

College of Engineering and Informatics,

National University of Ireland, Galway 2016

**Final Year Project Report**

**Project Title:**

Nutrition and training and recorder application

Submitted in partial fulfilment of the requirements for the BSc (Hons) Degree in

Computer Science & Information Technology

**Student Name Student ID Signature Date**

Nigel McIntyre 12393921

**Project Supervisor Signature Date**

Karen Young

Table of Contents

[1 Chapter 1 - Introduction 2](#_Toc447627243)

[1.1 Layout of Report 4](#_Toc447627244)

[2 Chapter 2 - Existing Diet and fitness tracking applications 5](#_Toc447627245)

[2.1 myfitnesspal 5](#_Toc447627246)

[2.2 LoseIt 6](#_Toc447627247)

[2.3 FitNotes 6](#_Toc447627248)

[2.4 Jetfit 7](#_Toc447627249)

[2.5 Review 7](#_Toc447627250)

[3 Chapter 3 – Technologies 9](#_Toc447627251)

[3.1 Android 9](#_Toc447627252)

[3.2 Java 12](#_Toc447627253)

[3.3 Android Studio 13](#_Toc447627254)

[3.4 Android SDK 14](#_Toc447627255)

[3.5 SQLite 15](#_Toc447627256)

[3.6 Justinmind prototyper 15](#_Toc447627257)

[4 Chapter 4 – Design 16](#_Toc447627258)

[4.1 Application structure 16](#_Toc447627259)

[4.2 Designing the layout 16](#_Toc447627260)

[4.2.1 Login 16](#_Toc447627261)

[4.2.2 Register 17](#_Toc447627262)

[4.2.3 Calculate Calories 18](#_Toc447627263)

[4.2.4 Homepage 18](#_Toc447627264)

[4.2.5 Add meal 19](#_Toc447627265)

[4.2.6 Add food 20](#_Toc447627270)

[4.2.7 Add new food/ edit food 21](#_Toc447627271)

[4.2.8 View past meals 22](#_Toc447627272)

[4.2.9 Add workout 23](#_Toc447627273)

[4.2.10 Add exercise 24](#_Toc447627274)

[4.2.11 View past workouts 25](#_Toc447627275)

[4.3 Designing the database 26](#_Toc447627276)

[5 Chapter 5 – Implementation 27](#_Toc447627277)

[5.1 Application Files 27](#_Toc447627278)

[5.2 Implementing the database 28](#_Toc447627279)

[5.3 Loging in/Creating a new user 29](#_Toc447627280)

[5.4 Add meal 30](#_Toc447627281)

[5.5 Add workout 32](#_Toc447627282)

[5.6 Main activity homepage 32](#_Toc447627283)

[5.7 SessionManager 34](#_Toc447627284)

[5.8 Options 34](#_Toc447627285)

[6 Chapter 6 Testing 35](#_Toc447627286)

[6.1 Adding a meal 35](#_Toc447627287)

[36](#_Toc447627288)

[36](#_Toc447627289)

[6.2 Adding a workout 36](#_Toc447627290)

[6.3 Adjusting remaining calories 37](#_Toc447627291)

[37](#_Toc447627292)

[7 Chapter 7 Evaluation 38](#_Toc447627293)

[7.1 Functional requirements 38](#_Toc447627294)

[7.2 Non functional requirements 39](#_Toc447627295)

[7.3 Project Scheduling 40](#_Toc447627296)

[7.4 Future work 42](#_Toc447627297)

[7.5 Conclusion 43](#_Toc447627298)

# Table of Figures

[Figure 1 myfitnesspal screen 10](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740488)

[Figure 2 LoseIt 11](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740489)

[Figure 3 FitNotes 12](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740490)

[Figure 4 JEFit 13](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740491)

[Figure 5 register screen 20](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740492)

[Figure 6 calculating calories screen 21](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740493)

[Figure 7 Add meal screen 22](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740494)

[Figure 8 add food screen 23](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740495)

[Figure 9 Add new food screen 24](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740496)

[Figure 10 View past meals 25](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740497)

[Figure 11 Add workout screen 26](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740498)

[Figure 12 Add excercise screen 27](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740499)

[Figure 13 View past workout screen 28](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740500)

[Figure 14 Android Architecture 31](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740501)

[Figure 15 Database entity relationship diagram 38](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740502)

[Figure 16 How TDEE is calculated 40](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740503)

[Figure 17 Creating new user use case diagram 41](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740504)

[Figure 18 Adding a meal 42](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740505)

[Figure 19 Adding a meal use case diagram 43](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740506)

[Figure 20 Adding a new workout use case diagram 44](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740507)

[Figure 21 View updated calories 45](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740508)

[Figure 22 Homepage 46](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740509)

[Figure 23 Options Button 47](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740510)

[Figure 24 Adding meal test 48](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740511)

[Figure 25 Adding food to meal test 49](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740512)

[Figure 26 Adding workout test 50](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740513)

[Figure 27 Adjusting calories test 50](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740514)

[Figure 28 Proposed Gantt chart 55](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740515)

[Figure 29 Actual Gantt chart 56](file:///C:\Users\Nigel\Desktop\FYP\report\second_draft.docx#_Toc447740516)

# Introduction

## Context

In recent years people have become more and more conscious of their health and physical appearance. Keeping fit and healthy is very important to ensure a good quality of life and can help prevent diseases such as obesity and heart disease. This can achieved by watching what you eat and by getting the right amount of physical exercise. To do this requires a certain level of control over your diet and exercise which can be done by keeping track of what you eat and how much, how often and in what way you exercise. Some people believe that they can get the weight or the physique they want by simply eating the type and amount of food they feel will allow them to achieve their goals and train without tracking their progress in the gym. This approach will work fine for a period of time but eventually your progress will stagnate and you will no longer see the changes you desire or feel you should have achieved.

In order to prevent this, and if you want your body to change or your health to improve, you need keep track of both your diet and training. Many people fall victim to the belief that it is enough to simply eat what they feel is right and to not keep track of their exercise and training. A way to avoid this is to write down what you eat and how you train in a notebook, this requires carrying around a notebook and pen everywhere you go which is impractical and could be easy to forget to write in it. Another more practical solution is to have an application on your phone to allow you to input the data and help you keep track of diet and training.

Since their invention smartphones have become a huge part of our daily lives. Today they can be used not only as a phone, but as a camera, map, music player, voice recorder, for gaming, for internet browsing and a wide range of other uses. The popularity of smartphones has led to creation of the growing industry of mobile application development. Mobile applications can extend the functionality of smartphones beyond what they were originally capable of.

As a result of this there are many mobile applications out there that allow users to record their diet and training such as myfitnesspal[[1]](#endnote-1), Nutrition Tracker[[2]](#endnote-2) and Lose it![[3]](#endnote-3). JetFit[[4]](#endnote-4) and BodyBuilding and fitness[[5]](#endnote-5) are both apps that allow users to track their workouts and training progress.

In my experience with using diet tracking applications I found that they did not have facilities to calculate how many calories you need to consume per day, which I feel is a shortfall of these applications as many users have no idea how many calories they should consume. Another service that I feel current diet tracking applications aren’t providing is the ability to set the amount of calories you would like from carbohydrates, fats and proteins. Knowing the correct amount of each you should be ingesting per day is vital to some diets and training programmes, and not having this facility in these applications could be seriously detrimental to the user’s progress in achieving their weight and fitness goals.

## Project goal

My goal is to create a mobile application that allows users to keep track of their diets and training all in one place. I will build an application that will allow users to calculate how many calories they need and how many calories they need from carbohydrates, fats and proteins. Users will be able to add foods to the application record their meals and the amount of calories and carbohydrates, fats and proteins are in their meals and be subtracted from their daily goal throughout the day and will be able to see their meals for previous days. Users will also be able to track their progress in the gym by recording their workouts, and will be able to view past workouts.

## Minimum Requirements

The minimum requirements for the project are to:

* Provide users with a method of calculating their daily calorific needs.
* Allow users to record their diet daily to keep track of the quantities and kind of foods they are eating while on the go.
* To provide users with a method of recording their training and exercise to help them keep track of their progress and keep motivated.

## Layout of Report

This report is laid out in the following chapters:

Chapter 2 researches the existing applications that provide a similar service to this project.

Chapter 3 analyses the existing applications similar to this project and shows how they influenced it. Chapter 3 also explains how the application was designed.

Chapter 4 explores the different technologies used to develop this project and how they wer implemented, along with how the use cases of the application were implemented.

Chapter 5 describes the different tests performed on the application and evaluates the results, chapter 5 also evaluates the application as a whole.

Chapter 6 is the conclusion of the report.

# Chapter 2 - Research

Diet and fitness tracking applications allow people to achieve their health and fitness goals through tracking their diets and training progress. Such applications provide a platform where the user can record their daily food intake and workouts to ensure that their diet and training will help them reach their goals.

