtastic and Strava, Endomondo also provides audio coaching, gear tracking and can be connected to a heart rate monitor.

The other main concept of this application is that it allows you to join groups with your friends in order to compete with them. This happens on a weekly basis and can act as a form of encouragement in order to ensure that you are always at the top of the leaderboard in your social circle, therefore encouraging fitness and an overall more healthy lifestyle for users of the application. Overall this application provides an effective fitness tracking application in which a multitude of features are focused on user engagement.

Feature Analysis

The applications analysed highlighted a number of useful features that mobile fitness tracking software can provide. Given the results of the analysis, a comprehensive list considering the main features of fitness tracking applications was comprised, which are as follows:

- User authentication system
- Recommended calorie intake calculations
- Calories burned during activity calculations
- Goal tracking
- Activity analytics panel
- Achievements
- Audio cues

The features listed above will be included in the development of the application, building a complete system that both provides a recommendation for a user's daily calorie intake, as well as providing a means of tracking the calories burned when a user undertakes a running activity.

2.3 Cross-platform Frameworks Analysis

The classic approach to developing a mobile application which can target numerous devices, would be to produce two different applications. One written in Java for Android and the other in Swift or Objective-C for iOS. This path is both time-consuming and expensive because the same code must be written twice for each operating system respectively. On the other hand, it ensures the highest code consistency with the original application look and feel.

Cross-platform development, in contrast, allows engineers to write the code once and apply parts of it across all platforms. Usually, this happens at the cost of performance and application behaviour. Cross-platform engineering communities strive to mitigate these disadvantages by continuously introducing new approaches and tools. Cross platform development is a beneficial approach to application development due to its cost-effectiveness [20]. This enables investing just once in a single team, utilising only one technology stack that developers can use for a variety of engineering tasks. This form of development also creates re-usable code. Up to 90% of the codebase can be reused from one platform to the other, instead of designing the same functionality in another language. It is also far easier to maintain and deploy changes as there is no need to maintain applications on each platform separately.

In order to develop a cross-platform application, there are three main frameworks that can be used in the development process. These are as follows:

- Xamarin - A Microsoft-supported framework for cross-platform mobile application development that uses C# and native libraries wrapped in the .NET layer.
- React Native - A framework which allows for building close-to-native mobile apps using JavaScript and React.JS.
- Ionic - A framework that aims at developing hybrid apps using HTML5 and Angular.

Through analysing the strengths and weaknesses of the different cross-platform frameworks available, the most suitable choice can be selected for the development of a fitness tracking application.

React Native

The React Native framework has a number of advantages and is one of the fastest and most efficient environments for mobile application development. It’s analogous to ReactJS and uses JavaScript. React Native renders some code components with native APIs, unlike other cross-platform frameworks which may render code via a WebView. Utilising a WebView greatly decreases performance, however React Native communicates with targeted components for iOS or Android and renders code to native APIs directly and independently. Doing so, React uses a separate thread from UI, which also increases the performance score of the application. To create a React Native app across platforms, developers also do not need to know the language of the native platform. They must only be proficient in JavaScript and familiar with the React syntax. Furthermore one of the main benefits of utilising React Native is that the framework is open source and completely free to use.

React Native still hosts numerous disadvantages and problems that arise when developing with the framework. Firstly, the React Native community is young and has only been around since

March 2015 [21]. Due to this, the documentation available is poor, especially for integration with additional tools. React Native also has issues using third-party components. React Native has a number of native modules for iOS and Android out-of-the-box, but the number of third-party components is still limited. Due to this, developers are required to integrate community-built modules, which may not be supported by further releases of the framework. This causes a disadvantage to utilising the framework as React Native may not offer a complete spectrum of possible features that developers may want to implement in their applications. Furthermore React Native tends to have instability, compatibility issues and errors. There are a number of problems that engineers face when working with React Native, including hot-reloading failures, incompatibilities between community libraries and different versions of the React native emulator, problems with navigation and the need to frequently re-install packages due to version control [22].

Xamarin

Xamarin is a software company founded back in 2011. It provides a developer with tools that can help them in building cross-platform mobile applications. The applications can have all the native features and also share the common codebase at the same time. Xamarin tools are available to download with visual studio and most of the common code is written in C#. Xamarin also supports wearable devices with the apple watch & android wear. This is based on a .NET framework. Xamarin uses C# and native libraries wrapped in the .Net layer for cross platform mobile application development [23].

Xamarin's main advantage is that it's performance is close to native. Unlike traditional hybrid solutions based on web technologies, a cross-platform application built with Xamarin can still be classified as native. The performance metrics are comparable to Java for Android or Swift for iOS. Xamarin performance is also constantly being improved in order to attempt to match the native standard. Xamarin allows a developer to create flawless experiences using platform-specific UI elements. Simple cross-platform apps for Android or Windows can be built using Xamarin.Forms tools, which converts application UI components into the platform specific interface elements at runtime, therefore significantly increasing the speed of cross-platform application development. Xamarin also boasts to have simplified maintenance, due to Xamarin including its own IDE, distribution and analytics platform. Therefore users do not need to invest in additional tools or integrate third-parts applications to build, test and deploy Xamarin applications.

Xamarin presents the problem of having limited access to open source libraries. Native development makes extensive use of open source technology, yet with Xamarin, you have to use only the components provided by the platform and some .NET open source resources. This choice is not as rich as it is for android and iOS mobile application development, however Xamarin components do provide many useful features. Xamarin also presents a problem that requires the developer to have some knowledge of the native languages required. When building a mobile application with Xamarin, developers will still need to write a platform-specific layer of code and therefore a basic knowledge of platform-specific code (Java for Android/Swift for iOS) is still required. Xamarin applications are also typically larger than native applications and in order to develop with Xamarin commercially, a user would be required to purchase a commercial Visual Studio licence.

Ionic

Ionic is a framework based on Apache Cordova and Angular, which enables developers to build fully functional and advanced mobile applications with the usage of web technologies, such as Angular, HTML and CSS [24]. Using the Ionic framework, a single code base written in Angular, HTML and Saas will transform into a mobile application, utilising Apache Cordova to compile components generating a native application with access to all functionalities specific to a phone. The cross-platform applications generated are not purely native and not entirely web-based. The layout is achieved utilising web views, but the application still has access to the native device's APIs. Therefore Ionic applications can be displayed natively, as well as in the browser as a progressive web application.

Ionic's advantages, like other cross-platform frameworks, provide a cost-efficient way of developing mobile applications for a range of platforms. Ionic also provides a developer with fully equipped documentation, which are accessible, very detailed and open source. Ionic also provides a wide range of components and plugins. A library offers components that act and look like native elements of all the supported platforms, however there is limited native functionality.

Ionic's disadvantages firstly include performance issues. Native mobile application's performance is significantly better than Ionic applications, therefore when developing an application with heavy performance or graphical needs, Ionic is not a suitable choice. Further the Ionic framework is lacking in terms of security and does not provide a robust enough system to be utilised with a financial application for example. Ionic requires advanced knowledge of multiple technologies in order to implement an application successfully and although the basic version is free, users will be required to purchase a licence in order to utilise the advanced features of the software. Finally Ionic is mobile targeted, therefore if developers are aiming to create a web application in conjunction with a mobile application, the web applications will have stylisation issues and therefore other frameworks would be more suited to implementing this functionality.

Framework Conclusion

Given the analysis undertaken when looking at the different frameworks available for building cross-platform mobile applications, Table 3 was created in order to highlight the significant features of each Framework. As the fitness tracking application will need to render the user's location on a map in real-time whilst tracking statistics, performance is a major issue which must be considered. Therefore Ionic cannot be utilised for this project considering the performance issues. Xamarin is also considered for this project, considering its performance is greater than Ionic. However in order to develop with Xamarin, Visual Studio IDE is a requirement in which a commercial licence would be needed. Therefore in order to create an application which works cross-platform, has close to native performance and is the most cost effective, React Native is the best choice. Danielsonn (2016) conducted a performance evaluation regarding a comparison between React Native applications and native android applications [25]. The result is very positive for React Native, given that some users could not tell the two applications apart and nearly all users did not have any issues with React Native applications. The performance evaluation measured GPU frequency, CPU load, memory usage and power consumption. Nearly all measurements displayed a performance advantage for the Android application, however the difference in results were not significant. The overall experience is that React Native is a very useful framework that can simplify the development process for mobile applications to a high degree. As long as the application itself is not too complex, the development is uncomplicated and one is able to create an application in a short time which can be compiled on both Android and iOS with a minimal performance decrease in comparison to natively built products.

Table 3 - Comparison of Cross-Platform Frameworks

	Xamarin	**Ionic**	**React Native**
Code	C#	HTML, CSS, JavaScript	JavaScript+, Java, Objective-C, Swift
UI Rendering	Native UI Controllers	HTML, CSS	Native UI Controllers
Price	Open Source	Open Source	Open Source
Additional Costs	Visual Studio IDE for commercial Use ($539-2999)	Ionic Pro ($29-199)	N/A
Cross-Platform Portability	Xamarin. iOS/Android. Xamarin. Forms Business Logic, Data access, Network communication — Up to 96 percent of source code reuse	Up to 98 percent of source code reuse	Adapting code to each platform
Performance	Close to native. Moderate-low	Moderate-low	Close to native
Use Cases	All apps. Simple apps, Corporate apps	Simple apps, corporate apps	All apps

3. DESIGN OF RUNTRACKER

3.1 Application Features

The main problem found when researching different fitness tracking applications is that no application provided a complete calorie tracking system. MyFitnessPal provides a useful system in calculating the recommended calories a user should intake each day, however provides no functionality to calculate the calories burned. MyPacer, Runtastic, Strava and Endomondo all provide useful features for tracking the calories a user burns whilst undertaking an activity, however these applications do not recommend the number of calories a user should consume in conjunction with this information. Therefore a key design feature of the application to be created will focus on providing a complete calorie tracking system for each user. As a result of analysing the top cross-platform fitness applications currently available in the market, a list of the important features required to create a fitness tracking application was created which will be discussed.

3.2 User Authentication

The first of these features comprises developing a user authentication system. When a user registers for an account, they will create both a username and password which will allow them to access their account. User authentication is important in order to manage the connection between the user and the application, to verify user's identities and to either approve (or decline) authentication so the application can authorise the user. User authentication will be vital for this project, in order to ensure that unauthorised users cannot access sensitive information, thereby providing a measure of security. Furthermore authentication will ensure that user A will only have access to the information they need and will not be able to see the sensitive information of user B [26]. With a user authentication system implemented, each user can be provided with a UID (unique user identification number), which will allow them to both store and retrieve information relevant to the user from a backend database.

When the user starts the application for the first time, the user should be initially presented with a screen to log into the application. From this initial screen, the user will either be able to type their credentials into a form in order to be able to log in, or the user will be able to navigate to another screen to sign up. If a user selects the log in button on the initial screen, their credentials will be attempted to be verified with an authentication service. If the authentication is successful, the user will log into the application, transitioning the user to their profile page. If the authentication is unsuccessful, the user will be informed that their credentials are incorrect and asked to retry.

3.3 Recommended Daily Calorie Intake

In order for a user to accurately track and manage their calories, the recommended amount of calories that a user should be consuming each day will be provided to the user. During the user's sign up process, the user will have to enter personal information for the application to provide an accurate metric for the user regarding their TEE. This includes gathering information such as the user's date of birth (in order to calculate age), weight, height, goals and lifestyle. The goals of the user represent what they would like to achieve by utilising the fitness tracking application, whether this is losing weight, maintaining their weight or gaining weight. A user's lifestyle references the users current level of physical activity, as provided by the data above in Table 2. With all of this information, the application will be able to calculate the user's BMI and recommended daily calorie intake which provides the second feature. This information will be shown to the user at the end of the sign up process and can also be visualised any time by accessing the user's profile screen.

3.4 GPS Activity Tracking & Calories Burned

One of the main features of this application will be the record screen. The record screen will allow the user to begin a running activity, whilst displaying the user's position on a map utilising GPS. If the user presses the start button, the activity will begin and the timer will start incrementing. As the user runs, the phone will provide a stream of location coordinates and speed, which can be utilised to present the user with different metrics about their run so far, such as the average speed, the time taken, the distance travelled and the calories burned. At any point the user will be able to either reset the activity, which will clear all of the current data so the user can begin a new activity again. Furthermore the user will also be able to complete an activity, transferring the user to a final screen in which they will be able to comment on their activity experience by entering a note and saving the activity to the database. A mock-up of the record activity screen can be seen in Figure 2.

When the user presses the start button, the start button will change to display a pause button and the reset button will change to display a finish button. Pressing the pause button will stop the activity, displaying the user with the initial buttons seen in Figure 2. Pressing the finish button will transition the user to a new screen where they will be able to see further activity statistics and create a note to be saved with their activity.

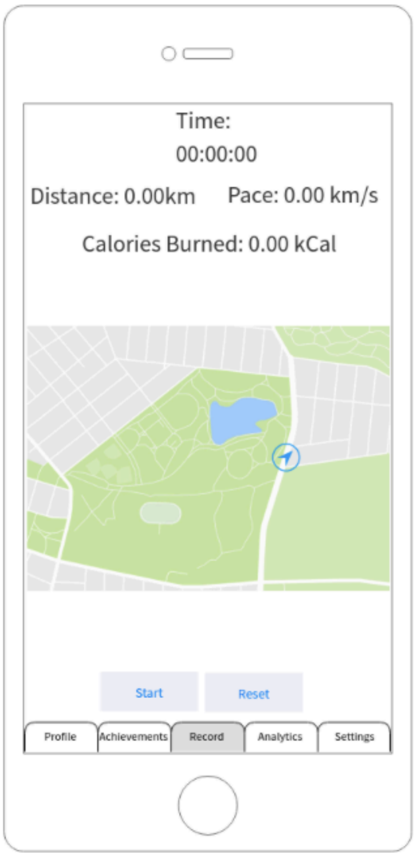


Figure 2 - Record Activity Screen Wireframe

Once the activity has started, the route coordinates will be saved, continually updated and drawn on the map as a line following the user's location. This will provide the application with the function of entertainment, allowing the user to see where they have run so far as well as displaying to the user updated statistics regarding the activity.

3.5 Goal Tracking

An important feature of the application is the ability to set a goal to work towards each week. Goal setting is crucial when considering fitness, as setting goals gives an individual both long-term vision and short-term motivation [27]. Through the practice of setting goals, users are rewarded for their achievements and guided towards a healthier lifestyle. On the last page of the sign up process, users will be recommended an amount of calories that they should aim to burn each week in order to meet their fitness aim (losing, maintaining or gaining weight). Users do not have to accept this recommendation and can choose their own goal if they wish. The goal and the user's progress towards their goal will be displayed to the user at all times on their profile page. The user's goal will be reset at the end of each week and each time a user completes an activity, the progress towards their goal will be updated.

3.6 Analytics Dashboard

Along with the features mentioned so far, another vital feature of this fitness tracking application is the means to track and analyse previous activities. This will be implemented by creating an analytics dashboard which can be accessed from the main tab bar navigator at the bottom of the screen. After each activity has been completed and saved, the activity statistics will be stored in the back-end database and the user will then be transitioned to the analytics screen. The analytics dashboard will fetch the activity statistics for the current week and display them to the user in a line graph format. The analytics that are displayed to the user will include distance, calories burned, speed and time, all sorted by date. Along with GPS tracking, goal setting, and calorie calculations, the analytics dashboard will provide the main function of tracking the user's performance as they attempt to meet their goal each week.

3.7 Achievements

The last feature that the application will include is an achievements screen, which will highlight to the user their all time personal best in the following categories:

- Highest calories burned in an activity
- Longest time spent running
- Furthest distance spent running
- Highest speed achieved while running
- Best 5km time
- Best 10km time
- Best 15km time
- Best 20km time
- Best Marathon time

Each time a user completes an activity, the user's previous activity statistics will be compared to the activity just completed in order to judge whether or not a new personal best has been set for the user. If the user has achieved a new personal best, they will be informed with a prompt in order to keep the user more engaged and provide a more entertaining and fun system.

3.8 Application Structure - Flowchart

Figure 3 details a flowchart regarding how the user would move through the screens of the application. When starting the application, the user can either choose to log in if they have already created an account, or they can choose to sign up. Electing to sign up will direct the user through numerous sign up screens, gathering the required information and permissions from the user in order to perform the relevant functionality such as location tracking, calculating calories burned/ recommended and calculating the user's BMI. If a user chooses to log into the application, they will initially be presented with their user profile page, in order to remind the user of their current goal and progress. Users will be able to navigate to the different areas of the application by using a tab bar navigator located at the bottom of the screen. Pressing on one of the tabs at any time will transition the user to relevant screen where different functionality is available. The five tabs available include the profile, achievements, record activity, analytics and settings.

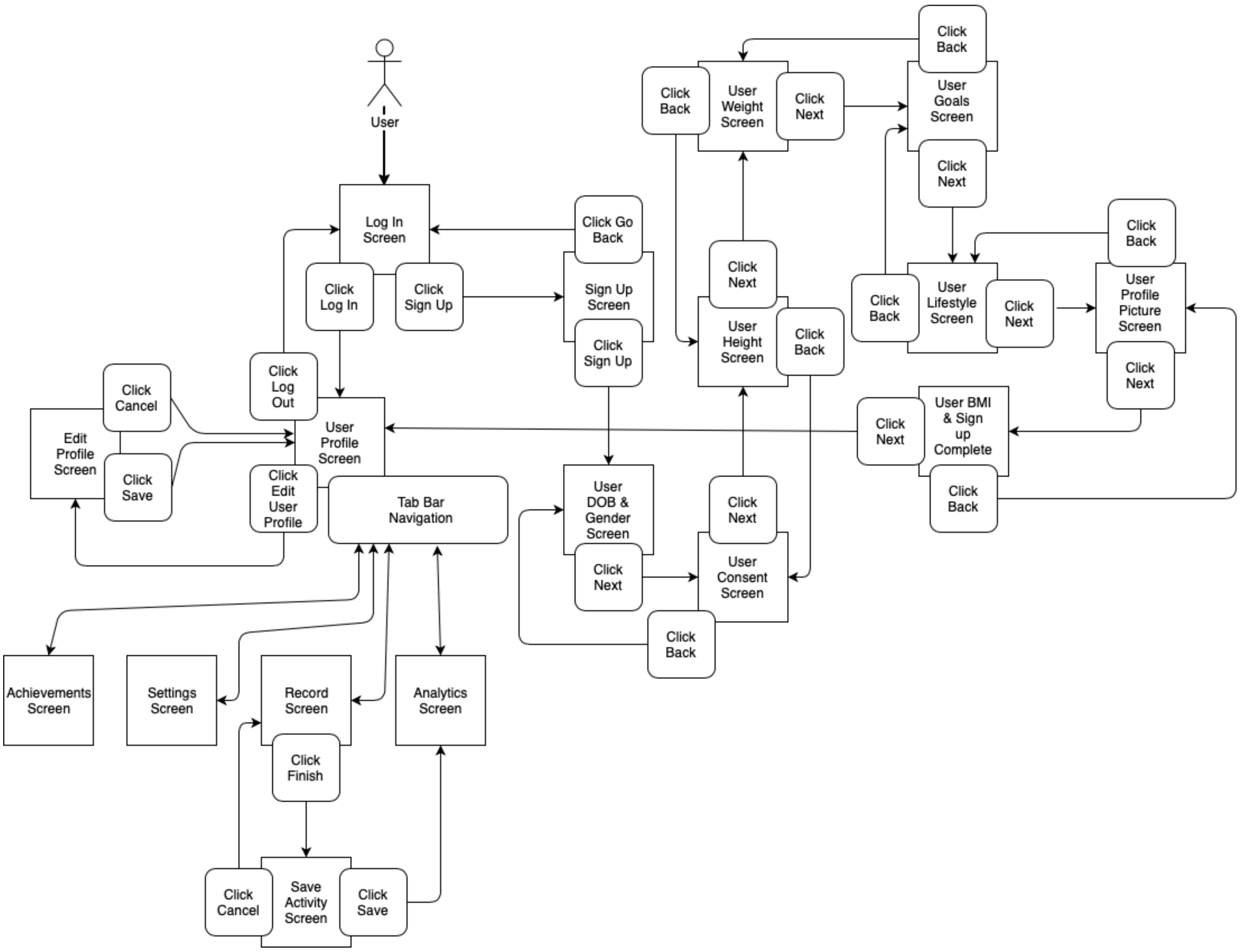


Figure 3 - Application Flowchart Design

3.9 Architecture Design

As a result of analysing the best cross-platform frameworks to utilise in development, React Native was chosen as providing the most cost-effective solution whilst also providing strong performance which is comparable to Android and iOS native applications. React Native will provide a framework for developing the mobile application using JavaScript and React, however other tools are also required in order to ensure that the application can function effectively.

