Fit-Life App

Project Engineering

Year 4

Shane Crotty

Bachelor of Engineering (Honours) in Software and Electronic Engineering

Atlantic Technological University

2021/2022

Declaration

This project is presented in partial fulfilment of the requirements for the degree of Bachelor of Engineering (Honours) in Software and Electronic Engineering at Galway-Mayo Institute of Technology.

This project is my own work, except where otherwise accredited. Where the work of others has been used or incorporated during this project, this is acknowledged and referenced.

Shane Crotty.

Acknowledgements

Supervisor: Michelle Lynch.

Lecturer: Paul Lennon, Report, Video, Slides, Poster, Teamwork.

Lecturer: Brian o shea, informed me about expo.io to learn React Native.

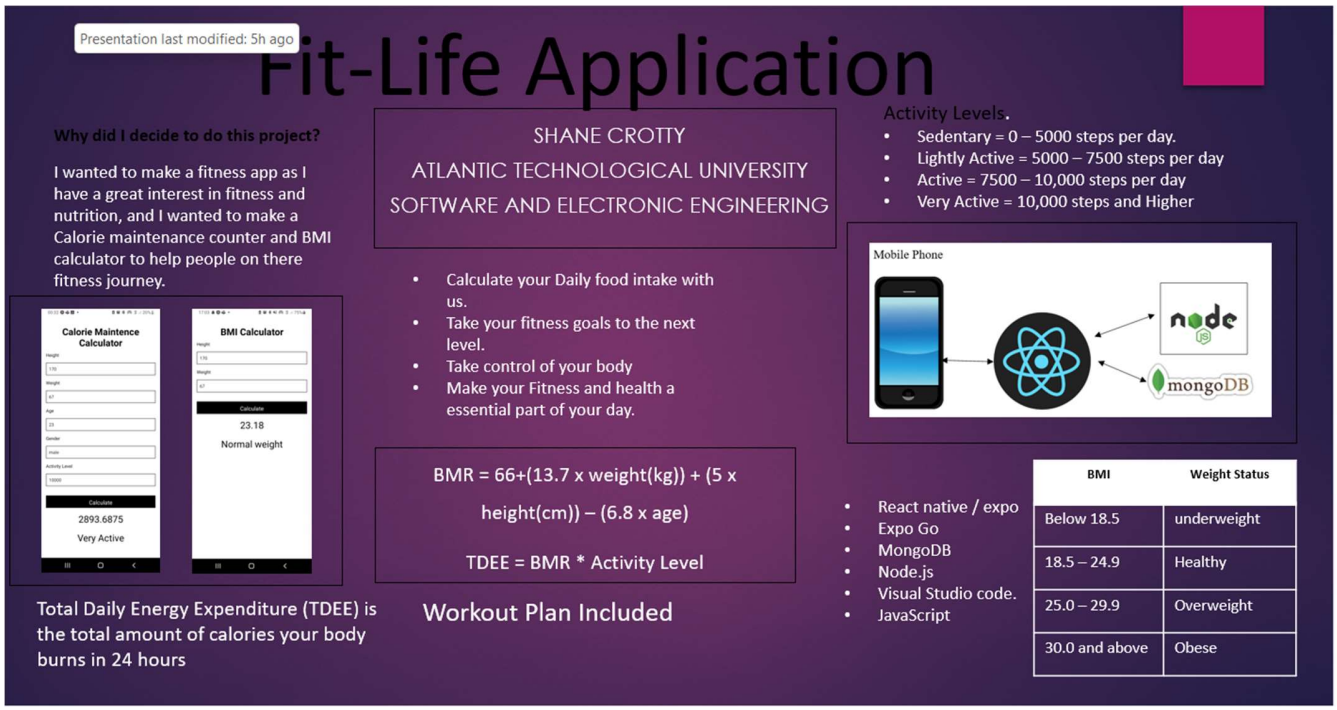
Table of Contents

1 Summary

For my final year project, I am creating a mobile fitness app that will help you on your fitness journey. This mobile app will have client user personalization such as age, gender, height, weight, and Activity level, this will then help create the client's daily calorie intake to help maintain, gain, or lose weight depending on the clients' personal goals. The number of steps you achieve will have an impact on the number of calories/macros you can eat in the specific day, as your activity level has an impact on how many calories you burn throughout the day. This app will help keep the client on track to achieve their goals whilst still being able to go above their steps for their selected activity level in the user personalization, I decided to do this for my project because I found whilst calculating calories/macros myself I would find myself going over my activity level steps on certain days thus I would not know what my daily macros should be. This app will help solve that problem.