## https://lh3.googleusercontent.com/WkGXEesGEu4ayZctjADlDKdscS3Ve1hmoVzhHQYh9NuK9QpDA3Rx6ae3jg7VH2dFZqk=w300myfitnesspal

myfitnesspal[[6]](#endnote-6) is diet recording website that allows users to record what they eat throughout the day. It was founded in 2005 and has since been purchased in 2015 by Under Armour, it is a free application with the option to upgrade to premium. It is the most popular of the diet tracking applications and has a website, Android and IOS application. Users track their diet by first entering a calorie goal, this is calculated by entering the weight you are and the weight you want to be and how long you would like it to take to get to that weight. From this mtyfitnesspal calculates how many calories a day the user would need to reach their goal in the preferred time. As they enter their meals throughout the day by adding food items through the barcode scanner or searching for a food item in their database of foods, the calories of each food is added to a total and subtracted from their goal.

myfitnesspal also facilitates recording exercise and can tell the user how many calories they used during exercise by the user entering length of time spent exercising, which is added to their daily total. Myfitnesspal also integrates with fitness monitoring devices such as Jawbone and FitBit to provide accurate calorie expenditure readings. Progress can be visualised using pie charts to see your macronutrient ratio of carbohydrates, fats and proteins each day and line graphs see how close you were to your goal each day.

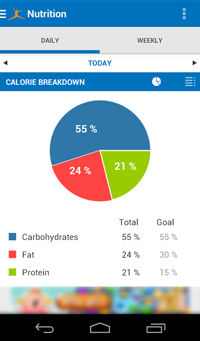
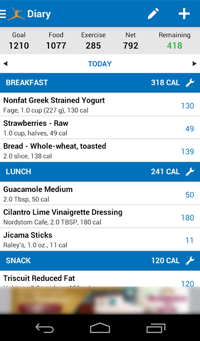


Figure myfitnesspal screen

## https://ascendapp.com/_images/loseit_icon.jpgLoseIt

LosieIt[[7]](#endnote-7) is another popular diet tracking application made by FitNow inc. It is a free digital health and fitness platform centred on the proven principals of calorie tracking to support weight management, founded in 2008. Users enter their weight, their weight goal and the time frame they wish to reach that goal in and their daily calorie intake is calculated. As users enter foods from a database their total calories consumed per day is calculated and subtracted from their daily goal. Users can join and online community with activities competitive challenges to help keep users motivated.

LoseIt can integrate with fitness monitoring devices and apps, such as FitBit, Google Fit and RunKeeper, to allow users to accurately track their calorie usage and adjust the amount of calories you need per day.



Figure LoseIt

## https://lh6.ggpht.com/mvY8S6uxR1AmEEt9nqPfqvnefM5JWe74YbblFkjPt7LDGHC62UD988-soAvu83qG6hA=w300FitNotes

FitNotes[[8]](#endnote-8) is a gym workout log to enable users to track their progress in the gym. Users can add exercises to a workout log and record their reps, sets and weight or distance and time for cardio, and has a rest timing feature to keep the users rest an appropriate length. Users can search through an exercise database to choose exercises for their workouts, while users can also enter their own exercises. FitNotes has a calendar view to see all of the days in which the user has trained, you can filter the calendar by only showing days trained a particular body part or when you did an exercise for a certain weight and number of reps or sets, which would provide the user with a good visibility of their progress.

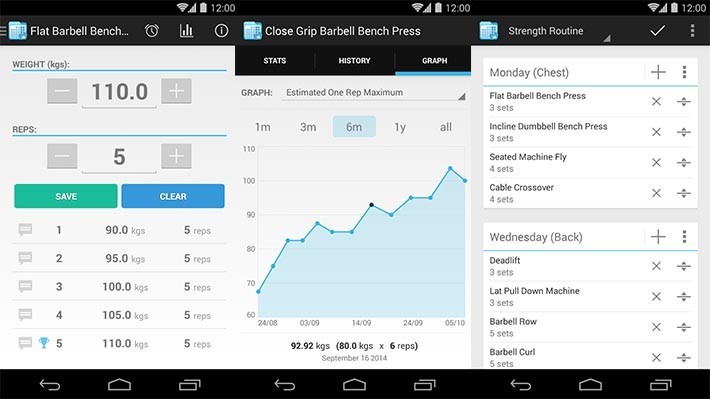


Figure FitNotes

## http://cdn.appstorm.net/android.appstorm.net/files/2011/07/JEFIT1.pngJefit

JeFit[[9]](#endnote-9) is another workout tracking application to make it easier for users to keep track of what they do in the gym. Users can record their workout by using a database of over 1300 exercises. Workout routines can be created by the user or searched for by goal such as if you were training for a sport, trying to lose weight or gain weight. JeFit have made it easy to input weight, repetitions and sets, as well as having a countdown timer for your rest to keep you on track with your workout.

Goals can be set for each exercise and progresses can be tracked as you enter in the exercise each time. Graphs and charts of your progress help the user to visualise improvements, and how far you are away from your goal. All of your personal data is synchronised to the server so you can access your data from any device. Users can help keep motivated by looking at other app users and friends’ progress as well as communicating with other app users. A summary of each workout can be viewed to view different stats from the workout such as duration, rest time and weight lifted.

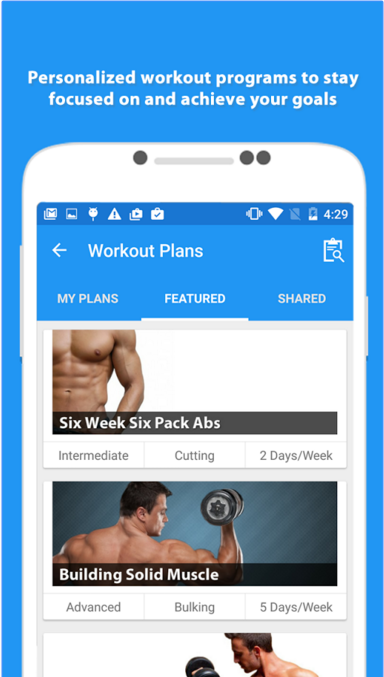
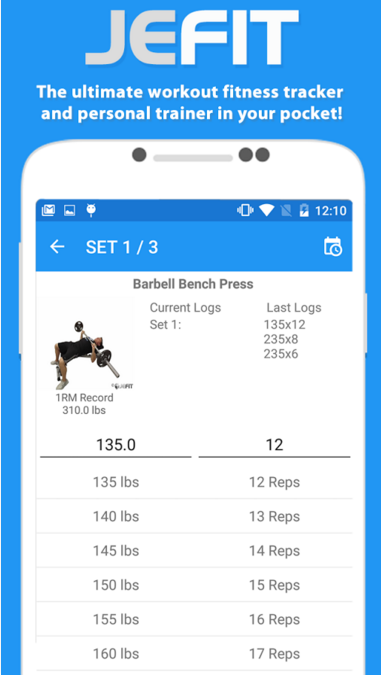
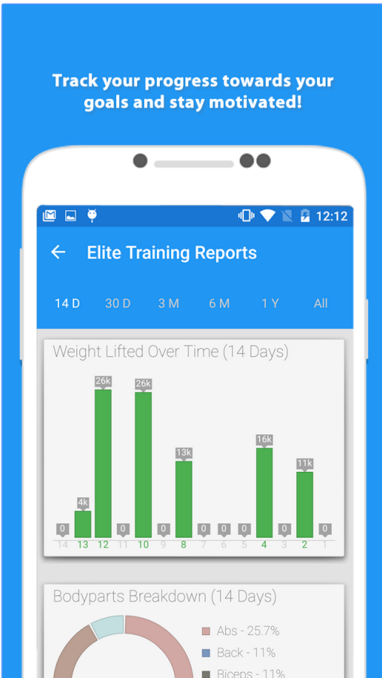


Figure JEFit

## Review

LoseIt and myfitnesspal provide and excellent platform to track your calories and reach your desired weight in a healthy and manageable manner. However I feel that they are restrictive in that they don’t allow for users to get specific with regards to the ratio of calories carbohydrates, fats and proteins. For people simply want to lose weight knowing how many calories they eat without knowing their nutritional breakdown can be enough, but if your goal isn’t just to lose weight, knowing and being able to specify the breakdown of your calories, and being able to see clearly how many calories from carbohydrates, fats and proteins can be very important. For example if you wished to gain weight, muscle mass and become stronger you would need to up the amount of calories from carbs and protein you wer eating. Or if you wanted to lose weight while retaining muscle mass you would need to lower you carbohydrate intake and increase your fat and protein intake. Although in these diet tracking applications you can see the how many carbohydrate, fats and proteins you have consumed, you cannot plan your diet with a specific goal in mind as you cannot set how many calories from carbohydrates, fats and proteins you have left to eat, only the calories you have left.