3.10 Redux

The first of these required tools is Redux, which provides a predictable state container for JavaScript applications [28]. Redux organises data in the front-end of the application by using three principles.

1. The state of the application is stored in a single object tree.
2. The state is read-only.
3. The state can be changed by emitting an action which describes the desired change.

Using Redux within an application allows for uni-directional data flow. When Redux is used in combination with React.js, the data from the Redux store flows through props in container components. The container components then pass this data down to other React components. Users interact with React components, therefore whenever a component attempts to change the data within a store, it prepares an action object and dispatches the action object to the store. The store then takes this action and passes it to several reducers. A reducer is a JavaScript function, which takes the previous state of the store and the action object and then creates a new state for the store. The organisation of modules inside the Redux layer can be seen in Figure 4.

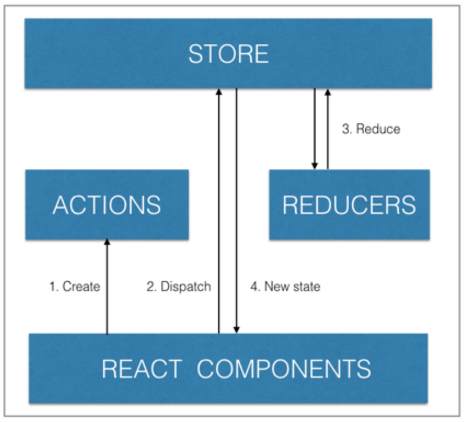


Figure 4 - Redux Modules

Actions are objects which each have a certain action type and each action object is dispatched to the store once created.

React applications often interact with data through an API. The API is responsible for either fetching data from the backend or updating data. In order to ensure asynchronous communication within the Redux layer, Redux utilises a middleware component called redux-thunk. This allows for react components to dispatch functions instead of action objects. The dispatched function interacts with the API and dispatches an action object when it then receives a response from the API. A second package, react-redux is also utilised to connect a React component to a Redux store. This is achieved through Providers and Connectors. The Provider component is responsible for providing the store to downstream components and the Connect function simply wraps a React component and provides the state of the store via props [29]. In order to communicate with the backend database, the user's identification number needs to be accessible in the majority of the screens present in the application. Therefore Redux is an essential component of the applications architecture, enabling the React components to subscribe to specific information held within the redux store in order to perform database actions.

3.11 Firebase

In order to store user's personal information, activity data, settings and the achievements gained for each user, a backend database must be implemented in conjunction with the front end of the application. The backend database chosen for this project is Firebase. Firebase is a Realtime cloud-hosted database from Google which is positioned as Backend-as-a-Service (BaaS). It is a NoSQL database and data is stored as key-value pairs [30].

Although Firebase stores data with key-value pairs, the data can be organised in a hierarchical tree. Each node in the tree is called a 'ref' and each ref has its own URL. A ref can be queried or updated like an SQL database. These queries or updates happen from the front end and Firebase provides a SDK which allows front-end applications to directly connect with the database. Firebase is a real-time database, given that a ref can emit events based on data changes at the server. It is essential to consider design best practices when implementing a large multi-user application at scale, therefore Firebase queries should be implemented within the Redux layer in order to simplify the code structure. Redux passes data to React components via props, which in turn allows writing functional or stateless components that just render the props, ensuring that React components are easier to maintain.

Data is stored in Firebase as a large JSON document. This is different to utilising a relational database management system with SQL and therefore NoSQL database design principles must be considered. Data should be stored in order to ensure that accessing objects is fast and efficient, therefore suggesting a strong importance in maintaining an effective database structure. If certain data will be unique to a particular object, it should be maintained as a static list for the object. Therefore activity data for each user should be implemented in this format. Redundant data should also be avoided as much as possible, therefore when considering groups of data, an index can be created in order to efficiently search this list [31].

When a user creates an account with the authentication service, they will initially be assigned a user identification key. This key will be essential for searching and updating the database records that are relevant to user. The database will consist of four main tables in order to carry out all of the functionality that the application requires. This includes implementing the tables 'Users', 'Activity', 'New_Activty' and 'Achievements'. The structure of each database table in JSON format is listed below.

3.12 Database Table Structure

Database Table - Users

A user profile will be located at the path, /users/uid. This table is utilised for storing all of the information relevant to the user's profile such as authentication data, consent, personal information and health related data. An example account will include the following fields:

```
{
        “users”: {
                “C9gJhfmZwfdoaavKzcpQ4DXHdJv2”: {
                        “profile”: {
                        “email”: “example@example.com",
                        “password”:“11c150eb6c1b776f39be60a0a5933a2a2f8c0a0ce766ed92fea5”,
                        “BMI”: {
                                “BMI”: 22.31,
                                “TEE”: 2361.77,
                                “weeklyCalorieGoal”: 1200
                                },
                        “PhysicalActivityIntensity”: {
                                “sedentary”: false,
                                “lowActive”: false,
                                “active”: true,
                                “veryActive”: false
                                },
                        “Consent”: {
                                “dataProtectionConsent”: true,
                                “dataTransferOutsideCountryConsent”: true,
                                “locationTrackingConsent”: true,
                                },
                        “Goal”: {
                                “gainWeight”: false,
                                “maintainWeight”: false,
                                “loseWeight”: true
                                },
                        “PersonalData”: {
                                “dateOfBirth”: 1994-03-05,
                                “age”: 25,
                                “firstName”: Bob,
                                “lastName”:  Tester,
                                “male”: true,
                                “height”: 165,
                                “weight”: 62,
                                }
                        }
                “0mJT5SbOkqYI4u5a1lDHw4xSjc82: {
                        “profile”: {
                        …
                }
}
```

Database Table - New_Activity

A new activity will be located at the path new_activity/uid. This database table is used for temporarily storing information regarding the user's current activity, before being stored in list of activity data. This also provides a back-up of current activity data, so if the user closes the application during an activity, the activity can be restarted. An example will include the following fields:

```
{
        "new_activity": {
                "C9gJhfmZwfdoaavKzcpQ4DXHdJv2": {
                        "new_activity": {
                                "distanceTravelled": 5032,
                                "kCal": 312,
                                "time": 9554,
                                "mapSnapshot": "Users/Example//Library/Developer/CoreSimulator/
Devices/7F3074A1-75D4-4A9A-9657-AF83A57674FB/data/Containers/Data/Application/
2BB8B668-93E2-4175-A21C-BF4C62BBF43C/Documents/
snapshot-1550852985.35653305053710937500.png",
                                "avgSpeed": 11.4
                                        }
                        }
        }
}
```

Database Table - Activities

The activities database table will store all of the past activity data that the user has completed and chosen to save and will be available at the path /activities/uid. Each activity will be stored with a unique key. This data will be fetched to present information to user in the analytics dashboard, providing the with tracking functionality. An example will include the following fields:

```
{
        "activities": {
                "C9gJhfmZwfdoaavKzcpQ4DXHdJv2": {
                        "activities": {
                                "LZKEyP9tUXfEji0Oxxh": {
                                "date": "16/2/2019,
                                "distanceTravelled": 3202,
                                "kCal": 214,
                                "time": 5598,
                                "avgSpeed": 5.44,
                                "notes": "This is an example of a user note",
                                "mapSnapshot": "Users/Example//Library/Developer/CoreSimulator/
Devices/7F3074A1-75D4-4A9A-9657-AF83A57674FB/data/Containers/Data/Application/
2BB8B668-93E2-4175-A21C-BF4C62BBF43C/Documents/
snapshot-1550852985.35653305053710937500.png”,
                                                }
                        “LZKFO3YdrzD4vS6hEXh”: {
                                “date”: “17/2/2019,
                                “distanceTravelled”: 4361,
                                …
                                }
                        }
                }
        }
}
```

Database Table - Achievements

The last database table that will need to be included in the application design is the table for achievements. This will be located at the path /achievements/uid and an example will include the following fields:

```
{
        “achievements”: {
                “C9gJhfmZwfdoaavKzcpQ4DXHdJv2”: {
                        “achievements”: {
                                “highestkCalBurned”: 378,
                                “longestTime”: 11392,
                                “furthestDistance”: 10123,
                                “highestSpeed”: 16.9
                                “best5K”: 6752,
                                “best10k”: 11392,
                                “best15k": 0,
                                “best20k”: 0,
                                “bestMarathon”: 0
                                }
                }
                “0mJT5SbOkqYI4u5a1lDHw4xSjc82”: {
                        “achievements”: {
                                “highestkCalBurned”: 312,
                                 …
                                }
                }
}
```

3.13 High-Level Logics

The tools that comprise the overall architecture of the application consist of utilising React Native as a Framework, Redux for a state management container and Firebase as a backend database. Figure 5 details the overall high-level architecture of the application, demonstrating how a user would interact with the software and how the individual components interact with each other. React Native components will provide either forms for the user to enter text, or this will also provide buttons and checkboxes for the user to select.

In the first example, a user may choose to view the terms and conditions of the application. This process will begin by the user clicking on a ‘Terms and conditions’ button inside a React Native component. The React Native component will then dispatch an action with the type ‘SHOW_MODAL’ and a value of terms_and_conditions. The action here is created by ‘actionCreators.js’. The dispatched action then updates the store. Once the store is updated, the terms_and_conditions_modal component runs its render method and sees that the value of visibleModal is now terms_and_conditions. This modal no longer returns null and renders the DOM, allowing the user to see the terms and conditions. Once the user reads this and clicks the close button, the click event dispatches an action with the type ‘SHOW_MODAL’ again with the value of null. This sets the visibleModal to null and therefore it is no longer displayed.

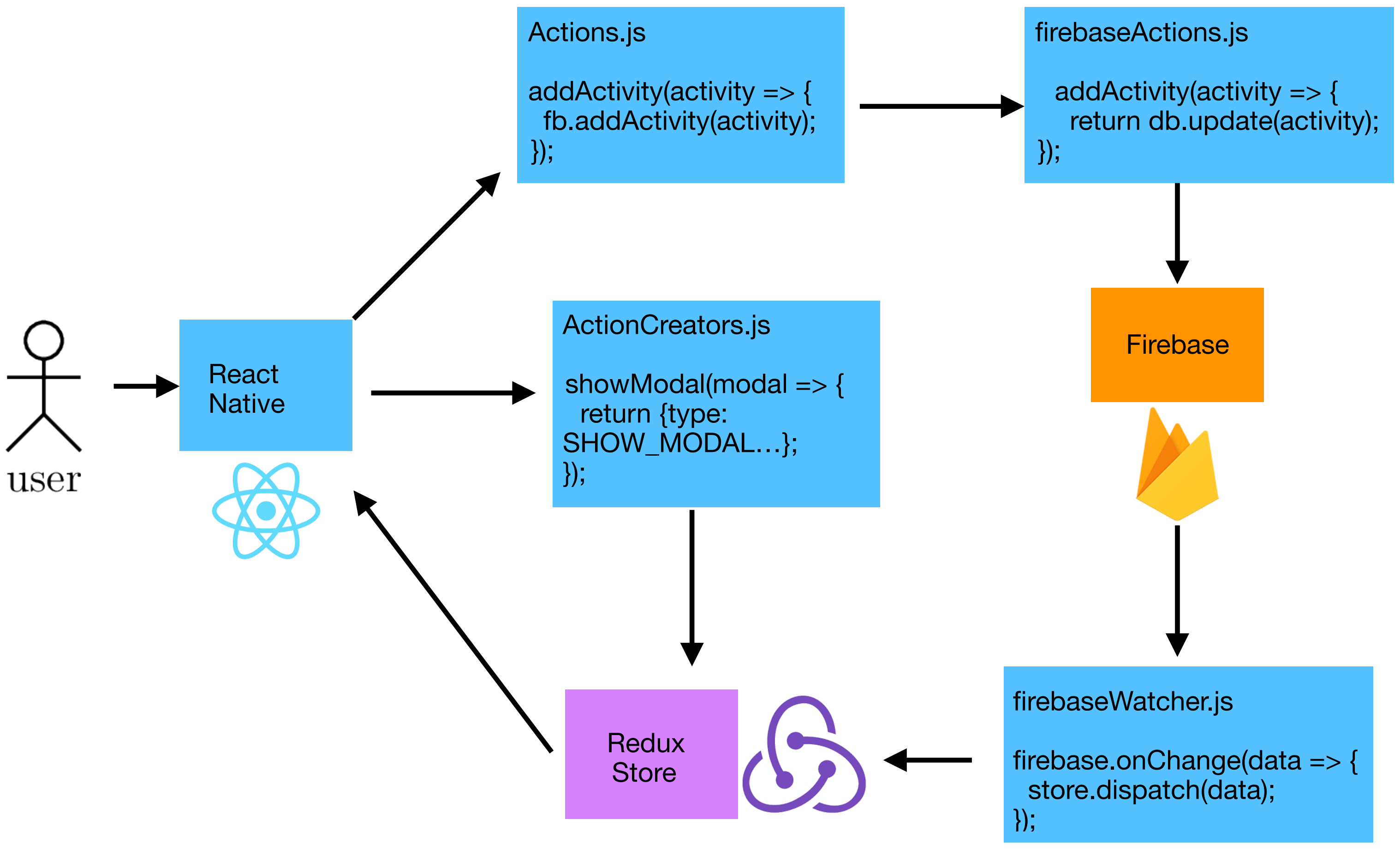


Figure 5 - High-Level Logics

In a second scenario, a user may wish to finish a running activity and save this activity to the database. The user will click on a button named ‘Save activity’ and the component will call a function called addActivity. The addActivity function then sends a message to Firebase, informing the database to be updated to add the activity to the list of previous activities. Firebase then updates the data and emits a change event. A listener for ‘activity added’ responds to the Firebase change event and sends the details of the new activity to the Redux store. The store then updates and emits a change. The React Native component then re-renders and the new activity is rendered to the screen in analytics tab.

3.14 Summary

In this chapter the overall design of the application was considered. This includes detailing the most important features of application in order to provide a fitness tracking application that functions as a complete calorie tracker, whilst also performing the functions of providing entertainment and allowing the user to compare their running performance over time. Initially, this involves considering a user authentication system in order to manage the connection between the user and the application. Furthermore the design details how the sign up process will collect relevant user information in order to be able to calculate the user’s recommended daily calorie intake, based on their height, weight, age, gender and physical activity intensity. The main feature of the application is the record activity screen, which utilises GPS and location tracking in order to track the user on a map as they complete a running activity, whilst calculating running statistics such as distance, time, calories burned and average speed. This feature is considered further in depth than other features as it will involve the most complex logic in the application. Furthermore the features of goal tracking, achievements and the way in which an analytics dashboard will included are also discussed in order to contribute to developing a fitness tracking application.

A flowchart detailing the way in which a user will move through the different screens of the application is created in order to ensure that the application structure is cohesive. The user experience in the application is key, in order to retain user engagement and create a successful product overall. When opening the application, the user will either be able to log in or sign up, in each which either of these options being a separate workflow that the user will progress through. When the user logs into the application, navigation will be controlled through a tab bar navigator placed at the bottom of the screen.

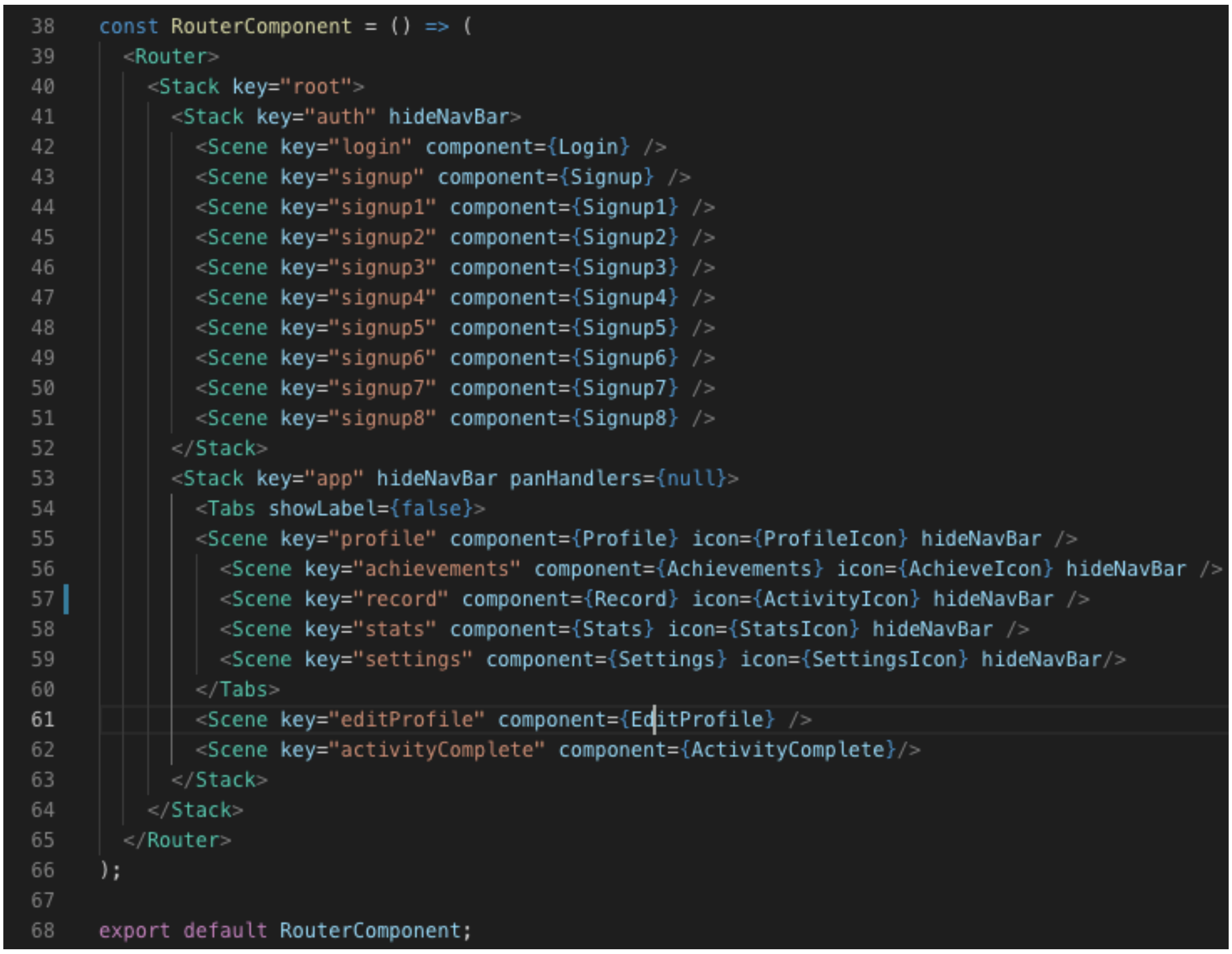
Redux and Firebase are also considered as tools in the design of the application, in order to be able to create a product which manages state effectively and is able to store information in a backend database. The database tables that are required for the purposes of this project are detailed in advance in order to ensure that the implementation of the product will result in efficient database searching and not result in redundant data. Finally the high-level design of the application is considered in order to describe how each of the tools utilised (React Native, Firebase and Redux) will work in conjunction with each other in order to be able to search and update information stored in database through a process for abstracting database management code to actions performed in the Redux layer.