When the client logs onto the app they will have the ability to customize the app to their specific goals and information. I will be using react native as it's a JavaScript based mobile app framework that allows you to build mobile apps for iOS and Android. The framework lets you create an application for various platforms by using the same codebase.

2 Poster


Presentation last modified: 5h ago
Fit-Life Application
Why did I decide to do this project?
I wanted to make a fitness app as I have a great interest in fitness and nutrition, and I wanted to make a Calorie maintenance counter and BMI calculator to help people on there fitness journey.
SHANE CROTTY
ATLANTIC TECHNOLOGICAL UNIVERSITY
SOFTWARE AND ELECTRONIC ENGINEERING
Activity Levels.
Sedentary = 0 – 5000 steps per day.
Lightly Active = 5000 – 7500 steps per day
Active = 7500 – 10,000 steps per day
Very Active = 10,000 steps and Higher
Mobile Phone
node
mongoDB
Calorie Maintence Calculator
2893.6875
Very Active
BMI Calculator
23.18
Normal weight
Calculate your Daily food intake with us.
Take your fitness goals to the next level.
Take control of your body
Make your Fitness and health a essential part of your day.
BMR = 66+(13.7 x weight(kg)) + (5 x height(cm)) – (6.8 x age)
TDEE = BMR * Activity Level
Total Daily Energy Expenditure (TDEE) is the total amount of calories your body burns in 24 hours
Workout Plan Included
React native / expo
Expo Go
MongoDB
Node.js
Visual Studio code.
JavaScript
BMI
Weight Status
Below 18.5
underweight
18.5 – 24.9
Healthy
25.0 – 29.9
Overweight
30.0 and above
Obese

3 Introduction

For my final year project, I have created a mobile fitness app that will help you on your fitness journey. This mobile app will have client user personalization such as age, gender, height, weight, and Activity level, this will then help create the client's daily calorie intake to help maintain, gain, or lose weight depending on the clients' personal goals. Thus, having different macros and information for each person. The calorie maintenance calculator will calculate the clients' calories to stay at the same weight and just maintain the size they have, the daily steps the client does will have a impact on this number as they could b burning more calories than they are consuming resulting in weight loss. So, to help with this the client may wear a step counter watch and have it count the number of steps they are taking throughout the day, the client will then use the Calorie maintenance calculator and enter the number of steps into the Activity Level section along with the other information required, morning weight, height, age, and gender. With this information the app will calculate the client's personal maintenance calories. This app will help keep the client on track to achieve their goals whilst still being able to go above their steps for their selected activity level in the user personalization, as the app will recalculate the calories/macros if the client goes over their step limit and changes their activity level for the day. I decided to do this for my project because I found whilst calculating calories/macros myself I would find myself going over my activity level steps on certain days thus I would not know what my daily macros should be. This app will help solve that problem.

I also have a Body Mass Index counter (BMI), this will help you figure out whether you are a healthy weight for your height and weight, it will show if you are overweight, underweight or at a healthy weight. This will help people take charge of their health and transform their lifestyle.

When the client logs onto the app they will have the ability to customize the app to their specific goals and information. They will enter their information and get the calories and macros to fit their goals, they will also get their step counter which will help them keep to their fitness journey by helping them eat the correct amount depending on their daily

activities. I have used react native as it's a JavaScript based mobile app framework that allows you to build mobile apps for iOS and Android. The framework lets you create an application for various platforms by using the same codebase

4 Background

Why did I decide to do this project?

I wanted to make a fitness app as I have a great interest in fitness and nutrition, and I wanted to make a Calorie maintenance counter that you have on your phone and allows you to enter the specific number of steps you did and have it calculate your maintenance. I wanted to have the app take the step count for the Activity Level instead of having it be off the ball more confusing sort of structure like some other fitness apps have.

How does Calorie maintenance work?