This is where my application will be different, by allowing the user to get specific and know exactly how many calories they have left to eat from carbohydrates, fats and protein. It will help users with specific goals to plan their diets around the ratio of calories they need from carbohydrates, fats and protein by allowing them to set the ratio themselves, and know before eating their meals what types of food they need to eat. My application will also calculate users calorific needs based on their BMR (Basal metabolic rate), which is how many calories your body burns at rest and by calculating your TDEE (Total Daily Energy Expenditure), which is how many calories your body needs taking into account how active you are in your everyday life and how often you exercise.

# Chapter 3 – Analysis and design

This chapter examines the various features and designs of the applications researched in the previous chapter. Examining the applications researched will allow for the extraction of the system requirements of the new application developed in this project.

|  |  |
| --- | --- |
| **Application** | **myfitnesspal** |
| **Features** | * Large Food database * Fast food entry * Connect to fitness devices * Progress reports * Customise goals * Track exercise |
| **Design** | * Light colour scheme * Easily navigate between screens * Intuitive layout * Information displayed where you would expect it to be |
| **Application** | **LoseIt** |
| **Features** | * Food and exercise database * Easy to use food entry and calorie counter * Customise weight goals * Fitness device connectivity * Connect with friends * Progress graphs |
| **Design** | * Bright colour scheme * Colourful graphs and icons * Some screens slightly cluttered with buttons and information * Graphs are colourful and easily interpreted * Intuitive layout, can swipe between screens |
| **Application** | FitNotes |
| **Features** | * Easily navigate through workout logs * Add exercise and record reps sets and weight or distance and time * Rest timer with sound and vibration * List of exercises by category * Create your own or use in built workout routines and exercises |
| **Design** | * Bright colour scheme * Well-structured layout * Colour coordinated Calendar screen allows to easily see days trained on * Pop-up dialog boxes helps reduce clutter on screen |
| **Application** | **JEFit** |
| **Features** | * Easily access workout logs * Large database of exercises * Pre-loaded workout routines * Online community of other app users * Provides personalized reports to track progress |
| **Design** | * Bright vibrant colour scheme * Well-structured layout allows for a lot of information on screen without seeming cluttered * Colourful graphs * Writing can be quite small and hard to see against background |

## Requirements gathering

### Functional requirements

Based on the analysis of the features of the other applications above, the functionality can be extracted that the project must, should, could and won’t have.

**What it must have:**

* Ability to record workouts.
* Ability to record meals.
* Show how many calories carbohydrates, fats and protein the user has left to consume.
* Save food items to be used again.
* Easy to use design and layout
* Ability to calculate the users required calories based on their goals.

**What it should have:**

* Graphs to visualise progress.
* Online database connectivity

**What it could have:**

* Alerts to remind the user to record their meals.
* Ability to share workouts and diets between users
* Graphs to visualise progress
* User profiles to share and view progress with friends.

### Design requirements

The design features that the application must haver are:

* Bright easy to read colour scheme
* Well-structured layout
* As little clutter as possible
* Pop-up dialog boxes to minimise clutter
* Easy learned, intuitive layout

### Non-functional requirements

Non-functional requirements are characteristics offered by a system as a whole, unlike functional requirements they do not apply to the individual features. The non-functional requirements for this project are usability, interoperability, reliability and flexibility.

**Usability –** The application should be easy to learn how to use and remember its layout. This means the user should easily make use of all of the applications functionality and navigate through the application. Items on screen should be comprehensive and explainable to provide a supportive environment to help the user as much a possible

**Interoperability –** The application should be compatible with as many devices as possible.

**Reliability –** The application should maintain all of its functionality over time and should execute without errors.

**Flexibility –** The project should be developed in such a way as to alow future development for either improvement or extension of its functionality.

## Design

### Application structure

From the gathered requirements and prior to designing each individual screen, a high level diagram of the general structure of the application was produced that best satisfies the requirements. In this diagram all the features of the application are outlined to give a general idea of how each feature would interact with the other. This diagram provided the foundation to which each other feature would be designed separately by giving me a better idea as to how the various screens of the application should flow from one screen to the next. Before each individual screen could be designed, the screens would be needed to provide the functionality required of them by the application had to be decided upon. Once the screens had been decided on, the general layout of each individual screen could be designed. A diagram of the application structure can be found in **Appendix A**

### Designing the layout

After looking at various design features of the applications researched at the beginning of this chapter and by reviewing what worked well in those applications, these are the designs produced that best follow the design requirements set out.

#### Login

The login/signup screen is the first screen the user is faced with. Here the user is asked to enter the login details or register and account with the app. This screen consists of two textboxes to allow the user to enter an email and a password. If the user has not yet registered and account with the app they would click the signup button which would then bring them to a registration screen. Otherwise the user would enter their log in details and click the login button and be brought to the main home screen of the application.

#### Register

The register screen allows the user to enter their details to create a new account with the application. The register screen consists of four textboxes, one for first name, last name, email, password and password verification.

Figure register screen

First Name

Last Name

Email

Password

Retype Password

Register

#### Calculate Calories

The calculate calories screen is where the user enters their age, height and weight to calculate the number of calories their body needs per day. This screen is scrollable to allow for space for all of it components. It has five text boxes for users to enter their height, weight age, and the percentage of calories the user wants from carbohydrates, fats and protein. There are three separate radio button groups, the first one is enter the users sex, another to enter how physically active the user is and the last, is for the user to enter whether they wish to gain weight, lose weight or maintain it.

Calculate your daily calories

Please enter your:

Age

Height in cm

Weight in kg

Female

Male

Calculate BMR

Very active

Moderately active

Extremely active

Please enter your goal

Loose weight

Maintain weight

Gain weight

Please enter percentage of calries from carbs, fats and protein

Carbs Fats Protein

Activity Level

Sedentary

Lightly active

Calculate TDEE

Figure calculating calories screen

#### Homepage

The Homepage is the main page of the application. On this screen the users remaining calories, carbs, fats and proteins are displayed. From the homepage screen the user can navigate to the add meal screen and add workout screen via the add meal and add workout buttons.

#### Add meal

The add meal screen is where the user creates a new meal. The user enters the name of the meal in the textbox and to add a new meal the “ADD” button is clicked, which leads the user to the add food screen. A list at the bottom of the screen displays all the meals for that day. To edit a meal the user clicks on the meal, and to view meals for previous days, the “view past meals” button is clicked. To delete a meal for the day the meal name is entered into the delete meal textbox and the “Delete” button is pressed.

## 

Please enter a meal name:

## 

Meal name

ADD

Delete meal

Delete

View past meals

## 

Meal 1 397cal 78c 5f 10p

Meal 2 560cal 80c 10f 25p

Figure Add meal screen

#### Add food

In order to add a new food to the meal, the user navigates to the add food page from the add meal page. Here the user can add a new food to the meal by either adding a new food by clicking the “add” button, edit or delete a food by clicking the “edit/delete” or search for foods the food name can be entered in the search foods textbox and clicking “search”. The current foods already in the meal would be displayed in a list at the bottom of the page. Above this the current total of calories, carbs, fats and protein for the meal can be seen. To delete a food item from that particular meal, the user can click on the food item to be deleted and it is removed from the meal and the calories, carbohydrates, fats and proteins are subtracted from the meal total.

Meal1

Add new food

Figure add food screen

ADD

Edit/delete

Total for meal:

Cals Carbs Fats Protein

460 80 12 26

rice

chicken

Search food

Search

#### Add new food/ edit food

On this screen the user can create a new food and edit existing foods. The same screen layout is used for both but are navigated to differently. The screen consists of four text boxes to enter the food name, carbs, fats, and protein. To add/update the food the add food button is clicked and the food is added/updated.

Add Food

Protein

Fats

Please enter:

Figure Add new food screen

Food name

Carbohydrates

#### View past meals

The view past meals screen enables the user to search previously entered meals by date. To search the meals eaten on a particular day the user enters the date in the date textbox and clicks the search button. A list of all the meals entered that day is shown at the bottom of the page.

Search

date

View past meals

Meal 1 397cal 78c 5f 10p

Meal 2 560cal 80c 10f 25p

Meals 3 600cal 90c 12f 26p

Figure View past meals

#### Add workout

To add a new workout the users navigates to the add workout screen from the homepage. This screen allows the user to add anew workout by typing the workout name into the textbox and clicking the “ADD” button. Workouts for the last seven days are displayed below and can be viewed and edited by selecting the name. If the user wishes to view an older workout the “view past workouts” button is pressed.

ADD workout

date

ADD

View past workouts

Workout Date

Chest and back 03/13/2016

Shoulders 03/14/2016

Legs 03/14/2016

Figure Add workout screen

#### Add exercise

The user can add exercises to a workout in the add exercise screen. In this screen the exercise name is set in the exercise text box and an exercise is added to the workout when the “Add exercise” button is clicked. A pop up dialogue would appear once the “add exercise” button is clicked to allow users to enter the weight, repetitions and sets, the user can select “ok” or “cancel”. Once added to the workout the exercise appears in a list at the bottom of the screen along with the reps sets and weight.