4. IMPLEMENTATION OF RUNTRACKER

4.1 Routes

The React Native framework does not inherently provide a means to navigate through different screens of the application, therefore by utilising the flowchart created in the design section, the first aspect of the application to be implemented was the navigation method. The navigation tool chosen was ‘react-native-router-flux’ [32], which allows users to define all routes in one central place, as well as navigating and communicating between different screens.

In Figure 6, a router component is defined which details two stacks, one for handling user authentication (including login and sign up components) and one for handling the main components of the application once the user has signed in. The icons utilised throughout the application and in the RouterComponent are from ‘react-native-vector-icons’[33].



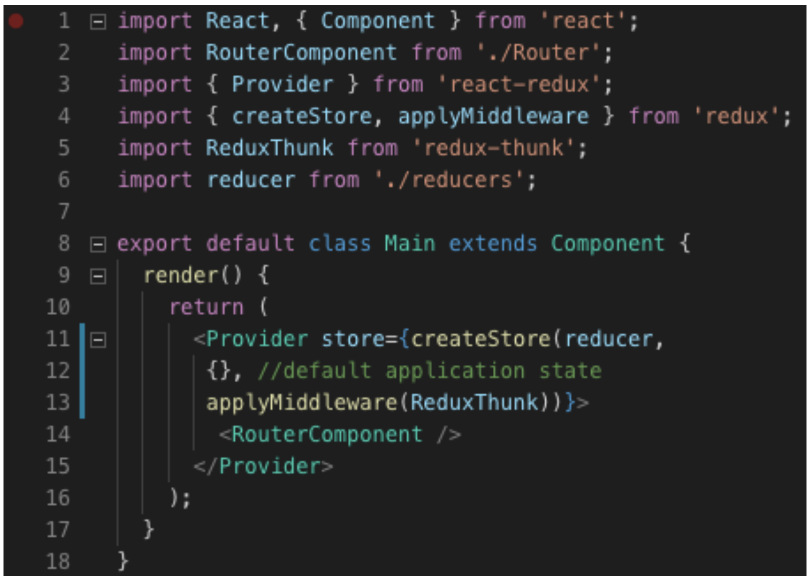
```
const RouterComponent = () => (
  <Router>
    <Stack key="root">
      <Stack key="auth" hideNavBar>
        <Scene key="login" component={Login} />
        <Scene key="signup" component={Signup} />
        <Scene key="signup1" component={Signup1} />
        <Scene key="signup2" component={Signup2} />
        <Scene key="signup3" component={Signup3} />
        <Scene key="signup4" component={Signup4} />
        <Scene key="signup5" component={Signup5} />
        <Scene key="signup6" component={Signup6} />
        <Scene key="signup7" component={Signup7} />
        <Scene key="signup8" component={Signup8} />
      </Stack>
      <Stack key="app" hideNavBar panHandlers={null}>
        <Tabs showLabel={false}>
        <Scene key="profile" component={Profile} icon={ProfileIcon} hideNavBar />
          <Scene key="achievements" component={Achievements} icon={AchieveIcon} hideNavBar />
          <Scene key="record" component={Record} icon={ActivityIcon} hideNavBar />
          <Scene key="stats" component={Stats} icon={StatsIcon} hideNavBar />
          <Scene key="settings" component={Settings} icon={SettingsIcon} hideNavBar/>
        </Tabs>
        <Scene key="editProfile" component={EditProfile} />
        <Scene key="activityComplete" component={ActivityComplete}/>
      </Stack>
    </Stack>
  </Router>
);

export default RouterComponent;
```

Figure 6 - Application Routes

With the routes of the application implemented, navigation between different screens is achieved by utilising the ‘react-native-router-flux’ API. For example, when the user starts the application the initial root stack will be called which takes the user to the login page. If a successful log in occurs, ‘Actions.app’ will be called which will transition the user to the ‘app’ stack which hold the React components for the main application features, therefore transitioning the user to the correct screen.

4.2 Redux

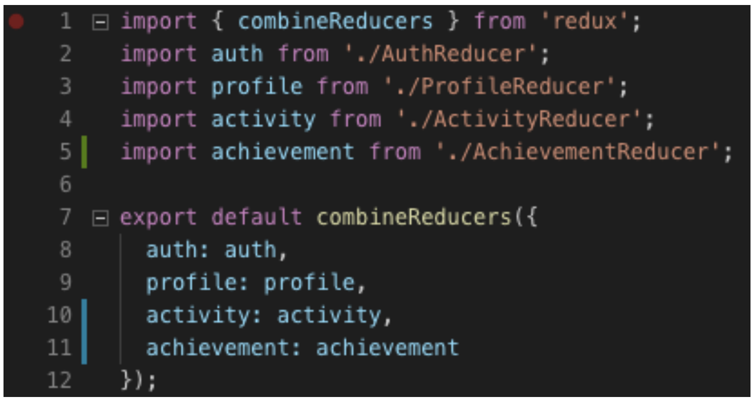
Given the application design requires the implementation of Redux, an instance of the redux store must be created and provided to the application routes in order for the application to have access to the store at all times. Therefore a provider tag is created from the React-Redux library. This provider tag is then rendered by passing in the store as a prop, so that any child component to this provider tag will have access to this store through the context system inside of React. In the case of this application, the child component of the provider tag is the application routes as seen in Figure 7, which will render all of the different screens of the application. Therefore this provides all of the screens of the application access to the Redux store, if they subscribe to the store using the connect helper from the React-Redux library.



```
import React, { Component } from 'react';
import RouterComponent from './Router';
import { Provider } from 'react-redux';
import { createStore, applyMiddleware } from 'redux';
import ReduxThunk from 'redux-thunk';
import reducer from './reducers';

export default class Main extends Component {
  render() {
    return (
      <Provider store={createStore(reducer,
       {}, //default application state
       applyMiddleware(ReduxThunk))}>
        <RouterComponent />
      </Provider>
    );
  }
}
```

Figure 7 - Redux Store

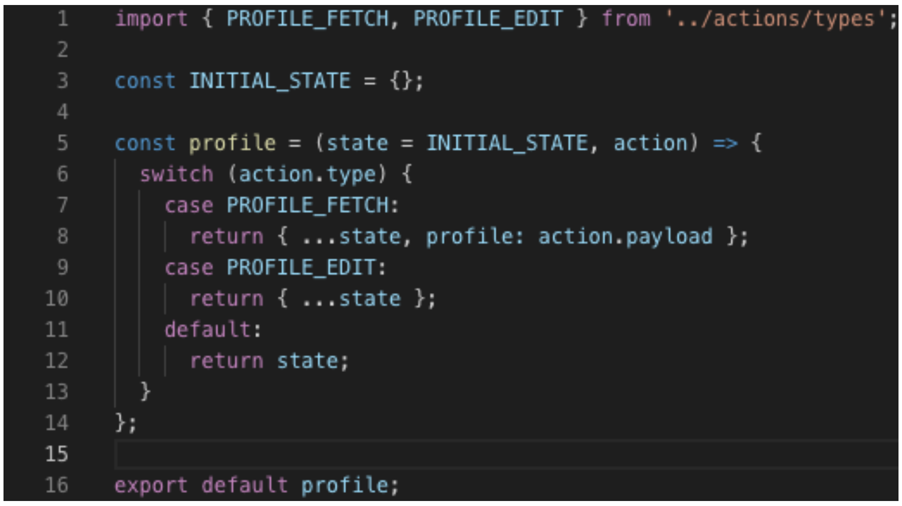
In Figure 6 above, applyMiddleware(ReduxThunk) is utilised in order to create a store enhancer which will apply the thunkMiddleware to the store’s dispatch function. The reducer is created in a file, index.js inside the ‘reducers’ folder. This combines all of the reducers, which are split into four separate groups in order to deal with providing functions for the application. This includes providing functions for authentication methods, profile methods, activity methods and achievement methods. The reducers are highlighted in Figure 8.



```
import { combineReducers } from 'redux';
import auth from './AuthReducer';
import profile from './ProfileReducer';
import activity from './ActivityReducer';
import achievement from './AchievementReducer';

export default combineReducers({
  auth: auth,
  profile: profile,
  activity: activity,
  achievement: achievement
});
```

Figure 8 - Redux Reducers

Each reducer will specify how the application's state changes in response to actions sent to the store. For example, when a user user logs in to the application, an action will be fired which will instruct the application to fetch the user's profile from the back-end database, updating the store respectively. Therefore the profile reducer includes the case 'PROFILE_FETCH', which is highlighted in Figure 9.



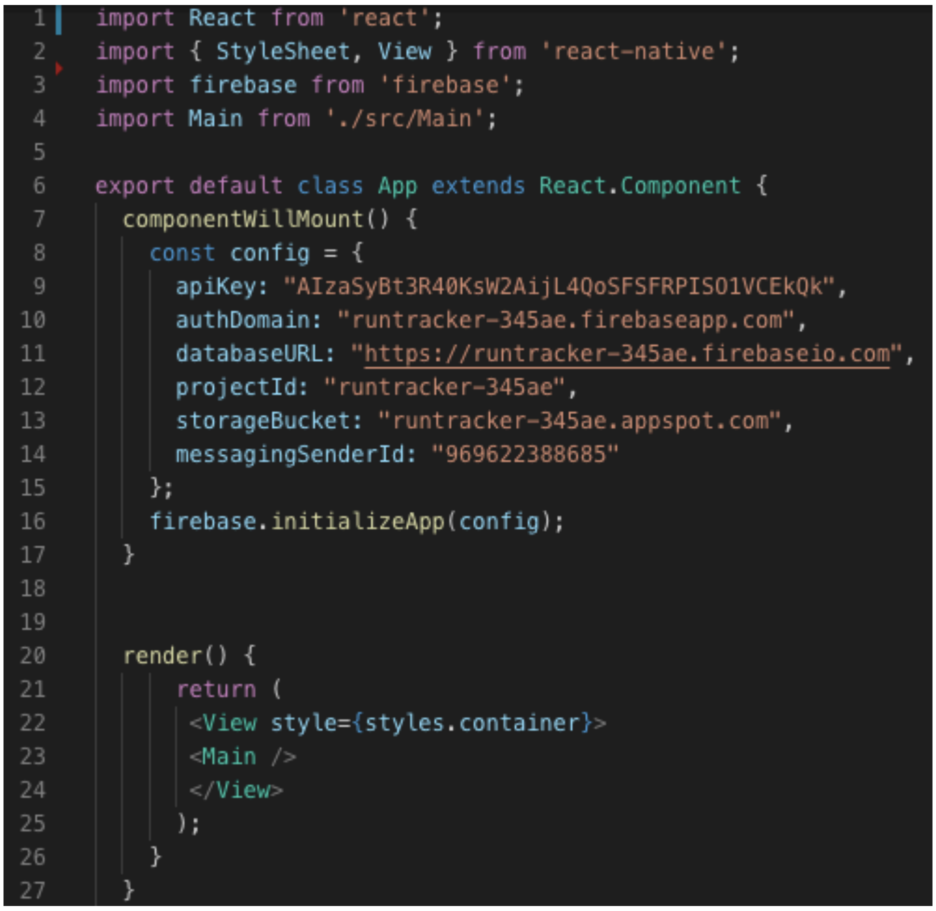
```
import { PROFILE_FETCH, PROFILE_EDIT } from '../actions/types';

const INITIAL_STATE = {};

const profile = (state = INITIAL_STATE, action) => {
  switch (action.type) {
    case PROFILE_FETCH:
      return { ...state, profile: action.payload };
    case PROFILE_EDIT:
      return { ...state };
    default:
      return state;
  }
};

export default profile;
```

Figure 9 - Profile Reducer

The response of the database query (action.payload), will then be mapped to the user's props under an object named profile in order to give the user access to this information. This allows the application to render the user's profile information to the screen through the profile object. Furthermore given that the profile information is now accessible through the Redux store, it can be utilised in other areas of the application without having to contact the database again. This reduces the amount of database queries the applications creates, by instead requesting the information from the Redux store using connect if it is required again.

4.3 Firebase

In order to connect to the back-end database, the Firebase configuration is handled within App.js in the application lifecycle step componentWillMount. This causes the database configuration and initialisation to be invoked just before mounting occurs and prior to the render function executing. The database configuration can be visualised in Figure 10.



```
import React from 'react';
import { StyleSheet, View } from 'react-native';
import firebase from 'firebase';
import Main from './src/Main';

export default class App extends React.Component {
  componentWillMount() {
    const config = {
      apiKey: "AIzaSyBt3R40KsW2AijL4QoSFSFRPISO1VCEkQk",
      authDomain: "runtracker-345ae.firebaseapp.com",
      databaseURL: "https://runtracker-345ae.firebaseio.com",
      projectId: "runtracker-345ae",
      storageBucket: "runtracker-345ae.appspot.com",
      messagingSenderId: "969622388685"
    };
    firebase.initializeApp(config);
  }


  render() {
      return (
       <View style={styles.container}>
       <Main />
       </View>
      );
    }
  }
```

Figure 10 - Firebase Initialisation

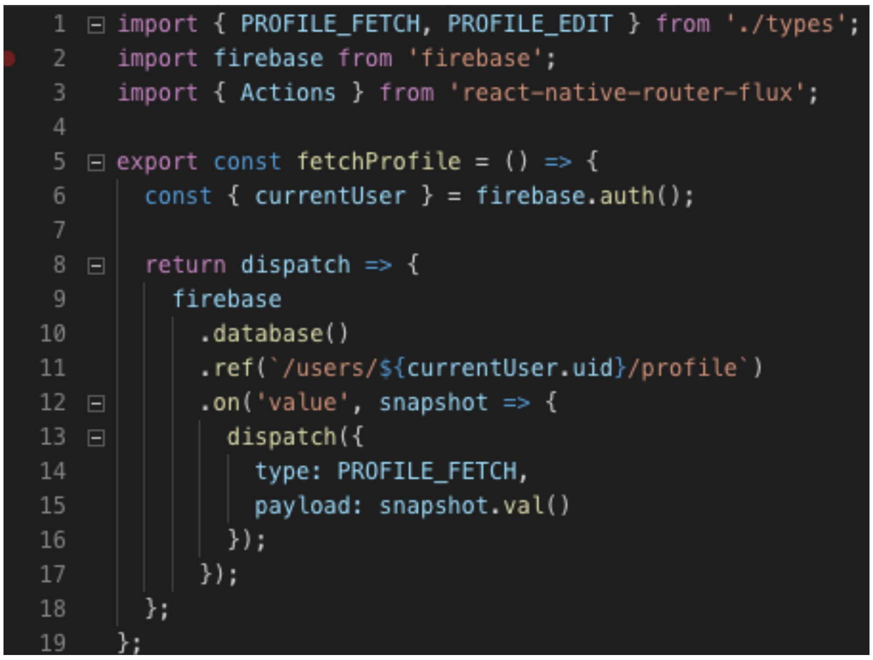
This implementation ensures that each time the application is instantiated, the application loads and connects to the back-end database at the provided ‘databaseURL’. Following connecting to the database, the application will return the Main component which connects to the Redux store and passes this to all of the child components.

The configuration of the database can remain public due to the database rules that are implemented. This involves incorporating the predefined ‘auth’ variable to create a complete solution for securing all of the user data. Once a user authenticates with the system, the ‘auth’ variable in Firebase Database Rules will be populated with the user’s information This contains a user’s unique identifier (uid) as well as their linked account data such as their email address [34]. An individual will only be able to write to the database if their user_id exactly matches their authentication uid. Therefore user’s must be logged in their account and will only be able to edit their own user data. Furthermore rules are also implemented in order to ensure security in terms of reading user data, which follows the same concept. In order to fetch information from the database, users must both be authenticated with the service and will only be able to read user information which exactly matches their corresponding authentication uid. The complete list of database rules implemented can be seen in Figure 11.

```
{
  "rules": {
    "users": {
      "$user_id": {
        ".write": "$user_id === auth.uid”,
            ".read": "auth != null && auth.uid == $uid"
                          }
                      }
    "activity": {
      "$user_id": {
        ".write": "$user_id === auth.uid”,
            ".read": "auth != null && auth.uid == $uid"
                          }
                      }
“new_activity": {
      "$user_id": {
        ".write": "$user_id === auth.uid”,
            ".read": "auth != null && auth.uid == $uid"
                          }
                      }
“achievements": {
      "$user_id": {
        ".write": "$user_id === auth.uid”,
            ".read": "auth != null && auth.uid == $uid"
                          }
                      }
             }
}
```

Figure 11 - Firebase Database Rules

Given that the implementation in Figure 11 provides a secure way of allowing the user to both read and write to the database, the way in which database queries are executed must also be considered. In order to provide modularity to the code and implement an effective means of separation of concerns across the application, database queries will only be executed within actions that are handled by a reducer. On the profile page, prior to the component mounting, the asynchronous database call fetchProfile will be executed. This firstly confirms the user’s authentication uid against the firebase service and then executes the query to return information present given in the location provided by the ‘ref’. As long as the user is authenticated with the service (has previously created an account and logged in), then the database rules will allow the user to fetch the relevant data. This query can be seen in Figure 12.



```
import { PROFILE_FETCH, PROFILE_EDIT } from './types';
import firebase from 'firebase';
import { Actions } from 'react-native-router-flux';

export const fetchProfile = () => {
  const { currentUser } = firebase.auth();

  return dispatch => {
    firebase
      .database()
      .ref(`/users/${currentUser.uid}/profile`)
      .on('value', snapshot => {
        dispatch({
          type: PROFILE_FETCH,
          payload: snapshot.val()
        });
      });
  };
};
```

Figure 12 - Firebase Profile Fetch Query

4.4 User Account Creation

When the user would like to create a new account with the application, they first must enter an email and password which will act as their credentials in order to log into the system. The sign up screen which can be visualised in Figure 13, consists of two text input fields to allow the user to input their credentials. Selecting the link below the sign up button will take the user back to the ‘Log in’ screen and selecting the button ‘Sign up’ will begin the user’s sign up process and call the action ‘createUser’ from the authentication actions. The ‘createUser’ authentication action, highlighted in Figure 14, will create a user account with the backend database, securely salting and hashing the password utilising bcrypt [35]. This ensures that no passwords are stored as plaintext, providing an extra layer of security to the user’s account. When creating a user, functions are also provided including ‘createUserSuccess’ and ‘createUserFail’, in order to handle any errors that may occur when creating a user account. In order to invoke ‘createUserSucess’, an email address must be provided in the correct format and a password must consist of at least 8 alphanumeric characters. If these conditions are met, an action will be called in order to transition the user to the first screen in the sign up process, gathering personal information from the user. If the condition is not met, a dispatch will occur of type ‘AUTH_CREATE_USER_FAIL’, in which the AuthReducer will then return the error message to be rendered on screen to the user.