Total Daily Energy Expenditure (TDEE) is the total amount of calories your body burns in 24hours, it can vary from person to person and is much higher for athletes or extremely active people. Calorie maintenance level refers to the number of calories you would need to consume to maintain your current body weight. There are a few different formulas you can use to determine your caloric maintenance level. These formulas consider the factors of age, gender height, weight, lean body mass and activity level. The easiest method is based only on total body weight,

Fat Loss= 12-13 calories per pound of body weight

Maintenance = 15-16 calories per pound of body weight

Weight Gain = 18-19 calories per pound of body weight

Eg: If you are a 160-pound male or female wanting to lose weight, you would multiply 160 by 12 or 13. So you should lose weight if you eat around 1900-2000 calories per day.

The most accurate method for determining TDEE is to determine your basal metabolic rate (BMR) by using your height, weight, age, and gender, then multiply the BMR by an Activity level to calculate TDEE. BMR is the total number of calories your body requires for normal body functions excluding Activity level.

The Harris-Benedict formula is the formula to calculate BMR:

BMR = 66+(13.7 x weight(kg)) + (5 x height(cm)) – (6.8 x age)

Eg. You are a 30-year-old man, you are 5’7” tall(170cm) and weight 70kg your BMR would be:

66+(13.7 x 70) + (5 x 170) – (6.8 x 30) = 1671 calories per day, but this is only your BMR, so this is what you will need to eat per day to maintain if you do not move at all throughout the day.

So, to calculate your TDEE you will multiply your BMR by your activity level.