ADD

excercise

Exercise sets reps weight

Curls 4 10 20

Extensions 4 10 35

Cancel

reps

Please enter reps:

Please enter sets:

Ok

sets

Please enter weight:

weight

Arms

Figure Add excercise screen

#### View past workouts

To view past workouts earlier than 7 days ago, from the add workout page the user can navigate to the view past workouts page. On this page the user views previous workouts by entering the date of the workout when it was entered into the date textbox and the search button is clicked. The workouts entered on the date searched appear at the bottom of the screen along with the exercises performed and the sets, reps and weight.

Exercise sets reps weight

Curls 4 10 20

Extensions 4 10 35

View past workouts

date

Search

Figure View past workout screen

# Chapter 4 – Technologies and implementation

In this chapter the technologies used and how they have been implemented is discussed.

## Technologies

This section goes through in detail all of the technologies used to develop the application.



### Android

Android id an open source Linux-based operating system designed by google for mobile and tablet devices[[10]](#endnote-10). The Android Software Development Kit (Android SDK) is an open source project, which is freely available on the internet from the android website[[11]](#endnote-11) and can be freely customised by companies and developers. As a result of being open source a large online community has since developed, where support is provided to help in the development of applications for this operating system.

As shown in figure 14, the architecture is layered and divided into five layers: The Linux Kernel, Android Libraries, Android runtime, the Application Framework and on the top the Android Applications.

**Linux Kernel –** Provides a lever of abstraction between the device hardware and it contains all the essential hardware drivers like camera, keypad etc. it also handles things such as networking and device drivers.

**Libraries –** On top of the Linux Kernel is a set of Android libraries written in C++/C, which are specific for different hardware. These libraries include ones such as WebKit, an open source browser engine, SQLite database, which is a useful repository for storage and sharing of application data and libraries to play and record audio and video.

**Android Runtime –** Consists of a set of libraries required by the Java core libraries allow application developers write applications in Java. This section provides a key component called Dalvik Virtual Machine(DVM) which is a Java virtual machine designed and optimized for android. The DVM enables every android application to run on its own process, with its own instance of the DVM.

**Application Framework –** The application framework layer provide higher level services to applications in the form of Java classes and developers can use these classes to make use of these services in their applications. The framework provides key services such as, Activity manager, which controls the application lifecycle, Content providers that allow applications to publish and share data, Resource manager that provides access to resources such as strings, user interface layouts etc. Notifications manager which allows applications to display alerts and View system which is an extensible set of views used to create applications users interfaces.

**Applications layer –** At the top of androids architecture is the application layer. This is where you find all the applications used by the user. It can include applications such as messaging, email, games, maps, browsers etc. This layer provides a user interface with which the user can interact with the operating system.



Figure Android Architecture

**Why choose Android over other mobile operating systems?**

When choosing which operating system to develop the project in, a few things had to be taken into account. One was which programming language would be preferred to write the project in. Android is chiefly developed in Java which has been widely used for more than twenty years and is supported by a vast amount of resources which can be freely accessed online. Apple IOS is developed in Objective-C, which is a language used by apple with limited online resources.

Java is a language I am comfortable with and developing and IOS application would require learning a new language. Given the limited timeframe to complete this project there would not have been enough time to learn a new language and complete the project. Another factor that needed to be taken into account was the fact that for developing IOS applications you need a Mac to run Xcode which is the Integrated Development Environment (IDE) used to develop IOS applications. Android studio, which is the IDE used to develop android applications is available for Mac and Windows.

Taking these factors into consideration, for this project it was concluded that Android was the best option. Considering the availability of online support for android development it reduced the possibility of reaching a dead end in the project, meaning that I would be able to complete this project to the best of my ability.



### Java

Java is the main language used to develop android applications, and the language I have chosen to develop my project. Java is an Object Oriented language and is one of the most popular programming language in the world and is primarily used to develop client and server applications. The introduction of MIDlets, applications that use the Mobile Information Device Profile (MIDP) which are executed on a Java Virtual machine (JVM) allows the abstraction of hardware on the on the smartphone and allows developers to build applications that run on devices that support Java runtime. As a result of this Java has become the most popular programming language for developing android applications.

Due to its popularity, there are many online resources for developing android in Java, such as tutorials such as tutorialspoint[[12]](#endnote-12) and answers to common problems encountered by developers on websites such as Stackoverflow[[13]](#endnote-13). The internet is a great resource for support and exchange of ideas between the developers. The vast quantity of online support for android development in Java coupled with the fact that Java is taught as part of my course, led me to choose Java as the language I would develop my android application in.

### http://stech4.firstpost.com/tech2images/640x359/proportional/jpeg/2014/12/Android-Studio-624x351.jpgAndroid Studio

Is the official IDE for android development[[14]](#endnote-14), which is freely available under the Apache License 2.0 and is the IDE used for the development of this project. The first stable release of the IDE, version 1.0 was released in December 2014. The IDE is based on the JertBrains IntelliJ IDEA software[[15]](#endnote-15) and is specifically designed for android development and can run on Windows, Mac OS X and Linux. Eclipse Android Development tools were replaced by Android studio as Googles primary IDE for native Android application development. Android studio includes many useful features designed to make your experience developing as productive and enjoyable as possible. These features include:

**Gradle[[16]](#endnote-16)-based build support –** Gradle is a build automation system to support multi-project builds and support incremental builds.

**Android-specific refactoring and quick fixes**

**Lint[[17]](#endnote-17) tools –** which are used to catch performance, usability, version compatibility and other problems.

**ProGuard[[18]](#endnote-18) integration and app signing capabilities-** ProGuard is a tool that shrinks, optimizes and obfuscates Java code.

**Temple-based wizards –** To create common Android designs and components.

**A rich Layout editor –** This allows users to drag and drop UI components while previewing the results on multiple screen configurations.

**Support for building android wear applications**

**Support for Google Cloud platform –** This built in support allows integration with Google Cloud messaging and App Engine.

Alternatively to using Android studio Eclipse could have been used, which was originally the official IDE for android development before android studio. Eclipse was enabled to develop android applications using the Android Development Tools (ADT) plugin. Netbeans also supports the development of android applications through plugins.

### Android SDK

Developing an android application requires the appropriate API libraries and tools. Google has made these available to download in the Android Software Development Kit[[19]](#endnote-19) (SDK) which are fundamental to developing and android application. Included in these libraries and tools are a debugger, handset emulator to allow the developer to run test on a computer, documentation, sample code and tutorials. The SDK provides support for older versions of android in case developers want to target their applications at older devices.

### https://upload.wikimedia.org/wikipedia/commons/thumb/3/38/SQLite370.svg/2000px-SQLite370.svg.pngSQLite

Is the database used by Android, SQLite[[20]](#endnote-20) is a relational database management system contained in a C programming language. Unlike other database management systems, SQLite is not a client-server database engine, instead it is embedded into the end program. SQLite is used as an embedded database software for local/client storage in application software for web browsers, and operating systems. This was used as the local database for the application in this project.

### https://www.cooper.com/api/uploads/prototyping_tools/logo_justinmind.pngJustinmind prototyper

Justinmind[[21]](#endnote-21) Prototyper is a prototyping tool used to produce web and mobile applications wireframe prototypes. Protoypes are produced by dragging and dropping widgets onto the canvas. Justinmind allows for the protoypes to be simulated and users can navigate through the app and get a feel of what the finished app would be like.

### https://upload.wikimedia.org/wikipedia/en/thumb/6/62/MySQL.svg/1280px-MySQL.svg.pngmySQL

Is a relational database management system used for the development of web applications. It is used in this application to allow the connection of the application to the Linux danu6 server, where a database was set up during my course at NUIG. This database is accessed using MySQL queries to push and pull data from and to the application.

## Implementation

This section explains how the different technologies and features of the application were implemented. It will also go through each use case and explain how they have been implemented.

### Android Application Files

Before examining the code written to produce the application, it is necessary to look at how an android application is commonly structured. There are certain components that are necessary to developing an android application for the android operating system. All android application vary in functionality, size, complexity and specifications but they all consist of the same structure which rely on five components.

**AndroidManifest.xml –** This important part of the application can be described as the application descriptor. In this file, the package name, which serves as the unique identifier of the application, is set. This file describes the activities and names the classes that implement of each component and under what conditions they can be launched. It also declares the theme of the application and the minimum API required to run the application[[22]](#endnote-22).

**Activities –** The User Interface of a screen in the applications is represented as an activity. Each activity has a corresponding Java class to give the activity is functionality and an xml layout file to determine its UI[[23]](#endnote-23).

**Intents –** Intents are abstract descriptions of an operations to be performed and is basically passive data structure holding abstract descriptions of an actions to be performed. startActivity() can be used to launch and activity, intents can also be used to communicate between activities and background services[[24]](#endnote-24).