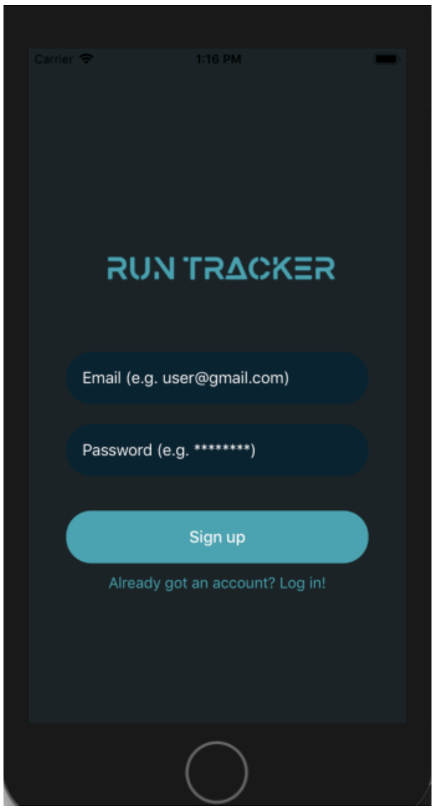
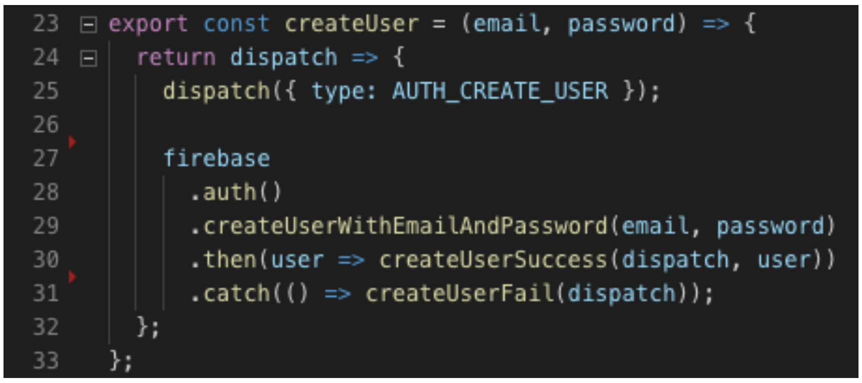
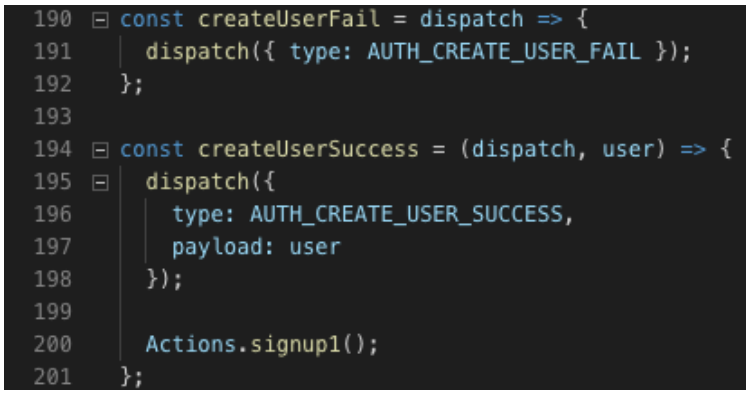


Figure 13 - User Sign Up Screen



```
export const createUser = (email, password) => {
  return dispatch => {
    dispatch({ type: AUTH_CREATE_USER });

    firebase
      .auth()
      .createUserWithEmailAndPassword(email, password)
      .then(user => createUserSuccess(dispatch, user))
      .catch(() => createUserFail(dispatch));
  };
};
```

Figure 14 - Create User Account Action

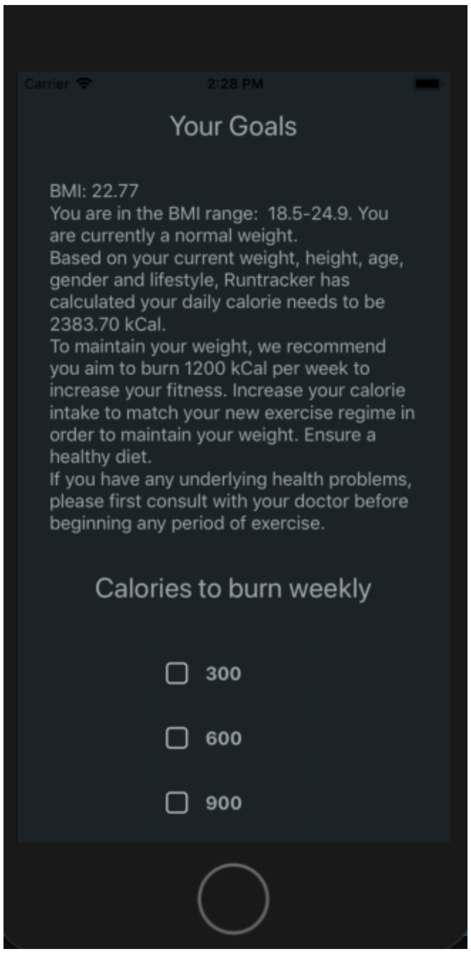


```
const createUserFail = dispatch => {
  dispatch({ type: AUTH_CREATE_USER_FAIL });
};

const createUserSuccess = (dispatch, user) => {
  dispatch({
    type: AUTH_CREATE_USER_SUCCESS,
    payload: user
  });

  Actions.signup1();
};
```

Figure 15 - Create User Success/Fail

Once the user's account has been created successfully, the user will move through the eight separate sign up screens of the application. The first sign up screen will require the user to provide their first and last name, along with their gender and date of birth. The user will also be required to accept the terms and conditions of the application on this screen. Throughout each of the sign up screens, the user will not be able to progress unless all of the fields have been filled out and completed, otherwise an error message will be presented to the user. The second sign up screen is used in order to ensure the user provides their explicit consent for the application to conduct sensitive data processing and data transfer outside of the country. Furthermore location data access is also required on the second sign up screen, which grants the application permission to track the user's location data. This is an essential step in order to ensure the GPS tracking functionality can communicate with the mobile phone co-ordinates to track the user's location when an activity is recording. The third sign up screen is utilised in order to gather the user's height along with the fourth sign up screen, which is utilised to gather the user's current weight. The fifth sign up screen is used for allowing the user to select a goal that they would like to achieve whilst using the application. This allows the user to choose from three different options, which include 'Lose weight', 'Maintain weight' and 'Gain weight'. The sixth sign up screen is utilised in order to gather the user's current physical activity level, which is required for calculating their recommended daily calorie needs. A user must select an option of either 'Sedentary', 'Low active', 'Active' or 'Very active' in order to progress to the next screen. The seventh sign up screen is utilised in order to allow the user to upload a profile picture of themselves, which will be displayed on their user profile at all times.

The final sign up screen collates the information that the user has provided so far in order to inform the user of useful health metrics that can be gathered, given their provided age, gender, height, weight and physical activity level. For example, when considering a male user that is 24 years old, who has a height 165cm, a weight of 62kg, an active lifestyle and also a goal that they are looking to maintain their weight, the information provided in Figure 16 will be displayed to the user.


Carrier 2:28 PM

Your Goals

BMI: 22.77
You are in the BMI range: 18.5-24.9. You are currently a normal weight.
Based on your current weight, height, age, gender and lifestyle, Runtracker has calculated your daily calorie needs to be 2383.70 kCal.
To maintain your weight, we recommend you aim to burn 1200 kCal per week to increase your fitness. Increase your calorie intake to match your new exercise regime in order to maintain your weight. Ensure a healthy diet.
If you have any underlying health problems, please first consult with your doctor before beginning any period of exercise.

Calories to burn weekly

☐ 300

☐ 600

☐ 900

Figure 16 - Complete Sign Up

The health metrics that are presented to the user firstly include providing the user with their BMI. This is calculated by taking the weight of the user in kilograms, divided by the height (squared) of the user in centimetres [36]. Given the user's BMI, the user's BMI range is provided alongside this in order to inform the user of their current weight range.

Based on eq. (1) and eq. (2) provided in the first chapter, the user's daily calorie needs can be calculated from the user's weight, height, age, gender and physical activity level (lifestyle). This provides the user with a calorie goal to work towards each day in order to ensure that they are currently consuming enough calories to match their exercise regime.

Furthermore a weekly calorie goal is then recommended to the user, depending on their choices throughout the sign up process. For example if a user wants to lose weight, a higher calorie goal will be recommended and conversely if a user wants to gain weight, a lower calorie goal is recommended. The user is not forced to selected the recommend goal and can independently select a different goal if they desire.

Due to health complications that different users will inherently have, users with any underlying health problems are instructed to consult with their doctor before conducting any period of exercise using the application. This is an important measure of compliance which ensures that the application serves as a tool to enhance user's fitness and improve their overall health, rather than introducing unnecessary complications to users with underlying health problems, such as those with a family history of heart disease.

Scrolling to the bottom of the sign up screen will display the rest of the options for the user to select in terms of the calories that they would like to burn weekly, as well as a button in order to complete the sign up process. Completing the sign up process will navigate the user to the user's profile, in which they can access the rest of the application through the tab bar navigator at the bottom of the screen.

4.5 Recording An Activity

Once the user selects the record icon at the bottom to the tab navigator, they will be transitioned to the 'Record' screen. This screen is utilised for allowing users to complete a running activity, whilst tracking their location on a map and visualising metrics available such as the time taken since the activity started, the distance travelled, the user's average speed and the kCal burned during the activity.

Given the user has provided location permissions during the sign up process, the user's location can be continuously drawn on a map, which updates as the user moves throughout the world. In order to display this component to the user in real-time, 'react-native-maps' is utilised to provide an API to either Google/Apple Maps depending on the software the phone has installed [37]. This provides the 'MapView' component which renders a map based on the user's region, as well as providing a 'Polyline', which acts as a child component to draw the user's route on the map as they complete a running activity. The implementation of the screen can be seen in Figure 17.

The timer is implemented through utilising the JavaScript package 'minutes-seconds-milliseconds', which provides a time formatter that begins to increment once the user selects the 'Start' button [38]. Once an activity has begun, the application will continuously watch to find a change in the user's location through utilising the following function:

```
let location = await Location.getCurrentPositionAsync({});
```

Each time a user's location changes, new latitude and longitude coordinates are provided from the respective Maps API in order to update the user's route, which is drawn on screen using the Polyline component. The user's route is stored as an array of coordinates and the distance travelled is calculated through utilising the haversine formula [39]. The haversine formula calculates the distance between two latitude and longitude coordinates and each time this calculation occurs, the resulting distance is appended to a variable which displays the distance travelled by the user.

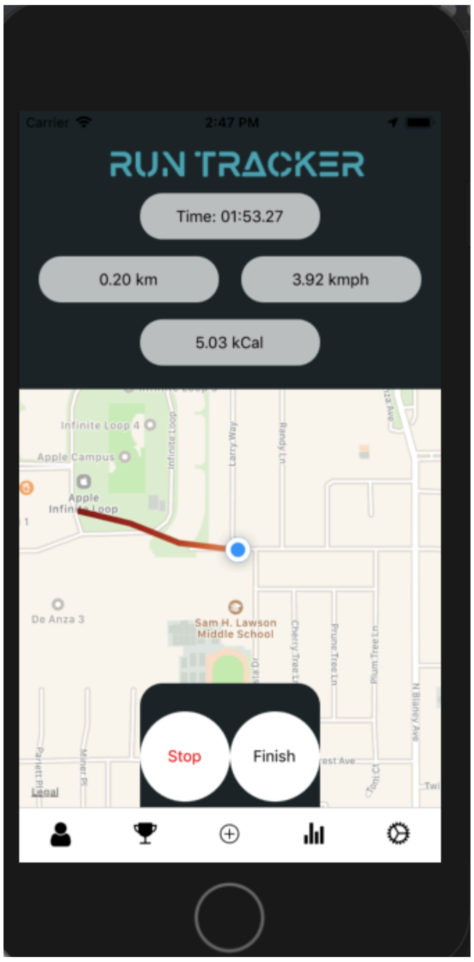


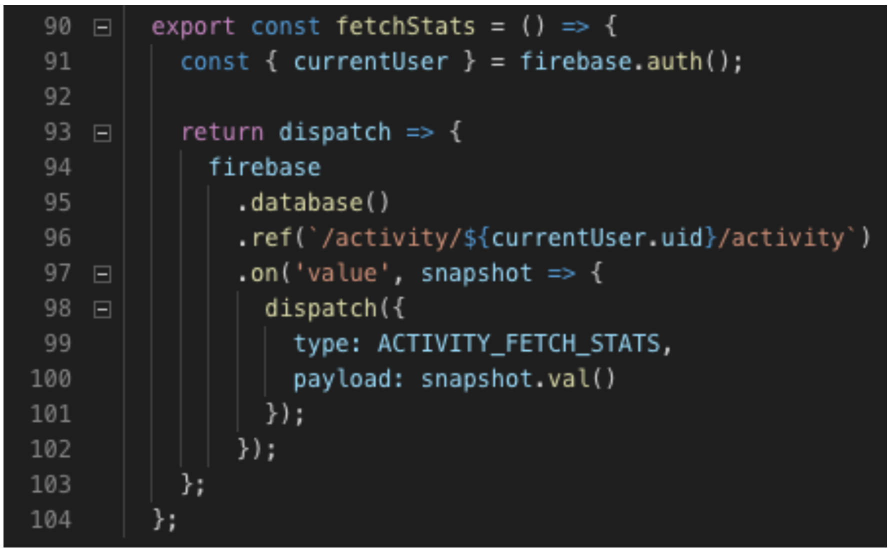
Figure 17 - Record Activity Screen

The user's current speed is displayed and provided to the user given the resulting payload from Maps API. Lastly the user's kCal burned is displayed to the user, which is calculated by considering the user's weight and current speed. As the location continuously changes while the user runs, the value for the amount of kCal the user has burned updates in order to display an accurate value, considering the MET tables for running at different speeds.

Once a user has completed an activity, they can select the 'Finish' button, which fires an action to take a photo of the resulting map and route taken, then transitions the user to a screen to save the activity data. On this final screen, users can also enter a note in order to record information regarding their activity.

4.6 Data Analytics

The data analytics page implemented utilises a database request in order to retrieve all of the user's saved activities. This request can be visualised in Figure 18. If the user has not yet completed any activities an informative error message will display to the user informing them that they will be required to complete an activity, before they are able to view their data analytics.



```
export const fetchStats = () => {
  const { currentUser } = firebase.auth();

  return dispatch => {
    firebase
      .database()
      .ref(`/activity/${currentUser.uid}/activity`)
      .on('value', snapshot => {
        dispatch({
          type: ACTIVITY_FETCH_STATS,
          payload: snapshot.val()
        });
      });
  };
};
```

Figure 18 - Data Analytics Request

Given the constraints of utilising Firebase, a request cannot be constructed in order to only fetch the activities completed within a set date being the current week. Therefore all of the activities are fetched and the data is sorted on the client in order to only display the activities completed in the last week to the user. The data analytics tab will currently only display the statistics for the activities that the user has completed over the course of the last week, in order to encourage the user to continually work towards completing their calories burned goal, which resets weekly.

In order to display this information to the user in a clear and easy to read manner, 'react-native-chart-kit' is implemented in order display a number of different line charts to the user. The user is able to scroll to select the data they wish to view, which includes their distance data, speed data and kCal burned, sorted by date. The result of implementing this page can be visualised in Figure 19.

A line chart was chosen for this implementation as it allows the user to clearly see the progression made over a period time. Each time a user completes an activity, their performance can be tracked and compared to previous activities by selecting a metric from a the scrolling picker module implemented.

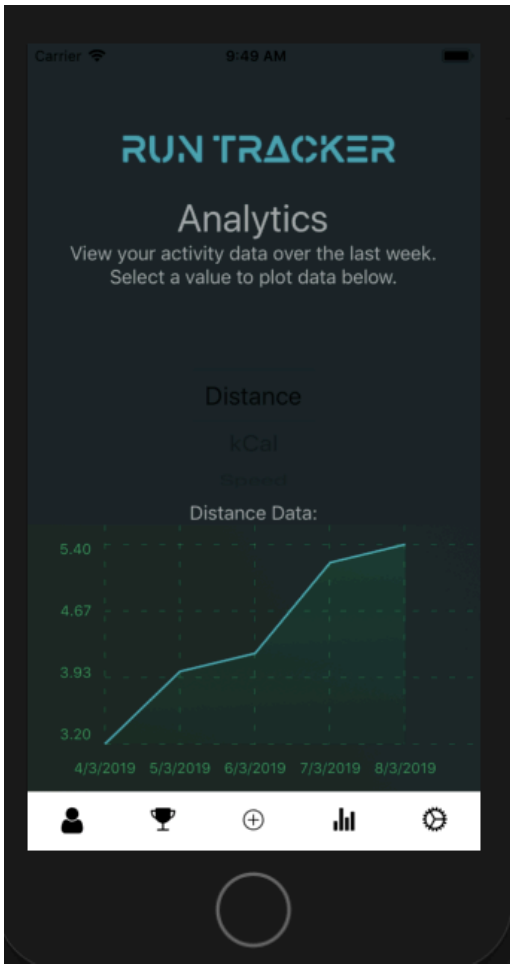


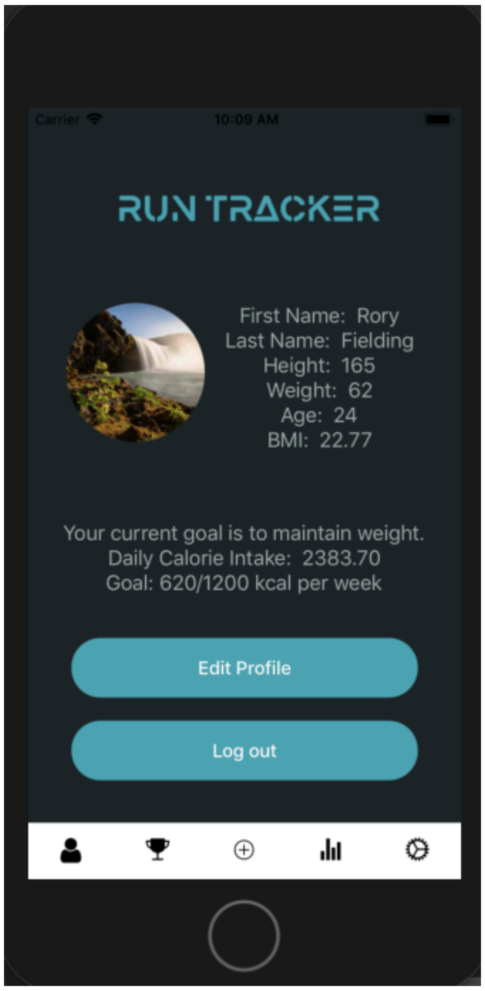
Figure 19 - Data Analytics Screen

4.7 User Profile & Goal Tracking

The user profile screen implemented allows the user to view the data gathered by the application throughout the duration of the sign up process, which can be seen in Figure 20. This includes detailing the user's first and last name, their height, weight, age and BMI. The user's goal, depending on their aim to either maintain, lose or gain weight is displayed to the user along with their recommended daily calorie intake that is calculated using the TEE formula.

Each time a user completes an activity, the user's progress towards their weekly calorie goal is updated in the user's profile screen in order to act as an encouraging reminder for the user to continue exercising. Each week on Monday at 12:01AM, the weekly goal progress resets to 0.