Sedentary = less than 5000 steps: BMR*1.2

Lightly Active = 5000 – 7500 steps: BMR*1.375

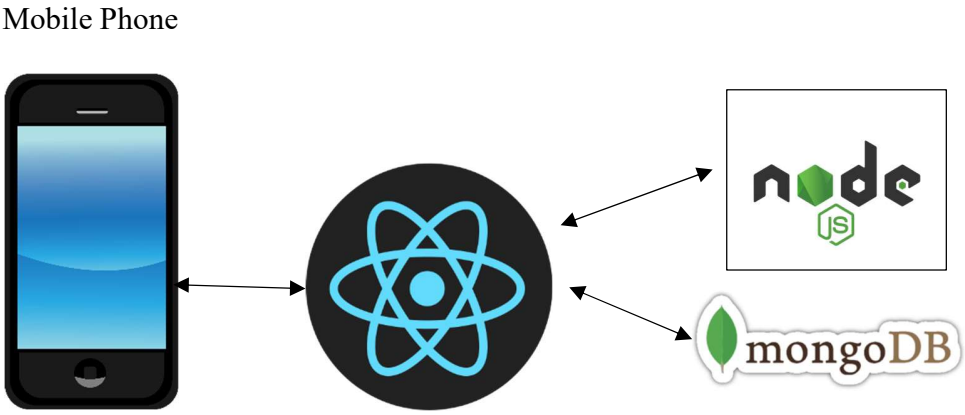
Active = 7500 – 10000 steps: BMR*1.55

Very Active = higher than 10000 steps: BMR*1.725

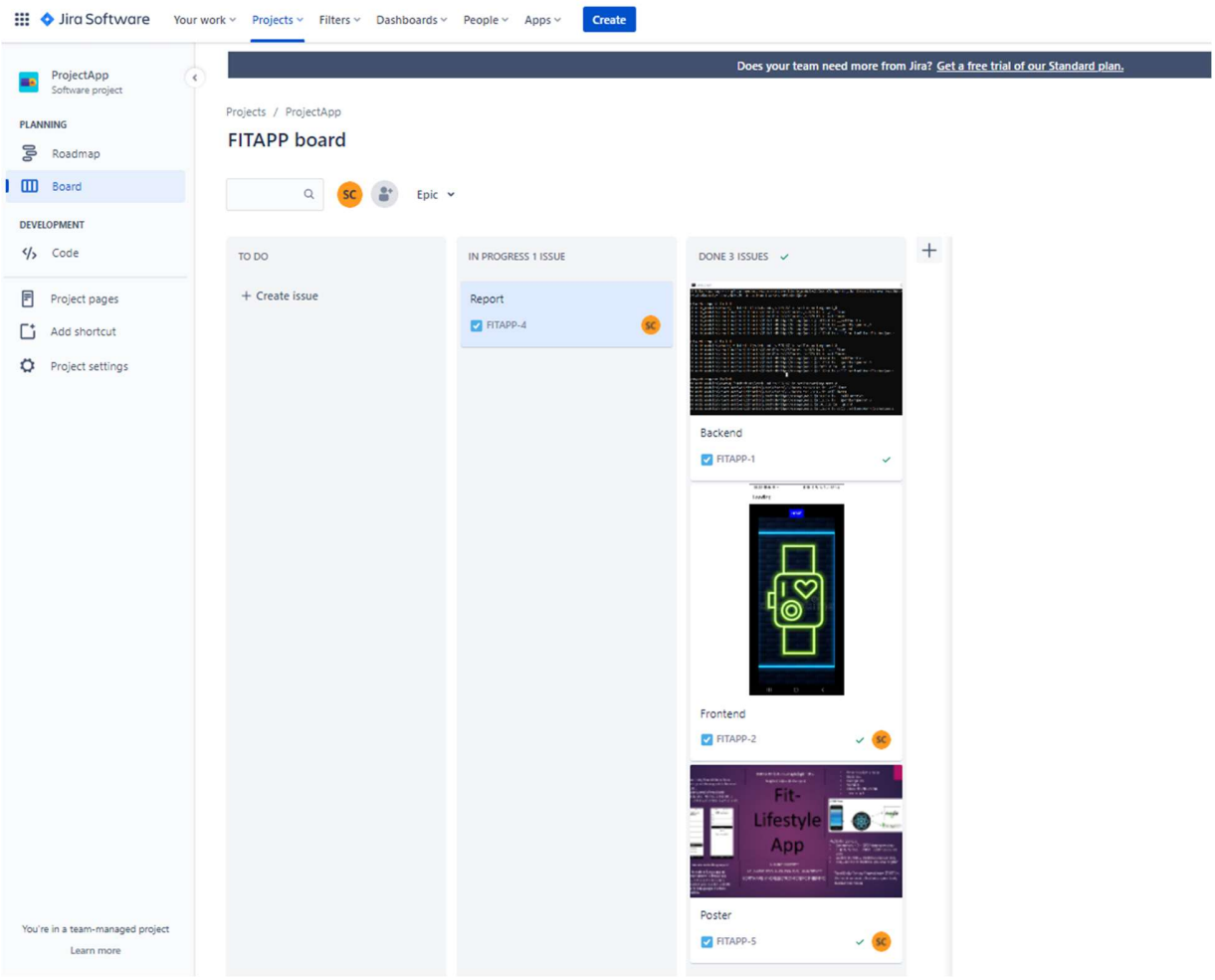
So, if then person above has an activity level of around 8000 steps per day, He is Active, so his TDEE is going to be 1671*1.55 = 2590 calories.

So, if this is what he will need to eat to maintain his current weight, say he wants to lose weight, you would either consume fewer calories per day or increase your step count (Activity Level).

5 Project Architecture



6 Project Plan



I used Jira to plan out my Project, I used this because I have experience using this form of planning from my Work placement in 2021. created ticket issues that needed to be completed and I set the status to in progress when I was working on the specific part of my project, I commented my progress in the ticket and when the ticket was completed I mark them as Done.

7 Creating the Project

7.1 React Native/ Expo

To set up the react native first I downloaded Visual Studio Code, Node.js, Git and on my mobile phone I downloaded an app called Expo Go.

I then opened a command prompt and went to the directory I wanted to create the project in.

I typed the below command to install Expo.