**Resource files** – Are files containing the different resources needed by the applications apart from the source code. These include the layout xml files used for the UI, images, icons and fonts.

### Class Diagram

Before the development of the application, a class diagram was produced to provide guidance during the implementation. This diagram helped to clarify the different classes that would be required to develop the application before during development. The diagram consist of a class for each activity in the application which controls its functionality. Along with the activity classes there are classes for all the objects needed in the application and classes that allow interaction with the application to the database. The class diagram can be seen in **Appendix B.**

### Designing the database

Before implementing the application and after the functional requirements had been set, the database needed to be designed to make the relationships between the tables needed clear in order to implement the SQLite database. Below is the Entity Relationship Diagram (ERD) that was produced, which shows the relationships between each of the tables in the database. As can be seen in the ERD the users table has a one to many relationship with the meals table. To prevent a many to many relationship between the meals and food table, the mealEnt table was added. Each row in the mealEnt table contains a food for particular meal on a particular day that belongs to a user, and has a one to one relationship with the foods table. The Users table has a one to many relationship with the workout table, this table has a one to many relationship with exEntry, where each row contains an exercise for a particular workout, date and user and has a one to one relationship with exercise.

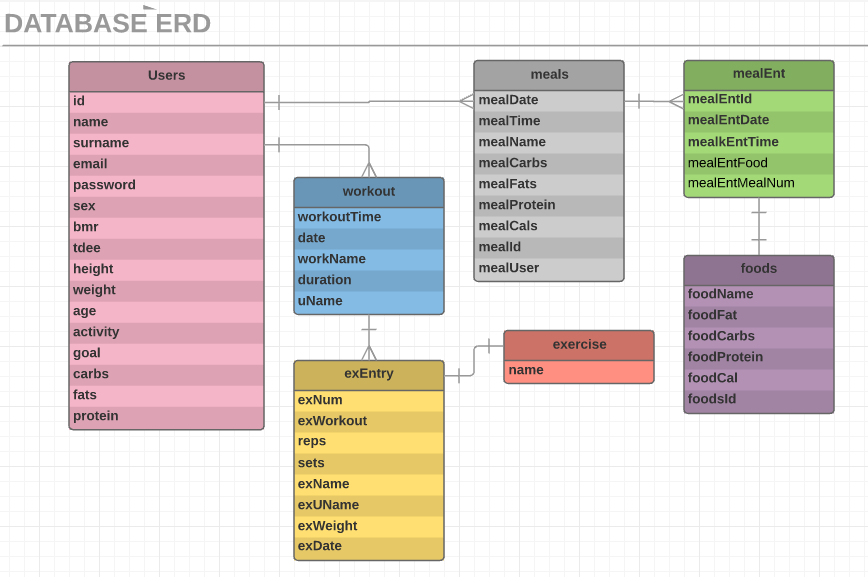


Figure Database entity relationship diagram

### Implementing the database

To create the SQLite database, the DbHelper.java class was created. This class extends SQLiteOpenHelper and contains an instance of SQLiteDatabase, which allows operations to be performed on the SQLite database. This class creates the database, the first time the application is run by executing SQL “create table” queries in the onCreate method. These queries were declared in the userDbAdapter.java, exerciseDbAdapter.java and foodEntryDbAdapter.java classes. Within these classes, the tables and their columns are declared along with methods to perform more specific operations on each of the tables.

public void onCreate(SQLiteDatabase db){  
 db.execSQL(userDbAdapter.*TABLE\_CREATE*);

Within the foodEntryDbAdapter.java class, there is an instance of SQLiteDatabase and the DbHelper.java class, which are needed to perform Create Read Update Delete (CRUD) operations on the database. The foodEntryDbAdapter consists of methods to perform these operations on the meals, mealEnt and food tables where needed. In the exerciseDbAdapter.java class are methods which provide similar methods to perform the CRUD operations on the exercise, exEntry and workout tables and again in the userDbAdapter.java class only on the users table.

db.update(*TABLE\_NAME*, values, "email = ?", new String[]{u.getEmail()});

db.delete(*MEALS\_NAME*, *MEALS\_MEALNAME* + "= ? AND "+*MEALS\_DATE*+" = ?", new String[]{String.*valueOf*(mealName),String.*valueOf*(todaysDate

### Implementing online database

To back up the data in an online database in the application, the MySQL danu6 database is used. Here all of the data which is store locally on the device using the SQLite database can also be stored in the MySQL database, to back up the data. Through phpMyAdmin, the tables in the database of the application were set up ready for data to be inserted and queried. To give access to the database from the application, the onlineDatabase.java class was created. Importing java.sql allowed access to the online database using a connection string and various SQL statements for performing read, update and delete operations on each of the tables in the online database. At each point in the application where and operation is performed on the locally stored SQLite database, a method from the onlineDatabse.java class is called to perform the required operation on the specific table in the MySQL database on the danu6 server. This allows for the information that would normally be stored on the local database to also be backed up on the MySQL database.

### Implementation of use case

This section goes through each use case and how it has been implemented.

#### Loging in/Creating a new user

The log in activity is controlled by the Login.java class and UI is in the activity\_login.xml file. To login the user must enter their log in details, which are checked to be in the database in the onButtonClick method in the Login.java class. If the log in details are valid the user is sent to the Homepage, otherwise they must either check their login details or create a new user by clicking the signup button which directs the user to the register activity.

Once the user enters their details and selects the register button, a new user is created in the database by creating an instance of the userDbAdapter class and calling the insertUser method and the calculateBMR activity is started. This activity is where the user calculates their daily calorific requirements based on various aspects of their daily lives and physical measurements. The users Basal Metabolic Rate, which is the number of calories the body needs to function per day at rest, is calculated using the Harris-Benedict[[25]](#endnote-25) equation, which consists of two equations, one for males and one for females.

if(sexstr.equals("Male")) {  
 BMR = ((66) + (13.7 \* weightstr) + (5 \* heightstr) - (6.8 \* agestr));  
 TDEE = BMR \* activityLevel;  
}  
else{  
 BMR=(655+(9.6\*weightstr)+(1.8\*heightstr)-(4.7\*agestr));  
 TDEE = BMR\*activityLevel;  
}

Once the BMR is calculated, this value is used to calculate the users Total Daily Energy Expendidture (TDEE), which is the amount of calories the body needs per day after taking into account the activity level of the user. The BMR is simply multiplied by a number corresponding to the activity level of the user to determine their TDEE[[26]](#endnote-26).



Figure How TDEE is calculated

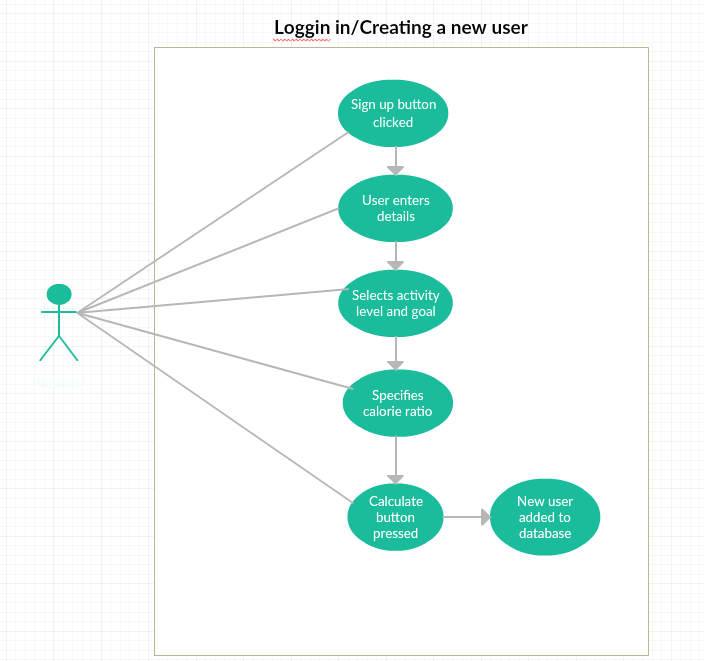
Based on the goal selected by the user, either to lose, maintain or gain weight a percentage of your TDEE is taken away (20% is recommended)[[27]](#endnote-27), left the same or added on (10% is recommended)[[28]](#endnote-28) respectively. To add these values to the users profile the updateCalUser method is called.

Figure Creating new user use case diagram

#### Add meal

In the addMeal activity the onCreate method is called when the activity is started. All the meals for the current day are displayed using the getAllMeals method which queries the database for meals with that days date and current user. When the user enters the meal name and the presses the add button, the insertMeal method is called, a new meal is added to the database and the addFood activity is started. The onCreate method in the addFood class, queries the database to display all the current food items in the meal by calling the getAllFoodsForMeal method, which takes in two parameters, the current date and current meal. In this activity the user can add a new food to the meal by either searching for a food already in the database using the getFoodInfo method, which takes in the foodName as a parameter and adds a new meal entry by calling the insertMealEnt that takes a MealEnt object as a parameter, or by adding a new food item to the database in the addNewFood activity. In both cases a pop up window appears to allow the user to input the quantity using LayoutInflater and AlertDialog classes imported from android.view library.