The user's profile screen features two buttons that the user can select. The first of these is the 'Edit Profile' button which allows the user to alter any of the information gathered during the sign up process. If the user decides to alter their height or weight, the values for BMI and daily calorie intake are updated respectively in order to display the correct information to the user. The user is also able to update and upload a new profile picture from the camera's photos from this page. The second button that the user is able to interact with is the “Log out” button. Selecting this button will revoke the user’s access token so they are no longer able to interact with the application. This will also prevent the user from being able to make database requests and they will be transitioned to the Log in screen.



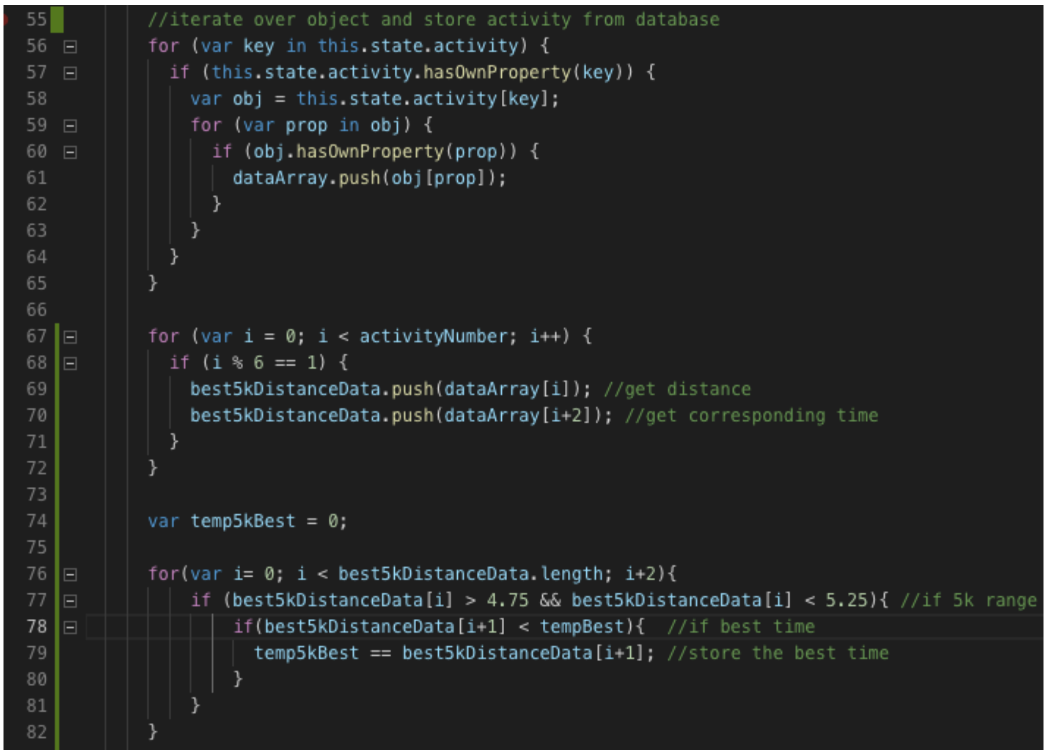
First Name: Rory
Last Name: Fielding
Height: 165
Weight: 62
Age: 24
BMI: 22.77

Your current goal is to maintain weight.
Daily Calorie Intake: 2383.70
Goal: 620/1200 kcal per week

Figure 20 - User Profile Screen

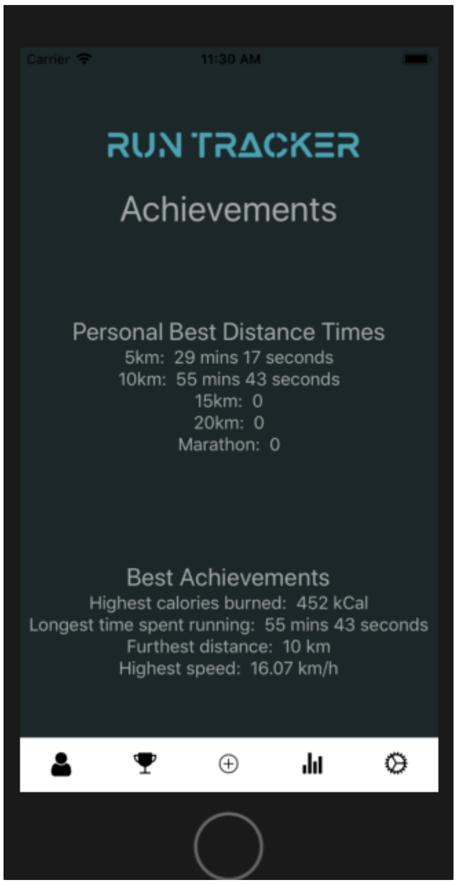
4.8 Achievements

The achievements screen implemented allows the user to view the personal best times for a range of different distances. This is achieved by implementing a database request to fetch all of the activities that the user has completed. This data is fetched prior to the component mounting and the data is then saved to the application state in ‘this.state.activity’. This object is then iterated over in order to gather all of the distance and time data, which is stored in array named ‘distanceData’. This implementation can be visualised in Figure 21. By comparing the values present in this array, the lowest time for an activities distance that is within 5% of the value is then chosen for the user’s personal best. For example, when considering the user’s best 5km running time, activities that are within the range of 4.75 and 5.25km and considered. For each activity, a temporary value is updated as the system iterates over the object in order to select the lowest time that the user completed this activity within. Conversely, the system works in the inverse way in order to select the values for the best achievements, such as the highest calories burned. The activities object is iterated over in order to select the highest value present in the array. This results in the user being able to visualise all of their best achievements while using the application, as Figure 22 highlights.



```
//iterate over object and store activity from database
for (var key in this.state.activity) {
  if (this.state.activity.hasOwnProperty(key)) {
    var obj = this.state.activity[key];
    for (var prop in obj) {
      if (obj.hasOwnProperty(prop)) {
        dataArray.push(obj[prop]);
      }
    }
  }
}

for (var i = 0; i < activityNumber; i++) {
  if (i % 6 == 1) {
    best5kDistanceData.push(dataArray[i]); //get distance
    best5kDistanceData.push(dataArray[i+2]); //get corresponding time
  }
}

var temp5kBest = 0;

for(var i= 0; i < best5kDistanceData.length; i+2){
    if (best5kDistanceData[i] > 4.75 && best5kDistanceData[i] < 5.25){ //if 5k range
        if(best5kDistanceData[i+1] < tempBest){  //if best time
          temp5kBest == best5kDistanceData[i+1]; //store the best time
        }
    }
}
```

Figure 21 - Best 5k Time Sorting



Carrier 11:30 AM

RUN TRACKER

Achievements

Personal Best Distance Times
5km: 29 mins 17 seconds
10km: 55 mins 43 seconds
15km: 0
20km: 0
Marathon: 0

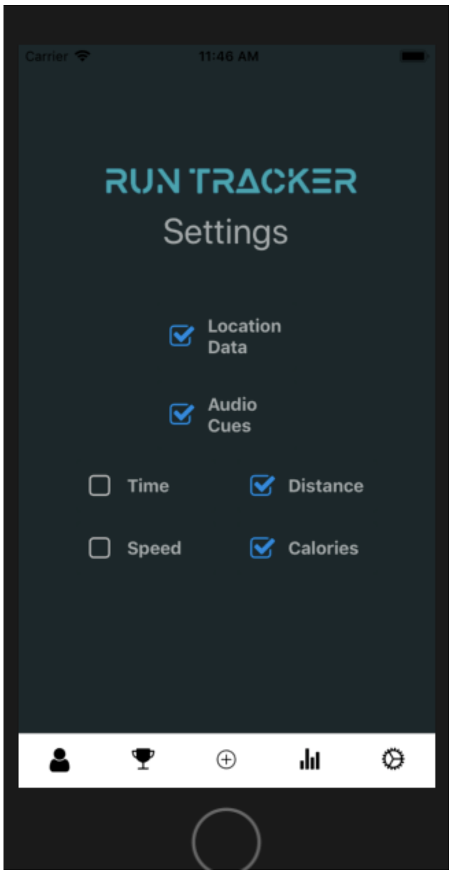
Best Achievements
Highest calories burned: 452 kCal
Longest time spent running: 55 mins 43 seconds
Furthest distance: 10 km
Highest speed: 16.07 km/h

Figure 22 - Achievements Screen

4.9 Settings

The final screen implemented was the application settings screen. This provides the user with two main functions in order to control the experience they receive whilst using the application. The first function is the ability of the user to revoke the application's permissions to access location data within the phone. If the user selects this option, the application will no longer be able to gather location data and therefore the process of recording an activity will not be able to function. However, this is a necessary implementation in order to achieve compliance. Furthermore the user is also able to select different measures of audio cues that they can hear, whilst completing a running activity. This is implemented by setting a flag when a user checks one of the provided boxes and utilising the 'nodejs-text-to-speech' package [40] from the Google Cloud API's library.

With the Audio Cues checkbox selected, the application will perform the text-to-speech request at intervals of every kilometre while the user is completing a running activity. This allows the user to maintain focus on their running activity and receive audio updates regarding their progress. The implementation of this screen can be seen in Figure 23.



✓ Location Data

✓ Audio Cues

☐ Time ✓ Distance

☐ Speed ✓ Calories

Figure 23 - Settings Screen

4.10 Summary

In conclusion, the implementation of the application effectively built upon the design created to deliver a product that utilises the React Native framework, in combination with using Redux and Firebase. This has helped to deliver a system that can manage state efficiently and communicate to a back-end real-time database.

Application security is implemented when considering the database rules in order to ensure that unauthorised users are not able to read or write form the database. Furthermore implementing user authentication also incorporates strong elements of security by avoiding storing passwords as plaintext and instead, utilising bcrypt in order to salt and hash passwords prior to storing them in the database.

The system implemented provides entertainment for the user, whilst they complete a running activity. This is achieved through utilising location tracking and the react-native-maps API in order to display the user's location in real-time, as well as present them information regarding statistics about their activity on screen and through audio cues. The system also provides the function of tracking activity data through the implementation of the data analytics screen. This allows users to select different metrics in order to compare their results against their previous performances over the course of the week. The application implemented also provides an important function of goal tracking, in which users are encouraged to work towards a goal of burning a certain amount of calories each week to improve their overall fitness and meet their fitness goals.

Overall, the system provides an effective means of implementing a complete calorie tracking system. The application implemented gathers information from the user during the sign up process in order to recommend the amount of calories a user should consume daily, in combination with providing the means for the user to track the amount of calories they expend through running activities.

5. TESTING: VERIFICATION & VALIDATION

In order to effectively test the functionality of the application, test cases were created in order to deal with each aspect of the system and ensure that components are working as expected. Test are carried out on both iOS and Android platforms to confirm that functionality and user experience does not differ across the platforms used. Testing was carried out utilising a virtual machine to run the application on a Samsung Galaxy S9 when testing the android application. A virtual machine was also utilised in order to test the application on an iPhone 6s for the iOS platform. The tests used in order to verify and validate the functionality of the application include unit tests, integration tests and user acceptance testing.

5.1 Unit Tests

Unit testing is a level of software testing in which individual components of software are tested. This validates that each unit of software performs as designed [41]. For the purposes of this application, this comprises of testing individual buttons, functions and procedures in order to identify any issues that may impact upon the behaviour of the system. The unit tests implemented are detailed in Table 4.

Table 4 - Unit Tests

Test ID	Description	Expected Outcome	Actual Outcome (iOS)	Actual Outcome (Android)	Action Taken
TC1:LoginSignUpButton	Pressing the sign up button transitions to the user to the sign up screen.	User is transitioned to the sign up form.	User is correctly transitioned to the sign up page.	User is correctly transitioned to the sign up page.	N/A
TC2:SignUpGoBack	Pressing the “Already got an account? Log In!” link transitions the user back to the Log In screen.	User is transitioned to the Log In form.	User is correctly transitioned to the log in page.	User is correctly transitioned to the log in page.	N/A
TC3:SignUp_01_NoFields	Pressing ‘Next’ without entering any information into the fields will display an error message to the user.	Error message states “Please ensure all fields are complete”.	Correct error message displays to user.	Correct error message displays to user.	N/A

Test ID	Description	Expected Outcome	Actual Outcome (iOS)	Actual Outcome (Android)	Action Taken
TC4:SignUp_01_FirstName	Clicking on the ‘First Name’ text box allows the user to enter a first name.	Text input modal is displayed to the user when form item selected.	User is able to enter a first name in the text input modal	User is able to enter a first name in the text input modal	N/A
TC5:SignUp_01_LastName	Clicking on the ‘Last Name’ text box allows the user to enter a last name.	Text input modal is displayed to the user when form item selected.	User is able to enter a last name in the text input modal	User is able to enter a last name in the text input modal	N/A
TC6:SignUp_01_MaleCheckBox	Selecting the box next to ‘Male’ provides a check mark.	Check box is verified once the user selects it.	Check box is accurately verified once selected.	Check box is accurately verified once selected.	N/A
TC7:SignUp_01_FemaleCheckBox	Selecting the box next to ‘Female’ provides a check mark.	Check box is verified once the user selects it.	Check box is accurately verified once selected.	Check box is accurately verified once selected.	N/A
TC8:SignUp_01_InverseCheckBoxSelection	Selecting the Male checkbox when Female is selected unticks Female and ticks Male (and vice-versa).	Check box deselects the previous option once the opposite is selected.	Previous check box is deselected, current check box is selected.	Previous check box is deselected, current check box is selected.	N/A
TC9:SignUp_01_DateOfBirth	Selecting the Date Of Birth Modal allows the user to select a date, month and year of birth and save this once confirm is pressed.	Modal pops up allowing the user to scroll through options to select a date of birth.	Modal correctly displays, user can select a date of birth and confirm.	Modal correctly displays, user can select a date of birth and confirm.	N/A
TC10:SignUp_01_TermsAndConditionsCheckBox	Selecting the box underneath terms and conditions provides a check mark.	Check box is verified once the user selects it.	Check box is accurately verified once selected.	Check box is accurately verified once selected.	N/A

Test ID	Description	Expected Outcome	Actual Outcome (iOS)	Actual Outcome (Android)	Action Taken
TC11:SignUp_01_TermsAndConditionsLink	Clicking on the terms and conditions link.	Correctly transitions the user to a page where they can scroll through the terms and conditions of the application.	No transition.	No transition.	Added terms and conditions to route path in order to fix transition from action.
TC12:SignUp_01_NextAllFieldsComplete	Pressing next with all fields completed.	User should only be able to transition to the consent screen once all fields have been completed.	If all fields are not completed, user cannot move forwards and error message displays. If fields are completed the user correctly transitions.	If all fields are not completed, user cannot move forwards and error message displays. If fields are completed the user correctly transitions.	N/A
TC13:SignUp_02_NoFields	Pressing 'Next' before all check boxes are selected.	Error messages displays to user stating "Please ensure all fields are complete".	Correct error message is displayed to the user.	Correct error message is displayed to the user.	N/A
TC14:SignUp_02_SDP	Select the checkbox next to 'Sensitive Data Processing'.	Check box is verified once the user selects it.	Check box is accurately verified once selected.	Check box is accurately verified once selected.	N/A
TC15:SignUp_02_DTOC	Select the checkbox next to 'Data Transfer Across Country'.	Check box is verified once the user selects it.	Check box is accurately verified once selected.	Check box is accurately verified once selected.	N/A
TC16:SignUp_02_LDA	Select the checkbox next to "Location Data Access".	Check box is verified once the user selects it.	Check box is accurately verified once selected.	Check box is accurately verified once selected.	N/A
TC17:SignUp_02_Back	Press back button.	User should be transitioned back to the initial sign up screen.	User is correctly transitioned back to the first sign up screen.	User is correctly transitioned back to the first sign up screen.	N/A
TC18:SignUp_02_AllFields	Press 'Next' once all check boxes have been selected.	No error message is displayed and the user transitions to Height screen.	User is correctly transitioned to the height screen.	User is correctly transitioned to the height screen.	N/A

Test ID	Description	Expected Outcome	Actual Outcome (iOS)	Actual Outcome (Android)	Action Taken
TC19:SignUp_03_NoFields	Press ‘Next’ with no information entered into height text input.	Error message displays to user stating “Please enter your height”.	Correct error message is displayed to the user.	Correct error message is displayed to the user.	N/A
TC20:SignUp_03_HeightInput	Selecting the text input modal allows the user to enter their height.	User can select the modal and enter a height.	Modal can be accessed and the user’s height can be entered.	Modal can be accessed and the user’s height can be entered.	N/A
TC21:SignUp_03_GoBack	Press back button.	User should be transitioned back to the consent screen.	User is correctly transitioned back to the consent screen.	User is correctly transitioned back to the consent screen.	N/A
TC22:SignUp_03_Complete	Press ‘Next’ after inputting information into the height input modal.	User should be transitioned to the weight screen.	User is correctly transitioned to the weight screen.	User is correctly transitioned to the weight screen.	N/A
TC23:SignUp_04_NoFields	Press ‘Next' with no information into the weight text input.	Error message displays to the user stating “Please enter your weight”.	Correct error message is displayed to the user.	Correct error message is displayed to the user.	N/A
TC24:SignUp_04_WeightInput	Selecting the text input modal allows the user to enter their weight.	User can select the modal and enter a weight.	Modal can be accessed and the user's weight can be entered.	Modal can be accessed and the user's weight can be entered.	N/A
TC25:SignUp_04_GoBack	Press back button.	User should be transitioned back to the height screen.	User is correctly transitioned back to the height screen.	User is correctly transitioned back to the height screen.	N/A
TC26:SignUp_04_Complete	Press ‘Next’ after inputting information into the weight input modal.	User should be transitioned to the goal screen.	User is correctly transitioned to the goal screen.	User is correctly transitioned to the goal screen.	N/A
TC27:SignUp_05_NoFields	Press ‘Next’ with no checkbox selected.	Error message displays to the user stating "Please enter a fitness goal”.	Correct error message is displayed to the user.	Correct error message is displayed to the user.	N/A

Test ID	**Description**	**Expected Outcome**	**Actual Outcome (iOS)**	**Actual Outcome (Android)**	**Action Taken**
TC28:SignUp_05_CheckEachBox	Press each check box once.	Pressing a check box should verify the selection. Pressing another checkbox should deselect previous option and select current.	Check box is verified once selected and previous selection is unchecked.	Check box is verified once selected and previous selection is unchecked.	N/A
TC29:SignUp_05_GoBack	Press back button.	User should be transitioned back to the weight screen.	User is correctly transitioned back to the weight screen.	User is correctly transitioned back to the weight screen.	N/A
TC30:SignUp_05_Complete	Press ‘Next' after selecting a check box.	User should be transitioned to the lifestyle screen.	User is correctly transitioned to the lifestyle screen.	User is correctly transitioned to the lifestyle screen.	N/A
TC31:SignUp_06_NoFields	Press ‘Next’ with no checkbox selected.	Error message displays to the user stating "Please select an activity level”.	Correct error message is displayed to the user.	Correct error message is displayed to the user.	N/A
TC32:SignUp_06_CheckEachBox	Press each check box once.	Pressing a check box should verify the selection. Pressing another checkbox should deselect previous option and select current.	Check box is verified once selected and previous selection is unchecked.	Check box is verified once selected and previous selection is unchecked.	N/A
TC33:SignUp_06_GoBack	Press back button.	User should be transitioned to the transitioned to goal screen.	User is correctly transitioned to the goal screen.	User is correctly transitioned to the goal screen.	N/A
TC34:SignUp_06_Complete	Press ‘Next’ after selecting a check box.	User is transitioned to the profile picture upload screen.	User is correctly transitioned to the profile picture upload screen.	User is correctly transitioned to the profile picture upload screen.	N/A