Installing Expo CLI

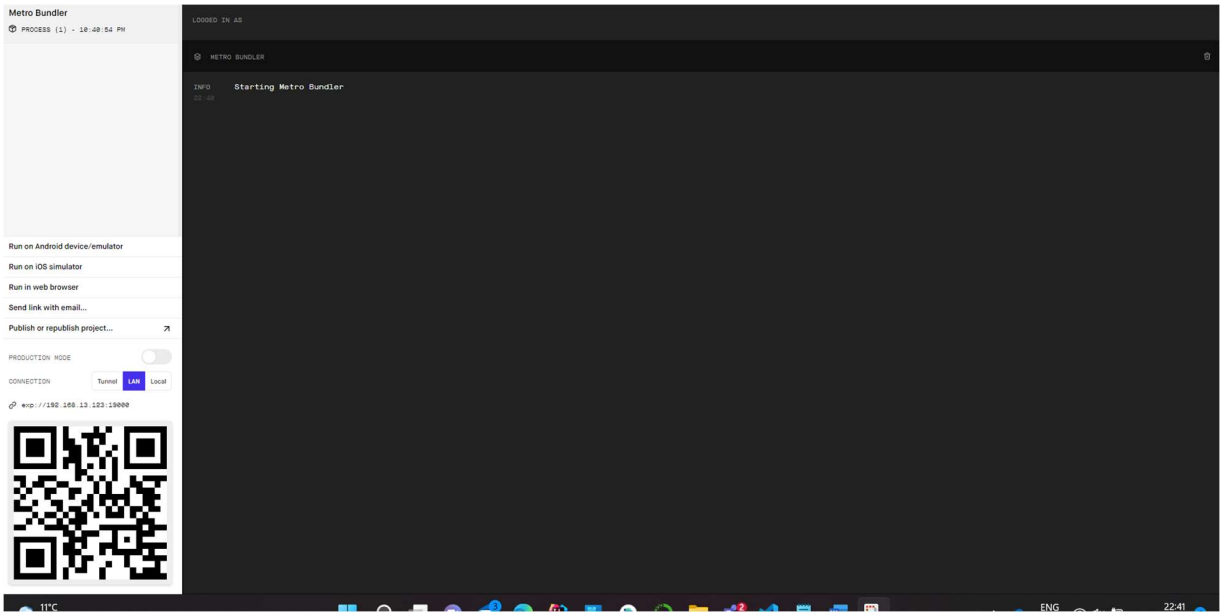
```
# Install the command line tools
$ npm install --global expo-cli
```

I created a new expo project with the following command

```
# Create a project named my-app. Select the "blank" template when prompted
$ expo init my-app
```

I then Navigated into this app and typed code . to open the project in Visual Studio Code.

To run the app, you will type in the command line expo start or npm start and this will start the metro bundler.



You would then go into the Expo Go app on your mobile phone and scan the barcode to download and install the project app on the phone.

7.2 Backend

For the backend of my project, I used mongo db. and node.js as the database to store the information for the Person using the App when creating an account.

I created a separate react native project for the server and installed libraries and modules I would need.

```
const express = require('express')
const bodyParser = require('body-Parser')
const mongoose = require('mongoose')

const app = express()
const PORT = 8000
const {mogoUrl} = require('./keys')

require('./models/User');

const requireToken = require('./middleware/requireToken')
const authRoutes = require('./routes/authRoutes')
app.use(bodyParser.json())
app.use(authRoutes)

mongoose.connect(mogoUrl,{
    useNewUrlParser:true,
    useUnifiedTopology:true
})

mongoose.connection.on('connected',()=>{
    console.log("connected to mongo")
})

mongoose.connection.on('error',(err)=>{
    console.log("this is a error", err)
})



app.get('/',requireToken,(req,res)=>{
    res.send("your email is " + req.user.email)
})


app.listen(PORT,()=>{
    console.log("Server running "+PORT)
})
```

You need to link your mongo DB account to the above code to use the mongoose connect code.

When the connection is established, the console will receive the above messages saying

```
PS C:\Users\35387\OneDrive - GMIT\Documents\Auth_Server> nodemon index
[nodemon] 2.0.15
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index index.js`
Server running 8000
connected to mongo
```

```
{
  "name": "auth_server",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  ▷ Debug
  "scripts": {
    "start": "nodemon index.js"
  },
  "author": "ZCROSHA",
  "license": "ISC",
  "dependencies": {
    "bcrypt": "^5.0.1",
    "body-parser": "^1.19.2",
    "express": "^4.17.3",
    "jsonwebtoken": "^8.5.1",
    "mongoose": "^6.2.4"
  },
  "devDependencies": {
    "nodemon": "^2.0.15"
  }
}
```

Express.js: this allows you to preform different tasks on request and response.

Bcrypt: this is a hashing function that allows you to create a password system with a salt

Body-parser: this parses your request and converts it so it's easier to get information.