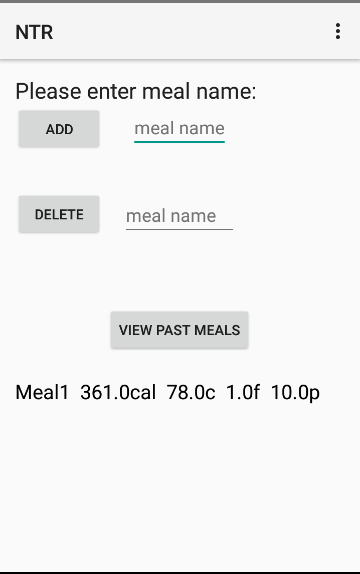
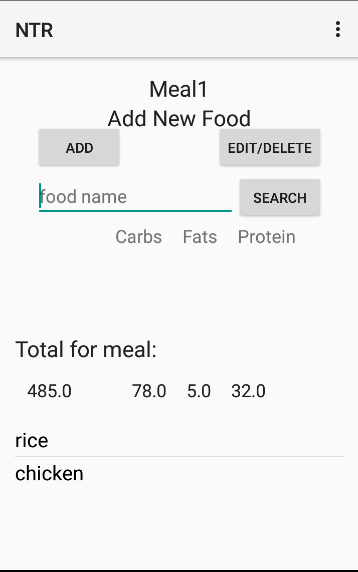


Figure Adding a meal

To edit/delete a food item the user selects the edit/delete button, another popup dialog box appears asking the user to enter the food name and whether they wish to edit or delete the food. If the edit button is selected they are directed to the edit food activity where the updateFood method is called to update the food in the database, which the food object as a parameter. If the delete button is selected the deleteFood method is called and the food is removed from the database using the foodName as a prameter. If the user wishes to delete a food from a meal, the food item is selected from the ListView at the bottom of the page and the calories, carbohydrates, fats and proteins of that food are subtracted from the meal and the meal is updated by calling the updateMeal method. To view meals from a particular day, the viewPastMeals activity can be started from the addMeal activity. This activity allows users to search meals by day by calling the getAllMeals method which takes in two parameters, the date entered by the user and the email of the current user and queries the database for corresponding meals.

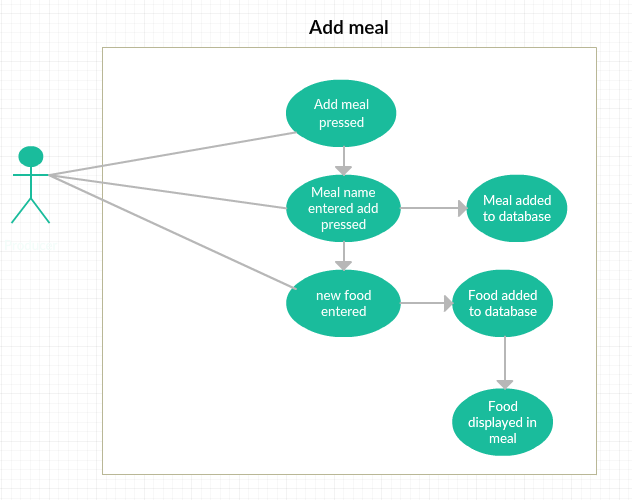


Figure Adding a meal use case diagram

#### Add workout

To add a new workout the user must enter the workout name into the textbox in the addWorkout activity. This calls the insertWorkout method, taking the Workout object as a parameter. The onCreate method of the addWorkout activity calls the getAllWorkout method, which queries the database for all workouts recorded in the last seven days, taking the current users email and current date as parameters.

Adding exercises to a workout is done in the addExercise activity, where the user enters a name of the exercise and clicks add. A dialog box appears allowing the user to enter the number of repetitions, sets and weight used for the exercise. Once “ok” is pressed the insertExEntry method is called, taking the exEntry object as a parameter. To delete an exercise from a workout the exercise is pressed and the deleteExEntry meathod is called which takes four parameters, workoutName, todaysDate, user email and exercise name. Previous workouts are viewed in a similar manner to viewing the past meals using the getAllWorkout method also used in the addMeal activity.

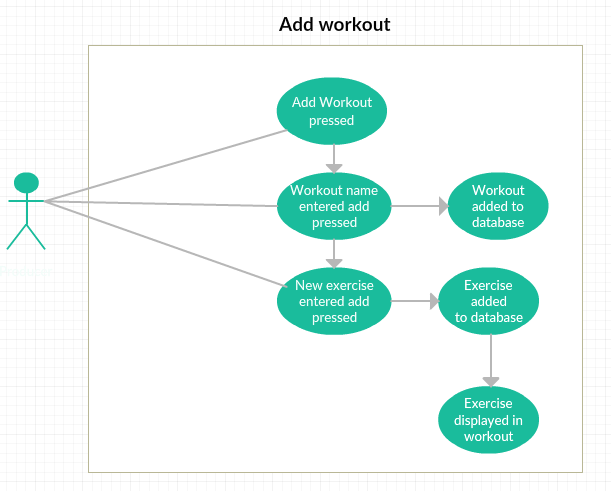


Figure Adding a new workout use case diagram

#### Calories being updated

This is the main screen in the application, it shows the amount of calories left for the user to consume, as well as the number of grams of carbohydrates, fats and protein left. The onCreate method which is called when the activity starts, gets all the goal calories, carbohydrates, fats and proteins the user has using the getInfo meathod which takes the email of the user as a parameter. The calories, carbohydrates, fats and proteins from the current meals entered by the user are subtracted from the goal meals and what remains is displayed. The getAllMeals method which uses current date and the users email as parameters, and the getMelaInfo method gets the nutritional value from each individual meal, using the mealName, todays date and user email to gather the information.

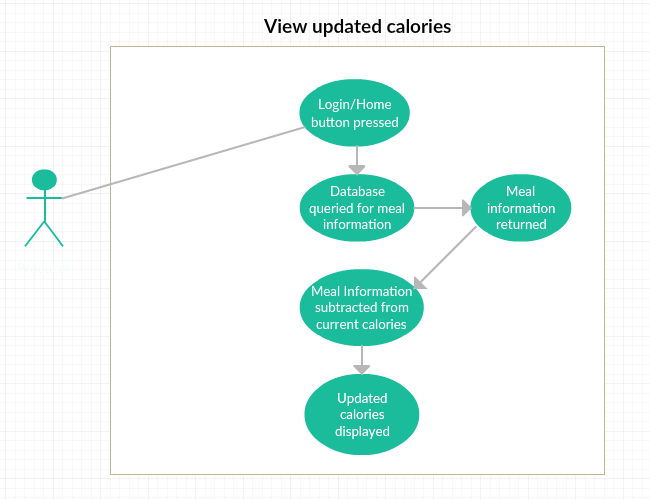


Figure View updated calories

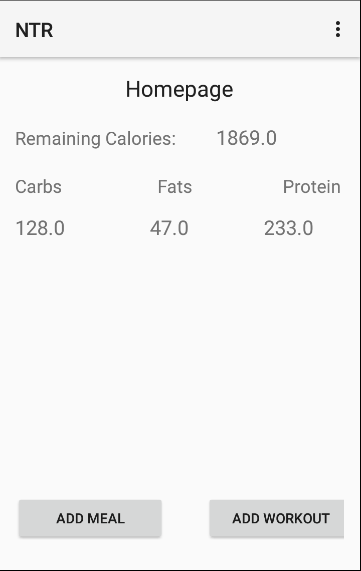


Figure Homepage

### SessionManager

At each point the users email was used as a parameter to a method to perform an operation on the database. During implementation, Intents were used to carry the email from when it was entered at login throughout the application. This is a very bad way of keeping track of sessions and so the session manager class was created to improve how sessions wer handled. In each activity an instance of the SessionManager class is used to obtain the email. This class enables the user to stay logged into the application, and view all of the users personal meals and workouts until the applications is shut down or the user logs out. The SharePreferences class imported from the android.content library and allows for data to be shared amongst all of the activities allowing the users email to be accessed from all of the activities without the uses of intents.