Test ID	Description	Expected Outcome	Actual Outcome (iOS)	Actual Outcome (Android)	Action Taken
TC35:SignUp_07_NoFields	Press 'Next' with no image selected.	Error message displaying to user stating "Please upload a profile picture".	Correct error message is displayed to the user.	Correct error message is displayed to the user.	N/A
TC36:SignUp_07_PickImage	Press 'Pick an image'.	Camera permissions are requested and user can choose an image from phone's photos.	User is able to choose an image after accepting camera permissions.	User is able to choose an image after accepting camera permissions.	N/A
TC37:SignUp_07_GoBack	Press back button.	User is transitioned back to the lifestyle screen.	User is correctly transitioned back to the lifestyle screen.	User is correctly transitioned back to the lifestyle screen.	N/A
TC38:SignUp_07_Complete	Press 'Next' after uploading a profile picture.	User is transitioned to the final sign up screen.	User is correctly transitioned to the final sign up screen.	User is correctly transitioned to the final sign up screen.	N/A
TC39:SignUp_08_BMI	Check the BMI value provided.	BMI value should be calculated correctly and the BMI range should match value.	Based on height and weight, the correct calculation is provided. Range is correct.	Based on height and weight, the correct calculation is provided. Range is correct.	N/A
TC40:SignUp_08_TEE	Check the TEE value provided.	TEE value should be calculated correctly based on age, weight, height, gender and PA level.	Correct value is provided to the user.	Correct value is provided to the user.	N/A
TC41:SignUp_08_GoalMatch	Check the goal is correct given the information from the goal screen.	If a user states they wanted to lose weight, lose weight should be displayed to user.	Correct goal is provided to the user based on their prior selection.	Correct goal is provided to the user based on their prior selection.	N/A
TC42:SignUp_08_RecommendedBurn	Check the value provided for recommended calories to burn weekly.	Value increases based on user goal, user can change value if desired.	Value increases appropriately. User can alter the goal if they select another box.	Value increases appropriately. User can alter the goal if they select another box.	N/A

Test ID	Description	Expected Outcome	Actual Outcome (iOS)	Actual Outcome (Android)	Action Taken
TC43:SignUp_08_GoBack	Press back button.	User is transitioned back to the profile picture upload screen.	User is correctly transitioned back to the profile picture upload screen.	User is correctly transitioned back to the profile picture upload screen.	N/A
TC44:SignUp_08_Complete	Press 'Complete Sign Up'.	User profile is completed and user is transitioned to the profile screen in the application.	User profile completes. User transitions to the profile page.	User profile completes. User transitions to the profile page.	N/A
TC45:Profile_LogOut	Press Log out on the profile screen.	User's access token is revoked and user goes to Log in screen.	Access token is revoked. User transitions to correct screen.	Access token is revoked. User transitions to correct screen.	N/A
TC46:Achievement_Tab_Navigation	Press the trophy icon in the tab bar at the bottom of the screen..	User is transitioned to the achievements screen.	User is transitioned to the achievements screen.	User is transitioned to the achievements screen.	N/A
TC47:Record_Activity_Tab_Navigation	Press the plus icon in the tab bar at the bottom of the screen.	User is transitioned to the record activity screen.	User is transitioned to the record activity screen.	User is transitioned to the record activity screen.	N/A
TC48:Analytics_Tab_Navigation	Press the bar chart icon in the tab bar at the bottom of the screen..	User is transitioned to the analytics screen.	User is transitioned to the analytics screen.	User is transitioned to the analytics screen.	N/A
TC49:Settings_Tab_Navigation	Press the cog icon in the tab bar at the bottom of the screen..	User is transitioned to the settings screen.	User is transitioned to the settings screen.	User is transitioned to the settings screen.	N/A
TC50:Profile_Tab_Navigation	Press the person icon in the tab bar at the bottom of the screen..	User is transitioned to the profile screen.	User is transitioned to the profile screen.	User is transitioned to the profile screen.	N/A
TC51:Profile_Edit	Press 'Edit Profile' on the profile screen.	User is transitioned to edit profile screen.	User is correctly transitioned to edit profile screen.	User is correctly transitioned to edit profile screen.	N/A

Test ID	Description	Expected Outcome	Actual Outcome (iOS)	Actual Outcome (Android)	Action Taken
TC52:Profile_Cancel_Edit	Press 'Cancel' on the edit profile screen.	User is transitioned back to the profile screen.	User is correctly transitioned back to the profile screen.	User is correctly transitioned back to the profile screen.	N/A
TC53:Profile_Save_Edit	Press 'Save' on the edit profile screen.	User is transitioned back to the profile screen.	User is not transitioned.	User is not transitioned.	Fix routes in the button action in order to correctly transition the user.
TC54:Record_Location_Permissions	If location permissions are granted, map should load.	User's position is loaded on the map.	User's position loads on the map after a few seconds.	User's position loads on the map after a few seconds.	N/A
TC55:Record_Start	Press start button on the record screen.	Buttons change to reset and finish. Timer increments, distance increments, kCal increments and speed displays. Polyline follows user route on map.	All tracking functionality works, however distance calculation is inaccurate.	All tracking functionality works, however distance calculation is inaccurate.	Distance metrics altered within haversine equation to display the correct distance travelled by the user.
TC56:Record_Reset	Press the reset button after starting an activity on the record screen.	Distance, Timer, kCal and Speed all reset to 0.	All variables reset to 0.	All variables reset to 0.	N/A
TC57:Record_Finish	Press the finish button after starting an activity on the record screen.	User is transitioned to the activity complete screen.	User is correctly transitioned to the activity complete screen.	User is correctly transitioned to the activity complete screen.	N/A
TC58:Completed_Activity_Save	Press the save button on the completed activity screen.	User is transitioned to the analytics screen.	User is correctly transitioned to the analytics screen.	User is correctly transitioned to the analytics screen.	N/A
TC59:Completed_Activity_Cancel	Press the cancel button on the completed activity screen.	User is transitioned back to the record activity screen.	User is correctly transitioned to the record activity screen.	User is correctly transitioned to the record activity screen.	N/A

5.2 Integration Tests

Following the completion of all unit tests, integration tests were implemented in order to combine all of the individual components of the application together and test these as a group [42]. This involved testing the interaction between integrated units, including the interaction between the front-end application and the back-end database including the user authentication system. The most important part of the integration tests was to confirm that actions dispatched from front-end React Native components resulted in the expected outcome in the back-end database.

Table 5 - Integration Tests

Test ID	Description	Expected Outcome	Actual Outcome (iOS)	Actual Outcome (Android)	Action Taken
TC1:SignUp_Completion	Complete sign up. Check data in backend.	The data in the backend database should match user information entered during sign up process.	Backend database accurately matches user entered information.	Backend database accurately matches user entered information.	N/A
TC2:Profile_loaded	Once logged in/ sign up completed, profile should populate with relevant data from backend database.	First name, last name, height, weight, age, BMI, goal, TEE and goal progress are fetched from database.	Database query executes successfully and renders information to user.	Database query executes successfully and renders information to user.	N/A
TC3:Edit_Profile_Save	Enter new information in the edit profile screen and press save.	The backend database is updated with the new information entered by the user.	Database query does not update the record of the user.	Database query does not update the record of the user.	Fixed the database query in order to correctly update the database records of the user given the new information provided.
TC4:Analytics_Load_Null	View the analytics tab on an account with no activities completed.	The tab should inform the user that there is no information present in the database.	An infinite loading spinner appears on the screen.	An infinite loading spinner appears on the screen.	Fixed the case in order to provide a message to the user if the query returns null.

Test ID	Description	Expected Outcome	Actual Outcome (iOS)	Actual Outcome (Android)	Action Taken
TC5:Analytics_Distance_Data	Select the distance data in the analytics tab with an account that has previous activities completed.	The query executes to fetch the distance data from the database and display this to the user.	The query executes successfully and the correct information is displayed to the user.	The query executes successfully and the correct information is displayed to the user.	N/A
TC6:Analytics_Speed_Data	Select the speed data in the analytics tab with an account that has previous actives completed.	The query executes to fetch the speed data from the database and display this to the user.	The query executes successfully and the correct information is displayed to the user.	The query executes successfully and the correct information is displayed to the user.	N/A
TC7:Analytics_kCal_Data	Select the kCal data in the analytics tab with an account that has previous activities completed.	The query executes to fetch the kCal data from the database and display this to the user.	The query executes successfully and the correct information is displayed to the user.	The query executes successfully and the correct information is displayed to the user.	N/A
TC8:Profile_Goal_Update	Complete an activity and check goal progress in the user profile.	The amount of kCal burned during the activity should be added to the user's goal.	The amount of kCal burned during the activity correctly updates the goal in the user's profile.	The amount of kCal burned during the activity correctly updates the goal in the user's profile.	N/A
TC9:Profile_Goal_Reset	The amount of progress towards a user's goal is reset at the beginning of each week.	On Monday at 12:01AM, the user's goal progress resets to 0.	The user's goal progress resets to 0 at the correct time.	The user's goal progress resets to 0 at the correct time.	N/A
TC10:LoginTrue	Enter user details with correct credentials and press 'Log In'.	User is authenticated with the service and is logged into platform.	User logs into platform.	User logs into platform.	N/A
TC11:LoginFalseEmail	Enter user details with incorrect email and press 'Log In'.	User is not authenticated. Error message displays "Login failed, please check the credentials".	Correct error message displays to user	Correct error message displays to user	N/A

Test ID	Description	Expected Outcome	Actual Outcome (iOS)	Actual Outcome (Android)	Action Taken
TC12:LoginFalsePassword	Enter user details with incorrect password and press ‘Log In’.	User is not authenticated. Error message displays “Login failed, please check the credentials”.	Correct error message displays to user	Correct error message displays to user	N/A
TC13:SignUpFalseEmail	Pressing ‘Sign Up’ with an invalid email address produces an error message.	Error message states “Creation failed! Please enter a valid email and an eight character alphanumeric password!.	Correct error message displays to user.	Correct error message displays to user.	N/A
TC14:SignUpFalsePassword	Pressing ‘Sign Up’ with an invalid password produces an error message.	Error message states “Creation failed! Please enter a valid email and an eight character alphanumeric password!.	Correct error message displays to user.	Correct error message displays to user.	N/A
TC15:SignUpCorrect	Pressing ‘Sign Up' with correct credentials transitions the user to the first personal details page.	An account is created with the authentication service and the user is transitioned to screen SignUp_01.	User account is created and user is correctly transitioned to the right page.	User account is created and user is correctly transitioned to the right page.	N/A
TC16:Completed_Activity_Save	Press the save button on the completed activity screen.	User is transitioned to the analytics screen and the backend database is populated with the activity data.	User is correctly transitioned to the analytics screen. Database is populated correctly with activity data.	User is correctly transitioned to the analytics screen. Database is populated correctly with activity data.	N/A
TC17:Profile_Save_Edit	Press ‘Save’ on the edit profile screen.	User is transitioned back to the profile screen. The user’s information is updated in the backend.	User is correctly transitioned and the information updates with correct calculations.	User is correctly transitioned and the information updates with correct calculations.	N/A

5.3 User Acceptance Testing

User acceptance testing (UAT) is the last phase of the software testing process used when developing this application. The goal of UAT is to assess if the system can support day-to-day business and user scenarios, ensuring that the system is sufficient and correct for business usage [43]. During UAT, three different types of users were provided with the application to test the software and make sure that it can handle the required tasks in real-world scenarios, according to specifications. The three types of users chosen include a user who does not run as a form exercise, a user that undertakes causal running (defined as no more than once a week) and a user that undertakes frequent running exercise (multiple times per week). Each user was provided with access to the application for two weeks, at which point they were record their experience with the application. The results of UAT can be seen in Table 6.

Table 6 - User Acceptance Testing

Tester Name	Running Tendency	Positive Feedback	Negative Feedback	Action Taken
Robert Good-Stephenson	Never	“I really like how the application shows you your progress each week and gets you to work towards a goal. It has made me much more interested in running and tracking my calories”.	“I wanted to change my profile picture, however there was no option to do so on the edit profile screen”.	Another modal was implemented on the edit profile screen in order to allow the user to upload a new profile picture.
Tony Maynard	Casual	“The audio cues are a great feature of the application which motivated me to improve my performance whilst running”.	“It’s possible to make notes after completing an activity, however this is character limited and long notes cannot be entered”.	Increased the character limit in order to allow users to enter notes greater than 255 characters.
Sian Baker	Frequent	“The GPS location tracking feature works well and the performance metrics match my experience with other applications.”	“I would like to be able to view all of my previous activity data in the application, not just the data over the past week. There needs to be a way to view all of the activities the user has completed”.	Reserved for future implementation.

5.4 Summary

In conclusion, testing the application proved useful in order to find a number of different issues that were present within the system. There were a few cases found throughout the unit tests highlighting that some of the buttons implemented did not correctly transition the user to the right location as expected. As a result of this, the actions responsible for controlling those buttons were revised in order to ensure that the functionality works as expected. Furthermore it was found that the distance calculation implemented utilising the haversine formula was out by a factor of 10, therefore this was corrected in order to present accurate information to the user regarding the distance they travel when completing an activity.

Integration testing was another important aspect of the testing process, which highlighted that once all components were connected, one database request in order to update the user's profile page was not working an intended. This request was promptly fixed in order to ensure that the user can edit and save the information in the user profile. Furthermore integration testing also highlighted an issue that was present when attempting to view the data analytics for a user which has not completed any activities. This caused a loading spinner to be displayed to the user indefinitely, as the database request could not be completed. A case was added in order to fix this problem, so that if the query returns null, an informative error message will be displayed to the user.

User Acceptance Testing was also conducted in order to allow different types of users to experience the application and report their feedback following a period of use for two weeks. This highlighted two important issues, the first of which being that no option was present for the user to alter their profile picture. This was implemented in the edit profile screen in order to allow a user to upload different photo of their choosing. Furthermore this also highlighted a problem in which activity notes were limited 255 characters. The storage method was therefore changed in order to allow users to enter notes which are not limited to a number of characters. Finally one user requested that they would like to be able to view all of the previous activities they have completed, not just the data for the previous week. This is a change that is feasible to be implemented, however due to time constraints this aspect of improving the application has been reserved for future work.

Overall testing the product thoroughly has proved to be very productive, highlighting areas of the application which needed fixing and were not working as intended. By testing application through unit and integration tests, a more robust and effective system was developed which helped create an effective product. UAT also highlighted further areas which were improved upon in order to enhance the user experience of the application.

6. DISCUSSION: CONTRIBUTION & REFLECTIONS

Following the results of testing the application, the progress of the project can be reviewed. By comparing the results of testing to the Project Initiation Document (PID), the extent to which the initial objects, outputs and features of the application that were achieved can be analysed.

6.1 Objective Outputs & Success

One of the initial project outputs was to analyse the available frameworks for cross-platform mobile development and select the most appropriate tool to conduct development. The framework chosen was React Native, due to this being open source, free to use and having the closest performance to natively built mobile applications in comparison to Xamarin and Ionic. Another reason for selecting React Native was that it would enable the largest amount of code re-usability between different platforms, therefore reducing the amount of platform specific code required. React Native proved to be an effective framework for implementing all of the features required of the application, however this did not solve the problem of removing platform specific code entirely. Even though best design practices were followed, such as creating the size of components based on the window size of the screen, platform specific code was still required to be implemented due to key differences in the way Android and iOS platforms operate. By testing the product thoroughly on both platforms, specific issues related to the individual platforms were able to be found and fixed in order to create a highly similar user experience, regardless of the user's platform.

Following analysing the top fitness tracking applications available in the market, a comprehensive list of features was created in order to discover the essence of what makes a fitness tracking application. Application screen wireframes were created, along with use-cases and a flow-chart in order to display how the user would interact with different components in the system. This was developed further in order to include high-level architectural diagrams of the system and define the database tables in advance. The final design of the application was detailed and useful when implementing the code. However the flow-chart of the application was re-designed four separate times in order to re-iteratively build upon knowledge and experience gained throughout the development process. The design of the application considered security and privacy concerns from the beginning, which allowed the application created to have a robust user authentication system which communicates to a backend database in a secure manner.

The testing of the application highlighted a number of successes that the application achieved in terms of meeting the targeted use cases. This included implementing a GPS location tracking feature which provided the function of entertainment for the user, along with providing them with useful performance metrics whilst undergoing a running activity. Furthermore the goal tracking method implemented worked well and encouraged users to continue exercise activities and work towards meeting their goal each week. The data analytics dashboard also easily allowed users to analyse and compare their running performance metrics across the duration of the week. Testing the application highlighted a few issues that had been overlooked during the development process. These issues were promptly fixed, yet this demonstrated the importance of testing in order to discover unforeseen issues. User Acceptance Testing also proved to be very useful in finding issues with the application that I was unable to discover through my own use.

Overall, the main objective of this project was to create a cross-platform fitness tracking application which communicated to a back-end database. I believe I have created an application which fits this criteria and performs well in order to meet the use cases of providing entertainment, goal tracking and data analysis for the user. The unique concept that this application delivers is the idea of presenting a complete calorie tracking application, which informs the user of the recommended daily calorie needs, as well as calculating a user's calorie expenditure when undergoing exercise. Considering this, I believe the application that was developed to be successful.

6.2 Limitations

One of the main limitations discovered through the process of testing the application is that it is not currently possible to view all of the activity data that the user has completed. It is only possible to view the analytics data for the activities completed in the past week. The application was implemented in this way in order to keep users focused on working towards a weekly goal. However given that the data is still being stored, further development work could be undertaken in order to display all of the past activity data to the user in a different screen, accessible from the analytics tab. This issue with the application was only discovered in user acceptance testing and considering it would require a significant amount of design and development work, implementing this feature was reserved for future work.

Another aspect of the application was that it was designed in order to function effectively at scale. Firebase was chosen as the backend database for this project due to it inherently having a strong integration with React Native, allowing for a greater ease of implementation. Furthermore having not used a NoSQL database before, developing with Firebase acted as a productive learning experience in utilising new software tools. The main problem found when using Firebase and NoSQL databases in general is the limitations present in querying the database. Considering the database implemented, it was not possible to implement an effective query to gather database objects within a rolling weekly date range. Therefore an inefficient query was implemented to return all of the activities completed, which were then sorted client-side. This is problematic and not efficient at scale. For example, if the user had completed upwards of 100 activities in total but only two in the past week, all 100 activities would have to be sorted in the client-side to display the information in the correct date range to the user. Therefore if I was to undertake this project again, I would have implemented an SQL database in order to utilise more powerful queries.

At the beginning of this project I had a limited experience in using both JavaScript and React Native. Completing this project was a significant learning experience that not only helped to develop my own skills in programming in a new language, but also helped to develop my understanding in working with new technical tools and understanding how to implement a product from the available documentation. The most difficult aspect I found in creating this application was developing a robust user authentication system. The main problem I had not considered at the start of the project was how to create a unique user identification key in order to be able to query the database and be able to access this information across different screens of the application. Due to way in which JavaScript handles state, this proved to be a non-trivial process which required a complete overhaul of the entire application design late quite late into the project development. Given multiple design iterations and attempts at fixing this issue, Redux was eventually found to be a solution in providing a state management system which allowed the user's uid to be accessible across all of the screens of the application. If I would to attempt this project again, Redux would have been implemented at the start of the project which would have saved a significant amount of time building the application, allowing me to develop extra features and improve application performance even further.