Mongoose: this is a node js based library.

```
const userSchema = new mongoose.Schema({
    email:{
        type:String,
        unique:true,
        required:true
    },
    password: {
        type:String,
        required:true
    }
})
```

This is the needed info in the screen input of the signup and login pages, they are both strings to input the information and are both required. The email is also set to unique as there cannot be two accounts with the same email address.

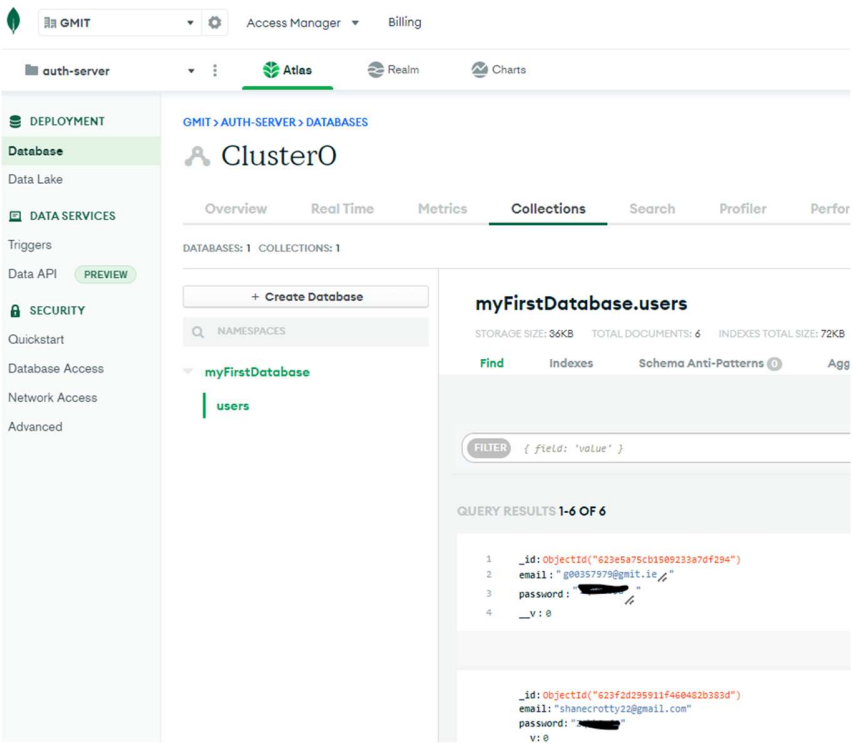
```
router.post('/signup',async(req,res)=>{
   // console.log(req.body)

    const {email,password} = req.body;

    try{
    const user = new User({email,password})
    await user.save();
    const token = jwt.sign({userId:user._id},jwtkey )
    res.send({token})

    }catch(err){
    return res.status(422).send(err.message)
    }
})
```

To sign up you will need to enter an email and password and you will receive a unique token and the accounts will be saved in the data base as seen in the above code.