SessionManager session;

session = new SessionManager(getApplicationContext());

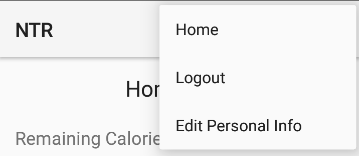
String email = user.get(SessionManager.*KEY\_EMAIL*);

### Options

In the early stages of the development navigating through the application was done by either using the back button or buttons that provided functionality, which made the application difficulty to navigate. To make navigation the application easier the options button was created. This button is present in the top right corner of each of the applications screens apart from the login, register and calculateBMR screen. This is achieved using the onCreateOptionsMenu method in each activity, which allows a drop down menu to appear when the options button is clicked. This is done by importing the android.view library, and creating an instance of the MenuInflater class. This looks for the activity\_main\_actions.xml layout, in the resource file, which sets the buttons to be displayed in the drop down menu.

public boolean onCreateOptionsMenu(Menu menu){  
 MenuInflater inflater = getMenuInflater();  
 inflater.inflate(R.menu.*activity\_main\_actions*, menu);  
 return super.onCreateOptionsMenu(menu);  
}

The function of the buttons in the dropdown menu are determined by the onOptionsItemSelected method in each activity with the options button. This contains a switch statement, which calls a method that gives that button its functionality.



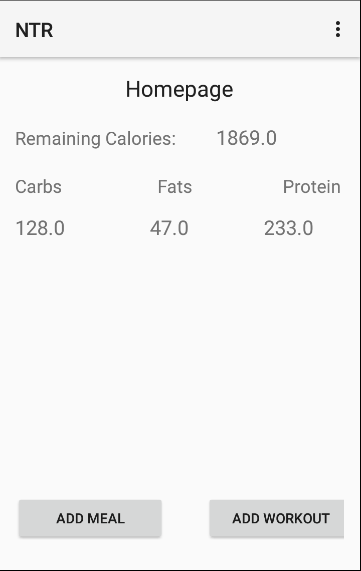


Figure Options Button

# CHAPTER 5 Testing and Evaluation

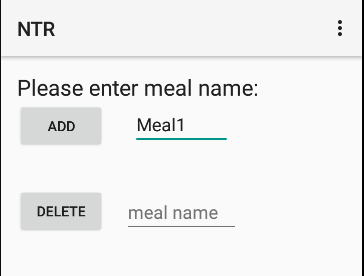
In this chapter the testing and evaluation performed on the application is discussed.

## Testing

This section goes through the tests performed on the application, to ensure each piece of functionality worked as it should. This was done by comparing what was expected to have happened and what actually happened for each of the features, by checking the SQLite database and what was displayed on screen.

#### Adding a meal

To test to see whether the application was correctly adding a meal, the database was viewed in a Mozilla Firefox extension called SQLite manager. The meal name was entered and add clicked. By looking at the database on SQLite manager, meal could be seen to have been correctly added to the database.



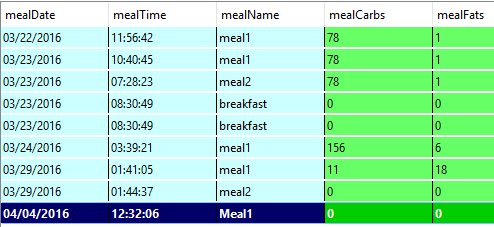


Figure Adding meal test

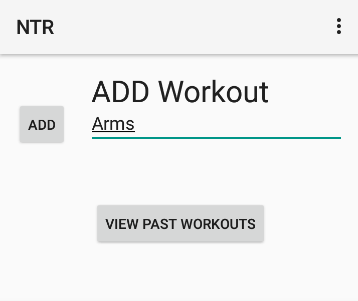
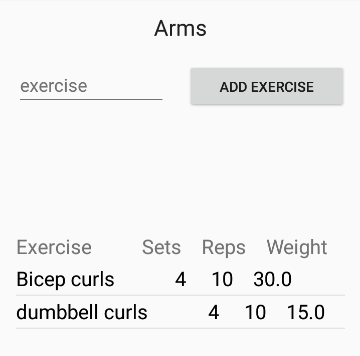
This method was used again to test that the application correctly added a food to the data base. When a new food is added, it should also add it to the current meal. To check this the database was examined again and it was observed that the food had been successfully added to the database, as well as to the mealEntry table which links a food to a meal. By looking at the addFood screen, it could see that the new food item was added correctly as it is displayed at the bottom of the screen, which is displayed by querying the database along with the updated calories, carbohydrates, fats and protein, which are updated on screen by querying the database.

## 

Figure Adding food to meal test

#### Adding a workout

Similarly to adding a Meal, to test whether the application correctly added the new workout to the database, to confirm it had been added by lo the workout table in the database. When adding an exercise to the workout the exercise is added to the exEntry table in the database which links and exercise to a workout.



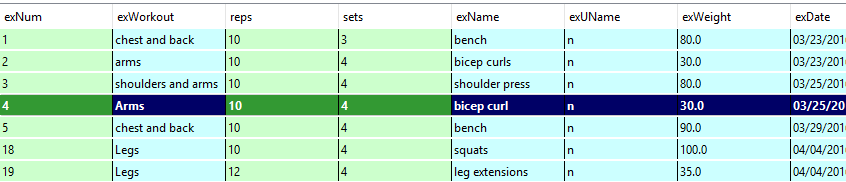


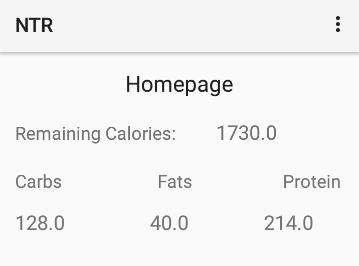
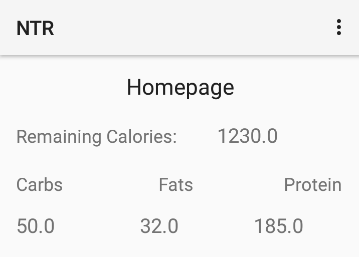
Figure Adding workout test

#### Adjusting remaining calories

In the homepage screen after a meal has been added, the remaining calories, carbohydrates, fats and protein should be adjusted by subtracting the amount of each in the meal and display the remaining. To test that this worked correctly the amount before adding a meal and after the meal was added was checked.

## 

Before adding a meal



After adding a meal

Figure Adjusting calories test

### Usability testing

Towards the end of the development of the application some usability testing was carried out. Usability testing is used to get feedback from the software’s target users. In this projects case the target users are people who want to lose weight, need to watch their diet while training for a sport or simply want to improve their health. During a usability test the test users were asked to complete tasks using the functionality the application provides. While the test users complete the task I observed, which allowed me to identify any problems with the application the users were having as they completed the tasks.

The task set for the test users were to simply create a new user, complete a meal entry with foods and observe the remaining calories change according to the nutritional value of the meal that was entered and complete a workout entry. During this test any difficulty the user had performing these tasks or any suggestions to the layout and design of the application was noted. The results of these tests were evaluated and where possible changes were made, which are both discussed in the evaluation section of this chapter.

## **Evaluation**

In this section I will talk about how well I completed the project based on what I had set out to do, what I would have done differently and future work I will do to improve it my project.

### Functional requirements

Based on the functional requirements gathered in the requirements gathering section in chapter 3, the functions of the application will be compared to the proposed functional requirements.

In the proposed functional requirements in chapter three, the must have functions are the most important pieces of functionality that the application will have and are essential to create the application. The application created for this project satisfies all of the must have functional requirements, as it can do all of the functions required. As described in the must have functional requirements the application has the ability to record workouts, record meals, show users remaining calories, carbohydrates, fats and protein, save foods, calculate users calories and has a easy to use design and layout.

In the should have section of the functional requirements gathered, the two functions that were said the application should have. Unfortunately due to my lack of experience in android application development and the learning curve involved with this project, I was unable to properly manage my time. This meant that by the time this report was written the connectivity to the online database was not fully implemented. Despite this, I will be working towards getting the connectivity to the online database fully operational by the time the application is demoed to the examiners. As the implementation of the online database was the next step in the development of the application, graphs that allow the visualisation of progress to the users will not be implemented by the time the application is demoed.

As some of the should have functionality was not implemented this meant that unfortunately none of the could have functionality was implemented.

### Non-functional requirements

Non-functional requirements can be described as the characteristics of the application as a whole, they do not apply to any individual function. These include Usability, Interoperability, Reliability and Flexibility.

**Usability –** It is important for the application to be easy to use to allow users to take advantage of the applications functionality. There is exactly everything needed to provide the functionality intended by each screen. All of the buttons and text are big enough to be seen but also not too big as to clutter the screen. The colour scheme is not too vibrant or dark to make using the application a pleasant experience. Finally the applications is easily navigated as all the screens are centred around the homepage and from each screen the user has the ability to log out and return to the homepage.

**Interoperability –** the minimum API my app is compatible with is version 21, as when developing various screens I was getting errors that the minimum API was too low for what was being implemented. The API was therefor set to 21, this reduces the amount of compatible devices. Along with this some of the screens, when used with different devices may appear differently due to the way some of the layouts are designed. Making the layouts compatible with more devices would have taken more time than was available, and so the problem was not fixed.