In reflection, the process of developing this application has helped me to significantly improve my understanding of building large complex systems. Whilst the application developed is by no means perfect, I believe I have been able to create an effective yet simplified version of the commercially available cross-platform fitness tracking mobile applications in the market. My understanding of JavaScript has vastly improved throughout the duration of this project, along with my ability to understand and work with new tools using documentation.

7. SOCIAL, LEGAL, HEALTH & SAFETY & ETHICAL ISSUES

The system developed will have to consider the impact of a variety of different issues, including legal issues, social issues, health & safety issues and ethical issues.

The application collects a reasonable amount of personal data, as well as a large amount of location data on the user in order to provide the function of GPS tracking. As location based services are used, the privacy of user data is a key element in the social, legal and ethical issues of this project. Due to this, the workflow approach of privacy by design (PbD) was implemented. PbD is an approach to development that takes privacy into account from the very beginning, across all aspects of the project. With the EU's data protection reform, which came into effect in 2018 [44], PbD requirements are part of the law and thus must be considered. The misuse of location data can have serious legal consequences, therefore in order to account for this, the processing and retention of location data is addressed in the terms and conditions of the application. A contract was developed in order to confirm that no uses of location data will violate the laws of the user's country and the user will have to accept this agreement in order to be able to create an account and utilise the application.

Along with ensuring the user accepts the terms and conditions of the application, the user must explicitly consent to three separate clauses when creating a user account. The first of these is to do with data transfer outside of the user's country. Due to General Data Protection Regulation (GDPR), restrictions are imposed on the transfer of personal data outside of the European Union, to third-party countries or international organisations [45]. As the user's data is stored in a cloud-based google service, the user's data may be transferred outside of the country, therefore it is paramount to ensure the individual gives informed consent on this issue, which occurs during the sign up process. Furthermore the user is able to request to delete their data from the application at any time through email in order to be compliant with GDPR. User data is also stored securely utilising encryption in order further enhance system compliance.

The user must also give explicit consent to allow the application to conduct sensitive data processing [46], considering the system collects user data regarding their physical health, including their height, weight and BMI. This is another aspect of GDPR that the application considers, therefore by obtaining consent from the user to allow the system to process this information, no infringements will occur on a legal basis.

In Europe, location data is regulated by a law named the 'Directive on Privacy and Electronic Communications' [47]. This has been implemented in all EU member states and therefore the application project must comply with a number of different points. Given that location data collected is not anonymised, the user most give consent for their location to be used an accessed. This is also handled during the sign up process, where the user must explicitly give consent for the application to access location data, which in turn syncs the application permissions to do so. Additionally, the user must be able to withdraw their consent of this information at any time, which is implemented in the application settings screen. If the user decides to withdraw their consent for the application to collect location data, they can simply check a box removing the application permissions. The application informs the user what location data will be collected and processed within the terms and conditions, as well as what it is processed for and whether it will be transmitted to any third parties, including social media websites. Location data is only used for a specific reason, being tracking a user's running activity, which only occurs for the duration of the activity which is being undertaken.

The application is developed in order to help improve the health and fitness of the individuals that use and interact with it on a regular basis. Considering this, users are instructed during the sign up process to contact their doctor before undertaking any exercise if they have any underlying health issues, or have a history of health problems within their family [48]. The terms and conditions of the application also inform the user that RunTracker is not responsible for any damages or health problems that the user may incur from using the application. Whilst the information presented to the user in the application is based on medical research, individuals are instructed that they are using the application at their own risk. The terms state that it is the individual's responsibility to exercise using the application considering their own safety and within their means.

Given that the application tracks location data on users, this also presents a relatively new social issue considering crimes. In certain cases, the government or a legal institution could request your fitness tracker information and use it against you in a court of law. Chris Bucchere, a San Francisco cyclist who struck and killed an elderly pedestrian was charged with manslaughter after prosecutors obtained data from his GPS-enabled fitness tracker, to highlight that he had been speeding before the accident occurred [49]. This highlights issues of data privacy and brings to light a new debate regarding data ownership. It is unclear who owns the right to a user's location data and who should have access to it, depending on what constitutes this need. Currently, data can be obtained through court on a case-by-case basis, yet further laws are required in order to explicitly state the requirements of fitness tracking applications to co-operate with police and developing new data management technologies.

8. CONCLUSIONS & FUTURE IMPROVEMENTS

8.1 Concluding Remarks

The main objective of the project was to develop a cross-platform fitness tracking application which communicates to a back-end database. Due to the results of testing, this has highlighted that the application developed achieved this objective, meeting three separate use cases. The application provides functions of entertainment through GPS location tracking, provides a goal tracking feature to encourage users to exercise and also provides a data analytics panel in order to allow users to analyse their performance on a weekly basis. The unique feature that this application provides is providing the user with their recommend daily calorie needs, along with providing a means for the user to track their calorie expenditure through running activities. Therefore, this project has resulted in the creation of a complete calorie tracker for users to improve their own personal health and fitness.

8.2 Personal Appraisal

Table 7 provides a comparison regarding the features of the final application developed (RunTracker) to the features of the top fitness tracking applications available in the market place.

Table 7 - Evaluation of RunTracker features to commercial market applications

	RunTracker	MyFitnessPal	MyPacer	Runtastic	Strava	EndoMondo
User Authentication System	✓	✓	✓	✓	✓	✓
Social Media Integration		✓	✓	✓	✓	✓
Calorie intake calculation	✓	✓				
Calorie burned calculation	✓		✓	✓	✓	✓
Goal Tracking	✓	✓		✓	✓	✓
Statistics Panel	✓	✓	✓	✓	✓	✓
GPS Activity Tracking	✓		✓	✓	✓	✓
Achievements	✓					✓
Audio Cues	✓			✓	✓	✓

As the Table 7 demonstrates, RunTracker successfully provides the majority of the main features that build the essence of what a fitness tracker should provide to a user. The only aspect that RunTracker does not include is the concept of social media integration.

8.3 Future Improvements

Considering the personal appraisal, future improvements to the application would involve initially implementing a means for the user to sign up using a social media account, such as Facebook, Twitter or Google+. With a user's account connected to their social media, other features could be added in order to further enhance the user's experience with the application. One of the features that could be implemented with social media integration, is the ability for users to share the routes that they have completed whilst using the application. Furthermore, building upon this concept a system could be developed in order to allow users to upload routes so they can compete against themselves, or other users in their social circle. This would also require the implementation of a leaderboard system for each route.

As the results of user acceptance testing highlighted, it would be beneficial to implement a system that builds upon the current analytics panel. This would allow the user to view all of the activities that they have completed, instead of just displaying the activity statistics over the current week. This could be implemented by constructing cards for each activity, with the user route image, notes and relevant statistics applied so that the user can scroll through the cards in a chronological order.

Push notifications could also be added to this application, given the tools available in google-cloud and firebase. This could be implemented in a way in order to encourage the user to exercise if they have not logged into the application over the course of a few days. Furthermore if a user is close to reaching their weekly calorie goal, encouraging push messages could be provided to the user to further boost user engagement and help to improve the user's overall fitness.

Furthermore premium features could also be added to the application, such as working with personal trainers in order to develop food and fitness activity plans. This could be implemented as a means of connecting individual users to qualified personal trainers, allowing the application to act as a middle-man that collects a fee for providing business to those that work in the fitness industry.

The application developed still requires further work in order to be truly competitive in the current market place, however the system does provide the main functionality of what a fitness tracking application should achieve.

REFERENCES

[1] - J. Hicks, Polar: The Original Fitness Tracker and Heart Rate Monitor, 2016 [Online]. Available at: https://www.forbes.com/sites/jenniferhicks/2016/02/28/polar-the-original-fitness-tracker-and-heart-rate-monitor/#680ea7915fe9. [Accessed April 2019].

[2] - A. Knapp, How Garmin Mapped Out A New Direction With Fitness Wearables, 2016 [Online]. Available at: https://www.forbes.com/sites/alexknapp/2016/09/14/how-garmin-mapped-out-a-new-direction-with-fitness-wearables/#47c14c8a27b9. [Accessed April 2019].

[3] - B. Anderson, What is the Purpose of a Gantt Chart, 2016 [Online]. Available at: https://www.successfulprojects.com.au/purpose-of-a-gantt-chart/. [Accessed April 2019].

[4] - NHS., Understanding Calories, 2016 [Online]. Available at: https://www.nhs.uk/live-well/healthy-weight/understanding-calories/. [Accessed April 2019].

[5] - J. Ingels, R. Misra, J. Stewart, B. Lucke-Wold, S. Shawley-Brzoska., The Effect of Adherence to Dietary Tracking on Weight Loss: Using HLM to Model Weight Loss over Time, 2017, Journal of Diabetes Research, Vol. 2017,Article ID 6951495, page 8.

[6] - S. Gradari, A. Pallé, K. R. McGreevy, A. Fontán-Lozano, J. L. Trejo., Can Exercise Make You Smarter, Happier, and Have More Neurons? A Hormetic Pespective, 2016, Journal of Frontiers in Neuroscience, Vol. 10, page 93.

[7] - P. Trumbo, S. Schlicker, A. Yates, M. Poos, Dietary reference intakes for Energy, Carbohydrate, Fiber, Fat, Fatty acids, Cholesterol, Protein, and Amino Acids. Food and Nutrition Board of the Institution of Medicine, 2003. [Online]. Available at https://www.ncbi.nlm.nih.gov/pubmed/12449285. [Accessed April 2019].

[8] - C. S. Johnston., Uncle Sam's Diet Sensation: MyPyramid - An overview and Commentary. MedGenMed: Medscape General Medicine, Vol. 7, pp. 3-78. 2005.

[9] - B. E. Ainsworth, W. L. Haskell, S. D. Herrmann, N. Meckes, D. R. Basset Jr., C. Tudor-Locke, J. L. Greer, J. Vezina, M. C. Whitt-Glover, A. S. Leon., 2011 Compendium of Physical Activities: A Second Update of Codes and MET Values. Journal of Medicine, Science and Sports Exercise, Vol. 43, No. 8 pp. 1575-1591, 2011.

[10] - Strava., Strava, 2018 [Online]. Available at: https://www.Strava.com. [Accessed April 2019].

[11] - A. M. Williams, King of the Mountain: A Rapid Ethnography of Strava Cycling, 2013 [Online]. Available at: https://uclic.ucl.ac.uk/content/2-study/4-current-taught-course/1-distinction-projects/5-2013/williams-2012.pdf. [Accessed April 2019].

[12] - MyFitnessPal, MyFitnessPal, 2019 [Online]. Available at: https://www.myfitnesspal.com. [Accessed April 2019].

[13] - B. Laing, C. Mangione, C. Tseng, M. Long, E. Vaisberg, M. Mahida, M. Bholat, E. Glazier, D. Morisky, D. Bell., Effectiveness of a smartphone application compared to usual care in

overweight primary care patients: a randomised controlled trial, 2014 [Online]. Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4422872/. [Accessed April 2019].

[14] - MyPacer, MyPacer, 2018 [Online]. Available at: https://www.mypacer.com. [Accessed April 2019].

[15] - Runtastic, Runtastic, 2019 [Online]. Available at: https://www.runtastic.com. [Accessed April 2019].

[16] - C. Allison., The best fitness applications for your wearables, 2018 [Online]. Available at: https://www.wareable.com/sport/the-best-fitness-apps-2017. [Accessed April 2019].

[17] - A. Anton., Runtastic: An Excellent All-Rounder for Logging Fitness, British Journal of Sports Medicine, Vol 50, Issue 11 [Online]. Available at: https://bjsm.bmj.com/content/50/11/705. [Accessed April 2019].

[18] - S. Perez., Under Armour Snatches Up Health And Fitness Trackers Endomondo And MyFitnessPal, 2015 [Online]. Available at: https://techcrunch.com/2015/02/04/athletic-apparel-company-under-armour-snatches-up-health-and-fitness-trackers-endomondo-and-myfitnesspal/. [Accessed April 2019].

[19] - A. Purpura, Endomondo Sports Tracker Pro Review, 2015 [Online]. Available at: https://www.pocketgamer.com/articles/078860/endomondo-sports-tracker-pro-review/. [Accessed April 2019].

[20] - H. Atha., Cross Platform App Development Demystified - How to Make it Work for You, 2017 [Online]. Available at: https://www.moveoapps.com/blog/cross-platform-app-development-demystified/. [Accessed April 2019].

[21] - Shouten., A brief history of React Native, 2017 [Online]. Available at: https://medium.com/react-native-development/a-brief-history-of-react-native-aae11f4ca39. [Accessed April 2019].

[22] - Altexsoft., The Good and the Bad of ReactJS and React Native, 2018 [Online]. Available at: https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-reactjs-and-react-native/. [Accessed April 2019].

[23] - Altexsoft., The Good and the Bad of Xamarin Mobile Development, 2018 [Online]. Available at: https://www.altexsoft.com/blog/mobile/pros-and-cons-of-xamarin-vs-native/. [Accessed April 2019].

[24] - O. Gierszal., Development With Ionic Framework - Pros and Cons, 2018 [Online]. Available at: https://medium.com/appstronauts/development-with-ionic-framework-pros-and-cons-bca65ab24dae. [Accessed April 2019].

[25] - W. Danielsson., React Native application development - A comparison between native Android and React Native, 2016 [Onlne]. Available at: https://pdfs.semanticscholar.org/ab1e/c8f5934290757a1310fb19194cc06d164d74.pdf?_ga=2.28543164.1889635435.1539260386-1755174898.1539260386. [Accessed April 2019].

[26] - L. Jamie., Understanding User Authentication - 3 Basics You Should Know, 2017 [Online]. Available at: https://swoopnow.com/user-authentication/. [Accessed April 2019].

[27] - J. Boss., Five Reasons Why Goal Setting Will Improve Your Focus, 2017 [Online]. Available at: https://www.forbes.com/sites/jeffboss/2017/01/19/5-reasons-why-goal-setting-will-improve-your-focus/#1491a4b8534a. [Accessed April 2019].

[28] - Redux, Redux - A Predictable State Container for JavaScript Apps, 2015 [Online]. Available at: https://redux.js.org. [Accessed April 2019].

[29] - V. Thirugnanam., Using Firebase with Redux for Building a React App, 2017 [Online]. Available at: https://www.codementor.io/vijayst/using-firebase-with-redux-for-building-a-react-app-du1086puw. [Accessed April 2019].

[30] - Firebase, Firebase, 2019 [Online]. Available at: https://console.firebase.google.com. [Accessed April 2019].

[31] - R. Khanna., Structuring your Firebase Data correctly for a Complex App, 2016 [Online]. Available at: https://www.airpair.com/firebase/posts/structuring-your-firebase-data. [Accessed April 2019].

[32] - P. Aksonov., React Native Router Flux, 2019 [Online]. Available at: https://github.com/aksonov/react-native-router-flux. [Accessed April 2019].

[33] - J. Arvidsson., React Native Vector Icons, 2019 [Online] Available at: https://github.com/oblador/react-native-vector-icons. [Accessed April 2019].

[34] - Firebase., User Security, 2019 [Online]. Available at: https://firebase.google.com/docs/database/security/user-security. [Accessed April 2019].

[35] - A. Swain., bcrypt, 2019 [Online]. Available at: https://www.npmjs.com/package/bcrypt. [Accessed April 2019].

[36] - B. Magnuson., How to calculate your Body Mass Index or BMI, 1998 [Online]. Available at: http://extoxnet.orst.edu/faqs/dietcancer/web2/twohowto.html. [Accessed April 2019].

[37] - L. Richardson, J. Harband, C. Dro., React Native Maps, 2019 [Online]. Available at: https://github.com/react-native-community/react-native-maps. [Accessed April 2019].

[38] - S. Grider., Minutes, Seconds and Milliseconds, 2019 [Online]. Available at: https://www.npmjs.com/package/minutes-seconds-milliseconds. [Accessed April 2019].

[39] - A. Upadhyay., Haversine Formula - Calculate geographic distance on Earth, 2018 [Online]. Available at: https://www.igismap.com/haversine-formula-calculate-geographic-distance-earth/. [Accessed April 2019].

[40] - J. Lui, J. Beckwith., Google Cloud Text-to-Speech API: Node.js Client, 2019 [Online]. Available at: https://github.com/googleapis/nodejs-text-to-speech. [Accessed April 2019].

[41] - Software Testing Fundamentals., Unit Testing, 2019 [Online]. Available at: http://softwaretestingfundamentals.com/unit-testing/. [Accessed April 2019].

[42] - Software Testing Fundamentals., Integration Testing, 2019 [Online]. Available at: http://softwaretestingfundamentals.com/integration-testing/. [Accessed April 2019].

[43] - M. Setter, User Acceptance Testing - How To Do It Right!, 2013 [Online]. Available at: https://usersnap.com/blog/user-acceptance-testing-right/. [Accessed April 2019].

[44] - European Commission., 2018 Reform of EU Data Protection Rules, 2018 [Online]. Availabel at: https://ec.europa.eu/commission/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules_en. [Accessed April 2019].

[45] - Clarks CAS., GDPR - Personal Data: Transferring Data Outside Europe, 2018 [Online]. Available at: https://cas.ltd/gdpr-6-personal-data-transferring-data-outside-europe. [Accessed April 2019].

[46] - A. Dunlop., GDPR: Personal Data and Sensitive Personal Data, 2016 [Online]. Available at: https://www.burges-salmon.com/news-and-insight/legal-updates/gdpr-personal-data-and-sensitive-personal-data/. [Accessed April 2019].

[47] - Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications), Vol. 201, pp 37-47, 2002. [Online]. Available at: https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX%3A32002L0058. [Accessed April 2019].

[48] - D. Ding, M. Conti and A. Solanas, A smart health application and its related privacy issues, 2016 Smart City Security and Privacy Workshop (SCSP-W), Vienna, 2016, pp. 1-5. [Online]. Available at: https://ieeexplore.ieee.org/document/7509558. [Accessed April 2019].

[49] - Roke., Could fitness tracking solve murders, 2018 [Online]. Available at: https://gdpr.report/wp-content/uploads/2017/04/Could-fitness-trackers-solve-murders.pdf. [Accessed April 2019].

APPENDIX A: PROJECT INITIATION DOCUMENT (PID)

Individual Project (CS3IP16)

Department of Computer Science

University of Reading

Project Initiation Document

PID Sign-Off

Student No.	**22002003**
Student Name	**Rory John Fielding**
Email	**R.fielding@student.reading.ac.uk**
Degree programme (BSc CS/BSc IT)	**BSc CS**
Supervisor Name	**Lily Sun**
Supervisor Signature	[illegible]
Date	**2/10/2018**

SECTION 1 – General Information

Project Identification

1.1	**Project ID** (as in handbook)
	110
1.2	**Project Title**
	A Cross-Platform Fitness Tracker Application Developed In React-Native.
1.3	**Briefly describe the main purpose of the project in no more than 25 words**
	The purpose of the project is to develop a cross-platform mobile fitness tracking application with a database backend.