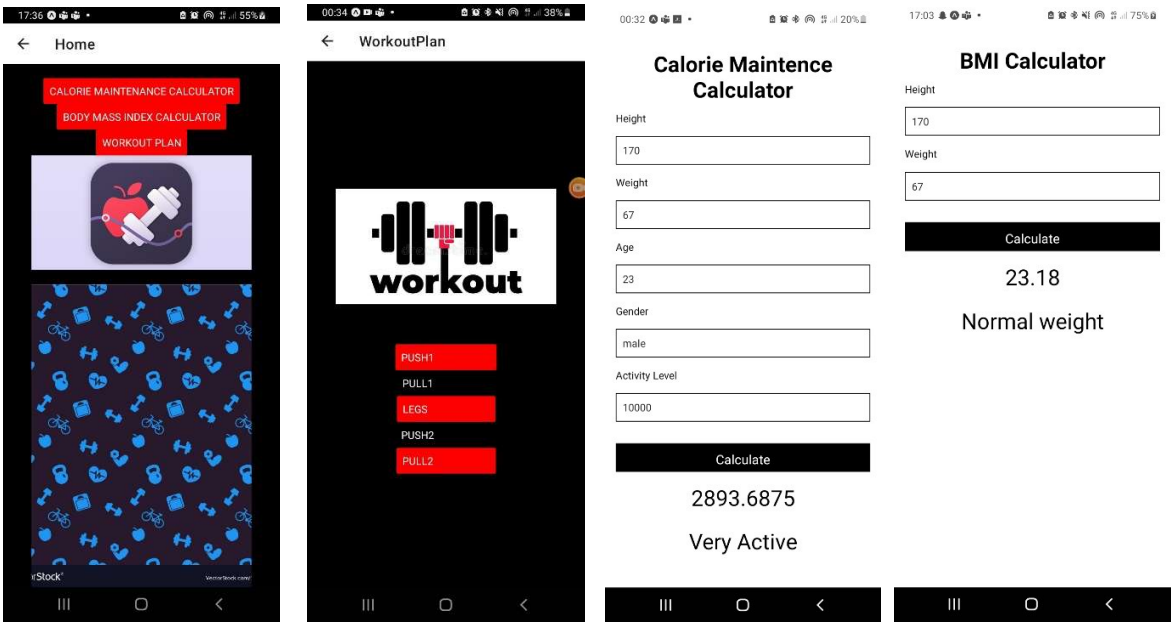


7.3 Frontend

Whilst designing the frontend of the Fit-life mobile app I focus on making the app look like a fitness app as that s what it is, I wanted the app to stand out to people who are interested in fitnes.

7.3.1 Navigation.

I wanted to make the app easy to use and have a clear understanding of where the various content is. I have various screens that are all linked to one home page of the mobile app and have clear labelled buttons to enable the client to know exactly where there going once the button is pressed.



From the home screen you will have a option to go to any of the 3 other screens, in the Workout plan screen you will have another choice to navigate to even more screens.

I have done this by installing the dependencies to make the navigation possible,

```
"@react-navigation/native": "^6.0.10",
"@react-navigation/native-stack": "^6.6.1",
```

These will allow us enable navigation between screens,

```
function Home({ navigation }) {
  return (
    <View style={styles.container}style={{ flex: 1, alignItems: 'center', backgroundColor: 'black', justifyContent: 'center' }}>
      <Text>Home</Text>
      <Button
      title="Calorie Maintenance Calculator"
      color="red"
      onPress={() => navigation.navigate('CalorieCalculator')}
      style={{ backgroundColor: 'green' }}
      />
      <Button
      title="Body Mass Index Calculator"
      color="red"
      onPress={() => navigation.navigate('BMI')}
      style={{ backgroundColor: 'green' }}
```

You enable the navigation at the top of the function screen, and you can then set up a button that when you press it will navigate to another screen such as the Calorie Calculator as shown above.

7.3.2 Calorie Calculator Screen.

When making this page I needed to be able to input certain information to have the formula calculate correctly. So needed to have placeholders to input data for Height, Weight, Age, Gender, and Activity Level.

```
    return (
      <View style = {styles.container}>
Text style={styles.title}>Calorie Maintence</Text>

          <Text  style = {styles.label}>Height</Text>
          <TextInput style = {styles.input}
            underlineColorAndroid = "transparent"
            placeholder = "Height (Cm)"
            autoCapitalize = "none"
            onChangeText = {this.handleHeight}/>
Text  style = {styles.label}>Weight</Text>
          <TextInput style = {styles.input}
            underlineColorAndroid = "transparent"
            placeholder = "Weight (Kg)"
            autoCapitalize = "none"
            onChangeText = {this.handleWeight}/>
            <Text  style = {styles.label}>Age</Text>
          <TextInput style = {styles.input}
            underlineColorAndroid = "transparent"
            placeholder = "Age"
            autoCapitalize = "none"
            onChangeText = {this.handleAge}/>
Text  style = {styles.label}>Gender</Text>
          <TextInput style = {styles.input}
            underlineColorAndroid = "transparent"
            placeholder = "Gender"
            autoCapitalize = "none"
            onChangeText = {this.handleWeight}/>
          <Text  style = {styles.label}>Activity Level</Text>
          <TextInput style = {styles.input}
            underlineColorAndroid = "transparent"
            placeholder = "Activity_Level"
            autoCapitalize = "none"
            onChangeText = {this.handleActivity}/>
```