**Reliability –** After usability testing and my own testing of the application had been carried out, the application ran without any errors, leading to the conclusion the application was quite reliable when used.

**Flexibility –** This refers to the ability to extend the applications. Android applications support a high level of flexibility, with no restrictions to the developer allowing extra features to be easily added.

### User testing

As outlined earlier in the chapter, some user testing was carried out on the application by some test users. As a result of the tests, feedback was received along with some suggestions. For the most part the feedback was good, users found the application easily navigable and information was presented where necessary which helped to reduce clutter on the screens. One suggestion was made was when an input of numbers was required, to have the keyboard automatically change to a number keyboard instead of the user having to change it manually. Another suggestion was to allow the users to search previous workouts based on the workout name so that the user would be presented with a list of workouts with the same name so users could see their progress for that particular workout. The last suggestion received was to implement some method of reminding the user to record their meals and workouts.

As the user testing was carried out towards the end of development, and as some of the main functionality had not yet been implemented this meant that unfortunately the suggestions made by the users would not be implemented into the application although they did provide some ideas as to what could be added to the project

### Project Scheduling

When evaluating the project, an important aspect of evaluation is to look at the proposed scheduling, and how well it worked throughout the development. Due to the fact that I had never before developed an android application, trying to plan for the development of one was a difficult task, as I did not know of the amount work that it would entail.

As a result of this much of the proposed functionality was not completed and can be seen in the updated Gantt chart. The length of time taken to develop the core functionality due to the learning curve that was involved, meant that some of the features I had wanted to include could not be implemented. Another reason it took so much longer to implement than expected was how slow android studio and the android phone emulator ran on my computer. Both applications which are essential to the development of android applications really slowed down the process. This has resulted in the only functionality of the application, is the proposed core functionality I had set when in the planning phase of the project.

As can be seen in the following figures, some aspects of the project took much longer than expected, which resulted in reduced functionality, while still maintaining the main functionality I had set out to achieve.

**Proposed Gantt chart**



Figure Proposed Gantt chart

**Actual Gantt chart**

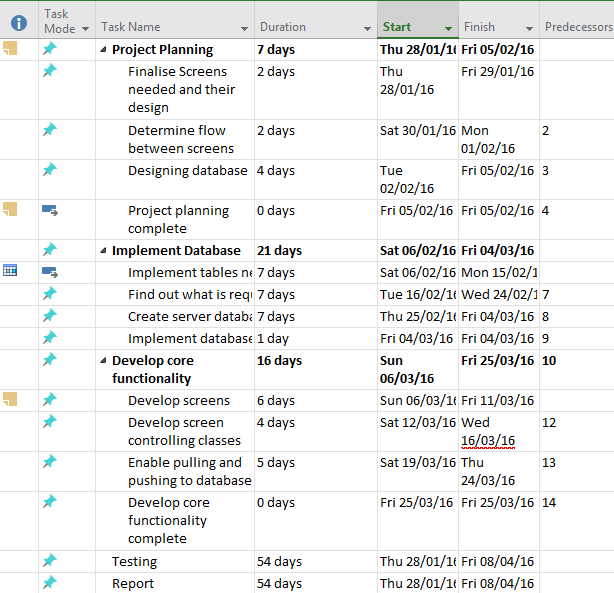
****

Figure Actual Gantt chart

## Future work

There were three other features that I wanted to put in my application, but was not able to simply because I was inexperienced in android application and was unaware of the time it would take to implement all the screens and functionality the application needed. Some of the features I could add in future are:

**Online database connectivity –** as development in this is already in progress, hopefully this functionality will be in full operation by the time the project is demoed.

**Progress graphs –** These would allow the user to be able to visualise their progress over a period of time. Some graphs that could be viewed would be to see how close the user was to reaching their calorific goals each day, or weights for different exercises. This could be done using GraphView libraries.

**Alerts –** This would involve reminding the user to keep entering their meals and workouts each day, which would help keep their diet and workout tracking consistent and help them keep track of progress.

**User profiles –** Users would be able to share their progress, diets and workouts with other users. This could help keep people motivated

**More detailed view of past workouts –** Based on user feedback, a better system of viewing past recorded workouts needs to be implemented, so that progress is made easier to access.

**Automatic keyboard change –** when number input is required, the keyboard would automatically change to numbers, which would help speed up the process of recording workouts and meals.

# Conclusion

Although I was unable to implement all of functionality I had hoped, I am still very pleased as to how the project went as a whole as the finished application was completed to a standard that I am happy with. I would have liked to implement some more functionality but time and inexperience with android development prevented this. While working on the project I learned a new technology and gained experience working on a large project from start to finish. I have learned a lot about time management, prioritising tasks and organisation. This experience will be invaluable to me in the future when working on other projects or in a professional environment, I will be able to use the lessons and new technologies I have learned.

**Appendix A: Application structure**

**View Past Workouts**

**Add Exercise**

**Add Workout**

**View Past Meals**

**Add Food**

**Add Meal**

**Homepage**

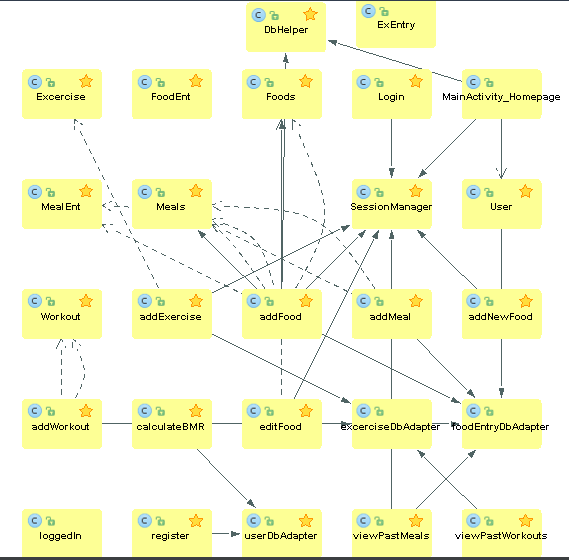
**Calculate Calories**

**Register**

**Login**

**Add New Food**

**Appendix B**

****

# References

1. https://www.myfitnesspal.com/ [↑](#endnote-ref-1)
2. https://play.google.com/store/apps/details?id=com.meuuha.nutritiontracker [↑](#endnote-ref-2)
3. https://play.google.com/store/apps/details?id=com.fitnow.loseit [↑](#endnote-ref-3)
4. https://www.jefit.com/ [↑](#endnote-ref-4)
5. https://play.google.com/store/apps/details?id=com.body.building&hl=en [↑](#endnote-ref-5)
6. https://www.myfitnesspal.com/ [↑](#endnote-ref-6)
7. https://www.loseit.com/ [↑](#endnote-ref-7)
8. https://play.google.com/store/apps/details?id=com.github.jamesgay.fitnotes&hl=en [↑](#endnote-ref-8)
9. https://www.jefit.com/ [↑](#endnote-ref-9)
10. https://www.android.com/ [↑](#endnote-ref-10)
11. http://developer.android.com/sdk/index.html [↑](#endnote-ref-11)
12. http://www.tutorialspoint.com/android/ [↑](#endnote-ref-12)
13. http://stackoverflow.com/ [↑](#endnote-ref-13)
14. http://developer.android.com/sdk/index.html?gclid=Cj0KEQjw8u23BRCg6YnzmJmPqYgBEiQALf\_XzWatFr8M9tgyb2TIJWneBdJk2kAVpZnPwgp4thkxkI8aAgR-8P8HAQ [↑](#endnote-ref-14)
15. https://www.jetbrains.com/ [↑](#endnote-ref-15)
16. https://en.wikipedia.org/wiki/Gradle [↑](#endnote-ref-16)
17. https://en.wikipedia.org/wiki/Lint\_(software) [↑](#endnote-ref-17)
18. http://proguard.sourceforge.net/ [↑](#endnote-ref-18)
19. https://en.wikipedia.org/wiki/Android\_software\_development [↑](#endnote-ref-19)
20. https://www.sqlite.org/ [↑](#endnote-ref-20)
21. http://www.justinmind.com/ [↑](#endnote-ref-21)
22. http://developer.android.com/guide/topics/manifest/manifest-intro.html [↑](#endnote-ref-22)
23. http://developer.android.com/guide/components/activities.html [↑](#endnote-ref-23)
24. http://developer.android.com/reference/android/content/Intent.html [↑](#endnote-ref-24)
25. https://en.wikipedia.org/wiki/Harris%E2%80%93Benedict\_equation [↑](#endnote-ref-25)
26. http://www.superskinnyme.com/calculate-tdee.html [↑](#endnote-ref-26)
27. http://www.acaloriecounter.com/diet/calorie-deficit-to-lose-weight/ [↑](#endnote-ref-27)
28. <http://www.acaloriecounter.com/diet/how-many-calories-to-build-muscle-or-gain-weight/> [↑](#endnote-ref-28)