Student Identification

1.4	**Student Name(s), Course, Email address(s)** e.g. Anne Other, BSc CS, a.other@student.reading.ac.uk
	Rory John Fielding, BSc CS, R.Fielding@student.reading.ac.uk

Supervisor Identification

1.5	**Primary Supervisor Name, Email address** e.g. Prof Anne Other, a.other@reading.ac.uk
	Lily Sun, Lily.Sun@reading.ac.uk
1.6	**Secondary Supervisor Name, Email address** Only fill in this section if a secondary supervisor has been assigned to your project

SECTION 2 – Project Description

2.1	**Summarise the background research for the project in about 400 words. You must include references in this section but don't count them in the word count.**
	In order to initially conduct some market research into the development of a fitness application, I decided to personally test the top 5 fitness applications over a period of a week, analyse the current market space and contrast and compare the benefits of applications currently on the Apple App Store. These applications included 'Strava', 'MyFitnessPal', 'Pacer Pedometer & Step Tracker', 'Map My Fitness by Under Armour' and 'Runtastic'. For each application the differing types of fitness apps offered were analysed along with the application layout and structure, the data being gathered on users and potential backend calculations being implemented in order to build the application. I also created storyboards for each application in order to break down each app into a varying set of screens to be analysed individually. Having compared and contrasted applications currently available, I have decided on a set of features I would to include in the application being developed. In terms of application design, I have considered the pros and cons of building a native application in comparison to a cross-platform application created with a framework, such as Ionic, Xamarin and React-Native. In order to be competitive in the current market, applications should be developed to be cross-platform. This allows for a shorter development process, more code re-usability across different applications and a smaller development team. Furthermore there is also the potential of code re-use through creating a web-application alongside mobile. Through considering the different frameworks available, I have decided that React Native will be the most ideal framework to use for this project. It is completely free, uses JavaScript and has a very strong performance on mobile environments. The React Native architecture makes use of the GPU, while native platforms are more CPU intensive. This makes React Native apps super fast in comparison to other options. Furthermore React Native is modular and intuitive, similar to react, however it also boasts the advantage of having a 'live reload' feature which will enable the user to immediately see the result of the latest change made. This will help well when seeing changes made in order to drive a Test-Driven Development model for this project. References: Applications: https://www.strava.com/ https://www.myfitnesspal.com https://www.mypacer.com https://www.runtastic.com https://www.mapmyrun.com Sources: Eisenman, B., Learning React Native: Building Native Mobile Apps with JavaScript Holmes, E., Bray, T., Getting Started With React Native Who, Richard., React Native By Example Neil, T., Mobile Design Pattern Gallery https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-reactjs-and-react-native/ https://www.altexsoft.com/blog/engineering/xamarin-vs-react-native-vs-ionic-cross-platform-mobile-frameworks-comparison/ https://www.thewindowsclub.com/what-is-xamarin-and-cross-platform-mobile-development https://www.iflexion.com/blog/pros-cons-mobile-development-xamarin/ https://www.icapps.com/blog/5-key-advantages-react-native https://www.intertech.com/Blog/ionic-framework-review/
2.2	**Summarise the project objectives and outputs in about 400 words.** These objectives and outputs should appear as tasks, milestones and deliverables in your project plan. In general, an objective is something you can do and an output is something you produce – one leads to the other.

The overall main objective of the project is to develop a cross-platform fitness tracking application which communicates to a backend database.

This will initially comprise of a number of tasks:

- Application screen wire-frames
- User storyboard
- Use cases

Once the initial design of the application has been completed. The first milestone will have been reached whereby the application will have been designed completely. This will include a deliverable in which a design document will have been created for the project detailing the layout and structure of the application.

Following this, analysis will take place on all of the design work constructed in order to improve further upon the application design and re-organise any modules identified which may cause issues. Once analysis of the application design has been completed a second milestone will have been reached in the project. This will include creating a deliverable of a design analysis document detailing the design considerations made, how this can be optimised for scale and any security/privacy details that also are required to be considered.

Following this, various test cases listed below will need to be developed in order to ensure that any programming work completed can be tested according to the specification.

- Unit test cases
- Integration test cases
- User acceptance test cases

Once the test cases have been developed, a third milestone will have been reached in the project. This will include developing a deliverable of a test cases document which will include an exhaustive list of all of the test cases which will be used in the project in order to ensure that the application to be developed will meet all of the requirements that are neccessary.

Application development will now begin in order to create a proto-type of the initial fitness tracker application. This will include undertaking the following tasks:

- Front end application developed
- Backend application developed
- Proto-type testing
- Final application development

Once the tasks above have been completed, the application will be fully developed and completed. This will be the fourth milestone in the project. The fifth and last milestone in the project will be reached once the project report and demonstration have also be constructed and finished.

Initial project specification - list key features and functions of your finished project.
Remember that a specification should not usually propose the solution. For example, your project may require open source datasets so add that to the specification but don't state how that data-link will be achieved – that comes later.

The finished project should be a cross-platform fitness tracker application. This application will include the following features:
- Log in screen
- Account creation screen
- Data feed of previous workouts
- Ability to start a new workout which tracks user via GPS
- Screen while using the app which plots the route on a map via GPS using the Google Maps API.
- Audio cues to the user which occur at half mile/mile intervals while user is running
- Screen at the end of the workout which provides the user with information (Calories burned, time spent exercising, route taken on a map, elevation gain, average speed)
- Data analytics screen of work-outs
- Settings screen which allows the user to alter music and audio settings
- Log out screen
- Data should be sent/retrieved from a backend database
- Specifically, the application will focus on displaying data around the physical set features that can be recorded (heart rate, steps, etc.)
- The application will allow the user to set fitness goals
- The application will provide the user with push notifications if they have not worked out in a while, have not met their goal for the week and suggest new goals for the following week (10-15% increase). This encourages the user to improve their fitness and examine their overall performance increase to their quality of life.

2.4 **Describe the social, legal and ethical issues that apply to your project. Does your project require ethical approval? (If your project requires a questionnaire/interview for conducting research and/or collecting data, you will need to apply for an ethical approval)**

This application will be collecting a large amount of data on the users which interact with it and use it regularly. It is likely that the application will require the user to allow access to health data on their phone and also require the user to allow the application access to their location data in order to provide GPS tracking. As location based services will be used, the privacy of user's data is a key element in the social, legal and ethical issues of this project. Considering this, a workflow approach will be used known as privacy by design (PbD). PbD is an approach to development that takes privacy into account from the very beginning across all aspects of a project, rather than bolting it on at the end in 'Settings' for example. With the Eu's data-protection reform which comes into effect in 2018, this will make PbD requirements part of the law and thus this must be considered. The misuse of location data can have serious legal consequences. In order to account for this, the processing and retention of location data will be addressed in the terms and conditions when the user creates an account on the application. A robust contract should confirm that no uses of location data will violate the laws of the user's country. The user should be able to request to have their data deleted from the application at any time and all of their data should be kept secure and stored securely, perhaps with some form of encryption methodology in order to stay in accordance with GDPR regulations.

In Europe, location data is regulated by a law named the 'Directive on Privacy and Electronic Communications'. This has been implemented in all EU member states and therefore the application project will be developed in order to comply with the following points:
- Location data should only be used when it is anonymised
- If it is not anonymised, the user must give their consent for their location data to be used and accessed. Additionally, the user must be able to withdraw their consent at any time. This can be stated in the terms an conditions of the application, however can ho further by adding an option in the application to disable location data within the app's settings.
- The service provider should inform the user what location data will be collected and processed, what it will be processed for and whether it will be transmitted to any third parties, including social media websites. Any transmission of data to a third party must also be anonymised
- Users should be able to temporarily stop the use of location data, even if they have previously given consent.
- Location data should only be used for a specific reason, and only for the duration necessary for the purpose.

As I will only be utilising the application myself, I will not require any ethical approval for this project. Other users will be utilising the application briefly during user-acceptance testing, whereby they will by required to agree to sharing their location data in a signed document before undertaking the testing process.

2.5	**Identify and lists the items you expect to need to purchase for your project. Specify the cost (include VAT and shipping if known) of each item as well as the supplier.** e.g. item 1 name, supplier, cost
	No items required for purchase.
2.6	**State whether you need access to specific resources within the department or the University e.g. special devices and workshop**
	I will not require access to specific resources.

SECTION 3 – Project Plan

3.1	**Project Plan** Split your project work into sections/categories/phases and add tasks for each of these sections. It is likely that the high-level objectives you identified in section 2.2 become sections here. The outputs from section 2.2 should appear in the Outputs column here. Remember to include tasks for your project presentation, project demos, producing your poster, and writing up your report.

Task No.	**Task description**	**Effort (weeks)**	**Outputs**
1	**Background Research**	3	
1.1	Market Research & Testing	1	Market research report and testing analysis of current fitness applications
1.2	Application Development Research	2	Application development research
2	**Analysis and design**	7	
2.1	Application design	4	Wire-frames and mock-ups of the application. Contrasting designs and comparisons.
2.2	Application analysis	3	Devices targeted, System software targeted, Design review
3	**Develop prototype**	12	
3.1	Front-end development	6	Working front-end application which communicates to a back-end database with API calls.
3.2	Back-end development	6	Database created which communicates with a server in order to accept API requests that can alter a database, providing information and statistics on user interaction.
4	**Testing, evaluation/validation**	3	
4.1	Unit testing	1	Test cases
4.2	Integration testing	1	Test cases
4.3	User acceptance testing	1	Test cases
5	**Assessments**	4.5	

5.1	Project report	3	Project Report
5.2	Produce poster	0.5	Poster
5.3	Produce demo	1	Demo
TOTAL	**Sum of total effort in weeks**	**29.5**	

SECTION 4 - Time Plan for the proposed Project work

For each task identified in 3.1, please *shade* the weeks when you'll be working on that task. You should also mark target milestones, outputs and key decision points. To shade a cell in MS Word, move the mouse to the top left of cell until the curser becomes an arrow pointing up, left click to select the cell and then right click and select 'borders and shading'. Under the shading tab pick an appropriate grey colour and click ok.

	START DATE: 24/09/2018 **Project Weeks**												
Project stage	0-3	3-6	6-9	9-12	12-15	15-18	18-21	21-24	24-27	27-30	30-33	33-36	36-39
1 Background Research	▨	▨											
1.1 Market Research & Testing	▨	▨											
1.2 Application Development Research	▨	▨											
2 Analysis/Design		▨	▨	▨									
2.1 Application design		▨	▨										
2.2 Application analysis			▨	▨									
3 Develop prototype.				▨	▨	▨	▨	▨					
Front-end prototype				▨	▨	▨							
Back-end prototype						▨	▨	▨					
4 Testing, evaluation/ validation								▨	▨				
4.1 Unit testing								▨					
4.2 Integration testing								▨	▨				
4.3 User-acceptance testing									▨				
5 Assessments									▨	▨			
5.1 Project report		▨		▨		▨		▨	▨	▨			
5.2 Poster & Demo										▨			

RISK ASSESSMENT FORM

Assessment Reference No.		**Area or activity assessed:**	**Market/Background research process**
Assessment date	**1st October 2018**		
Persons who may be affected by the activity (i.e. are at risk)	**Rory John Fielding**		

SECTION 1: Identify Hazards - *Consider the activity or work area and identify if any of the hazards listed below are significant (tick the boxes that apply).*

	Fall of person (from work at height)			Lighting levels			Use of portable tools / equipment			Vehicles / driving at work			Hazardous fumes, chemicals, dust			Occupational stress	
	Fall of objects			Heating & ventilation			Fixed machinery or lifting equipment			Outdoor work / extreme weather	✓		Hazardous biological agent			Violence to staff / verbal assault	
	Slips, Trips & Housekeeping			Layout , storage, space, obstructions			Pressure vessels			Fieldtrips / field work	✓		Confined space / asphyxiation risk			Work with animals	
	Manual handling operations			Welfare facilities			Noise or Vibration			Radiation sources			Condition of Buildings & glazing			Lone working / work out of hours	✓
	Display screen equipment			Electrical Equipment			Fire hazards & flammable material			Work with lasers			Food preparation			Other(s) - specify	

SECTION 2: Risk Controls - *For each hazard identified in Section 1, complete Section 2.*

Hazard No.	Hazard Description	Existing controls to reduce risk	**Risk Level** (tick one)			Further action needed to reduce risks
			High	Med	Low	*(provide timescales and initials of person responsible)*
17.	Outdoor work/ Extreme weather	Weather conditions will be assessed prior to work in order to ensure that conditions are safe to conduct work.		✓		Excessive activity should not be attempted within a short period of time and will be limited to no more than 1 hour twice a day in order to prevent injury. RJF
18.	Field trips/ field work	Participant should not conduct field work in low light in order to be as visible as possible. Field tests should be carried out in as much of a safe environment as possible (e.g. Running in a park instead of along a main road)		✓		Participant should inform an emergency contact when undertaking field work. RJF
29.	Lone working/work out of hours	Ensuring permission is obtained for lone working/long hours from supervisor. Informing supervisor of health risks or concerns.			✓	An emergency contact can be provided to the supervisor for when lone working/our of hours work occurs. RJF
Name of Assessor(s)		Lily Sun	**SIGNED** [illegible]			
Review date		2/10/2018				

APPENDIX B: LOGBOOK

CS3IP16

Individual Project

Logbook

Date	Notes	References	Actions
24-28th September 2018	Began background market research regarding creating a fitness tracking application.	https://www.endomondo.com https://www.runtastic.com https://www.mypacer.com/ https://www.myfitnesspal.com https://www.strava.com/	Tested 5 applications: - Strava - MyFitnessPal - Pacer Pedometer & Step Tracker - Runtastic - Endomondo Sports Tracker
1st-5th October 2018	Researched cross platform frameworks for mobile application development	https://www.altexsoft.com/blog/engin eering/the-good-and-the-bad-of-reactjs-and-react-native/ https://www.altexsoft.com/blog/engin eering/xamarin-vs-react-native-vs- ionic-cross-platform-mobile-frameworks-comparison/ https://www.thewindowsclub.com/wh at-is-xamarin-and-cross-platform-mobile-development https://www.iflexion.com/blog/pros- cons-mobile-development-xamarin/	- Compared the advantages and disadvantages of using three different systems including Xamarin, React Native and Ionic and created initial document.
Supervisor:	[illegible]	Date:	18/10/2018

Date	Notes	References	Actions
8th-12th October 2018	Research react native application performance . Research tools for react native development . Research best practices for application development design.	https://pdfs.semanticscholar.org/ab1e/c8f5934290757a1310fb19194cc06d164d74.pdf?_ga=2.28543164.188963 5435.1539260386-1755174898.1539260386 https://www.codementor.io/kapilrawal/12-most-popular-react-native-development-tools-kf45eu4uh https://www.uxpin.com/studio/blog/gui de-mobile-app-design-best-practices- 2018-beyond/ https://careerfoundry.com/en/blog/ui- design/how-to-design-a-mobile-app-using-user-interface-design- principles/ https://medium.com/blueprint-by- intuit/native-mobile-app-design- overall-principles-and-common-patterns-26edee8ced10	- Conducted a performance evaluation regarding a comparison between react native applications and native android applications.
15th-19th October 2018	Construct a front-end application design.	Project Icons https://ionicframework.com/docs/ionic ons/	- Worked on storyboarding the front-end of the application
22nd-26th October 2018	Research back-end database management standards. Research into backend databases. Research version control.	https://www.ibm.com/support/knowledgecenter/en/SSEPH2_15.1.0/com.ibm.ims15.doc.dag/ims_dbsystemstds. htm https://www.service-architecture.com/articles/database/dbms_standards.html http://www.informit.com/articles/articl e.aspx?p=1963781&seqNum=4	- Worked on analysing data structures required from front-end screens to communicate with database. - Re-evaluate application design.
Supervisor	[illegible]	Date:	31/10/2018

Date	Notes	References	Actions
29th October - 2nd November 2018	Implementation of google firebase of user authentication.	https://www.npmjs.com/package/react-native-firebase https://firebase.googleblog.com/2016/ 01/the-beginners-guide-to-react- native-and_84.html	- Created wireframes for front-end screens - Implemented initial user registration screens for the application - Added navigation between screens and logic
5th-9th November 2018	Research into user permissions for camera, locations and health data for application use. Consider ethical, legal, social concerns of tracking location data.	https://docs.expo.io/versions/latests/sdk/permissions	- Set up GitHub project for front-end application - Created database table strcuture
12th-16th November 2018	Connect back-end database to front-end application. Create architecture view of the system.		- Created use case model for the project - Worked on front-end application design - Connected ack-end database
19th-23rd November 2018	Handle user data across multiple screens. Complete user registration to begin working on location tracking	http://www.reactjstutorial.net/props.html	- Completed the initial version of registration screen in application - Added tab bar and 5 different screens
26th-30th November 2018	Implement the activity screen for location tracking	Https://github.com/react-native-community/react-native-maps	- Incorporated react native maps into project in order to handle location tracking - Created activity panel, which handles map tracking, calculates distance, time, speed and calories burned over duration
Supervisor	[illegible]	Date:	30/11/2018

Date	Notes	References	Actions
2nd-7th December 2018	Implement activity screen re-design. Update literature review section.	http://www.healthfitnessrevolution.co m/top-10-benefits-of-fitness-trackers/ https://www.theguardian.com/science /2017/feb/21/health-apps-could-be-doing-more-harm-than-good-warn- scientists https://www.nhs.uk/conditions/nhs- health-check/tools-and-technology- that-can-help/ https://journals.plos.org/plosone/articl e?id=10.1371/journal.pone.0156164	- Re-design the activity screen to show a smaller map - Created critique as a tick box diagram against applications analysed
10th-14th December 2018	Complete second iteration of registration screens. Add logic to deal with password hashing.	https://auth0.com/blog/hashing- passwords-one-way-road-to-security/	- Re-design the goal screen in order to recommend appropriate goal based on BMI calculation - Implement text across application - Updated profile screen to display user information - Updated settings screen to match wireframe
14th-18th January 2019	Implement functions in order to post data to the database	https://firebase.google.com/docs/data base/web/read-and-write	- Allow the user to update and edit their profile - Allow the user to save individual activities and location data
21st-25th January 2019	Implement functions in order to retrieve profile and activity data from the database	https://firebase.google.com/docs/data base/web/lists-of-data	- Fix fetch profile error - Manage state across the user application in order to fix issues with fetching data from the database.

Date	Notes	References	Actions
28th January - 1st February 2019	Research into application state management	https://blog.bitsrc.io/react-context-api- a-replacement-for-redux-6e20790492b3?gi=d91d9c2e878d https://daveceddia.com/context-api- vs-redux/	- Implement Redux in order to effectively manage state
4th - 8th February 2019	Redesign of entire application utilising Redux	https://redux.js.org	- Implemented Redux
11th - 15th February 2019	Fix user authentication workflow by removing the access key from async storage. Update the user authentication workflow to include profile pictures		- Added user profile picture - Fixed user authentication workflow and redesigned authentication system
Supervisor:	[illegible]	Date:	20/02/2019
18th - 22nd February 2019	Analytics panel to be implemented in order to display user activity data		- Implemented analytics pane allowing user to view their completed activity data for the past week, detailing their distance, speed and kCal burned by date.
25th February - 1st March 2019	Discuss demonstration and presentation.		- Add tables and diagrams to presentation - Create architectural diagram detailing how the new components relate to each other - Created demonstration and presentation slides

Date	Notes	References	Actions
4th - 8th March 2019	Fixed stylisation errors across platform. The architectural diagram created needs to be updated in terms of scale and also to include the user in the diagram.		- Created critique of the different cross platform networks - Update architectural diagram in order to incorporate the user - Discussed report structure and how best to layout design and implementation sections - Updated demonstration and presentation time
Supervisor:	[illegible]	Date:	08/03/2019
11th - 15th March 2019	Added validation and error messages for each of the sign up steps. Completed first draft of project poster.	https://blog.testlodge.com/how-to-write-test-cases-for-software-with-sample/ https://geteasyqa.com/hot-to/how-to-write-test-cases	- Create test cases for sign up steps of the application. - Create test cases for MainTabNavigator
18th - 22nd March 2019	Write up report. Fix background colour differences between Android and iOS platforms.		- Provided the application to three different types user for user acceptance testing
25th - 29th March 2019	Continue working on report write up.		- Test the application thoroughly - Practice demonstration