I have put the name of the required above the placeholder and also, I have it in the placeholder and I have set it that when the input in the placeholder is changed it will turn into the information handle that was made earlier to use as the data for the formula.

```
<Text  style = {styles.label}>Weight</Text>
          <TextInput style = {styles.input}
            underlineColorAndroid = "transparent"
            placeholder = "Weight (Kg)"
            autoCapitalize = "none"
            onChangeText = {this.handleWeight}/>
            <Text  style = {styles.label}>Age</Text>
```

I set the certain text earlier in the code.

```
    handleHeight = (text) => {
        this.setState({ height: text })
    }
    handleWeight = (text) => {
        this.setState({ weight: text })
    }
   handleAge = (text) => {
        this.setState({ age: text })
}
handleActivity = (text) => {
    this.setState({ Activity_Level: text })
}
```

I used the Formula to calculate BMR first.

```
calculate = (height, weight, age, Activity_Level) => {
    //calculation
    var result = 66 + (parseFloat(weight)*13.7)+(parseFloat(height)*5) - (parseFloat(age)*6.8);
    result = result.toFixed(2);
```

This will give us the BMR result that will let us then set a different Activity Level.

The Activity level is set by the certain Steps counted in a specific day. When you get to a certain Activity level you will multiply your BMR result by a certain number, the higher your step count the higher the number.

```
//display result
this.setState({ bmi: result })
if(Activity_Level<4999){
   this.setState({ bmi: result*1.2 })
  // result = result*1.2
   this.setState({BmiResult:'Sedentary'})
}
else if(Activity_Level>=5000&&Activity_Level<7499){
   //result = result*1.375
   this.setState({ bmi: result*1.375 })
   this.setState({BmiResult:'Lightly Active'})
}
else if(Activity_Level>=7500&&Activity_Level<9999){
   this.setState({ bmi: result*1.55})
   this.setState({BmiResult:'Active'})
}
else if(Activity_Level>=10000){
   this.setState({ bmi: result*1.725})
   this.setState({BmiResult:'Very Active'})
}
else{
   alert('Incorrect Input!');
   this.setState({BmiResult:''})
}
```

7.3.3 BMI Screen.

I used the same idea when designing the BMI Calculator code., this code is a smaller code as the formula is smaller and you do not need much information off the client to calculate the result.

```
//calculation
var result = (parseFloat(weight)*10000)/(parseFloat(height)*parseFloat(height));
result = result.toFixed(2);
//display result
this.setState({ bmi: result })
if(result<18.5){
   this.setState({BmiResult:'Underweight'})
}
else if(result>=18.5&&result<25){
   this.setState({BmiResult:'Normal weight'})
}
else if(result>=25&&result<30){
   this.setState({BmiResult:'Overweight'})
}
else if(result>=30){
   this.setState({BmiResult:'Obese'})
}
else{
   alert('Incorrect Input!');
   this.setState({BmiResult:''})
```

8 Conclusion

Overall, I enjoyed the challenge of creating the Fit-Life mobile App, I feel like it has potential to go further in the fitness industry as it is a huge market. My app calculates a person's daily maintenance calories to help the client stay at a specific weight. If the client wishes to lose weight or gain weight depending on their fitness goals, they will simply add or minus 300 calories from there given maintenance. I feel like that Fit-Life could have a huge impact on the fitness industry as it allows you the freedom to change your activity level depending on your day and still reach your fitness goals. In the future I hope to continue working on fit life and improve upon the idea by adding its own step counter and add more nutritional information such as information on Protein, carbs and fats and then benefits these has when consumed at certain times of the day.

I found learning react native an interesting experience as I have always been interested in how mobile apps work and how the function. I mainly used expo.io to find out most of the information I needed to learn how to create this app, I would recommend this website to anyone wanting to learn how mobile applications work.

9 References

[1] Expo.io https://expo.dev.

[2] Google.

[3] MongoDB: https://account.mongodb.com/account/login

[4] Harris Benedict Formula: Harris-Benedict Equation - Resources (bmicalc.org)

[5] BMI Formula: BMI Formula (bmi-calculator.net)

[6] React Navigation: Getting started | React Navigation

[7] React Native: React Native · Learn once, write anywhere

[8] W3schools: W3Schools Online Web Tutorials

[9] MongoDB with Node.js MongoDB With Node.js | MongoDB