

Motorcycle Dashboard App

DT211c

BSc in Computer Science (Infrastructure)

**Luke Dowdall**

**C18375931**

**Supervisor: Basel Magableh**

School of Computer Science

Technological University, Dublin

**15/1/2022**

Abstract

Motorcycles can be tricky machines to maintain and some older motorcycles have seen a lot of use and their speed sensors and speedo cables may not be as accurate as they were when they were first manufactured. Using modern technology such as GPS, we are able to accurately track speed and this can be a much more costly fix instead of repairing/replacing an inaccurate or broken speed sensor.

The goal for this project is to create a fully functioning motorcycle dashboard interface that the user can use during their trip and then observe their trip after it is finished to see data at different stages and see where they can improve their driving style. This will tell the user where they may be over the stated speed limit on the road and can also show them their lean angle.

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Luke Dowdall

15/1/2022

Acknowledgements

I would like to thank my motorcycle friends for their support and help with the aspect of giving feedback and testing of my app.

I would also like to thank my family and friends for being supportive and understanding for the many days and hours I spent at my computer!

Lastly, I would like to thank my supervisor, Basel Magableh for his continued support and his expert knowledge in this field.

Table of Contents

[Table of Figures 7](#_Toc100660939)

[1. Introduction 8](#_Toc100660940)

[1.1. Project Background 8](#_Toc100660941)

[1.2. Project Description 9](#_Toc100660942)

[1.3. Project Aims and Objectives 11](#_Toc100660943)

[1.4. Project Scope 11](#_Toc100660944)

[1.5. Thesis Roadmap 12](#_Toc100660945)

[Research 12](#_Toc100660946)

[Design 12](#_Toc100660947)

[Development 12](#_Toc100660948)

[Testing and Evaluation 12](#_Toc100660949)

[Redevelopment 13](#_Toc100660950)

[Conclusions and Future Work 13](#_Toc100660951)

[2. Literature Review 14](#_Toc100660952)

[2.1. Introduction 14](#_Toc100660953)

[2.2. Alternative Existing Solutions 14](#_Toc100660954)

[GPS Speed – Joao Silviera 14](#_Toc100660955)

[Speedometer 55 – Stanislav Dvoychenko 15](#_Toc100660956)

[2.3. Technologies you’ve researched 16](#_Toc100660957)

[Mobile Technology 16](#_Toc100660958)

[Java 17](#_Toc100660959)

[Android Studio 17](#_Toc100660960)

[APIs 19](#_Toc100660961)

[GPS Speedometers 20](#_Toc100660962)

[2.4. Other Research you’ve done 20](#_Toc100660963)

[Lean Angle Calculation 20](#_Toc100660964)

[2.5. Existing Final Year Projects 21](#_Toc100660965)

[SmartDrive: Exploring the Use of GPS in Order to Track Speed and Promote Safe Driving – Richard Meyer 21](#_Toc100660966)

[Dynamo Dublin: Real-Time Bus Arrival Predictions – Anton Medves 23](#_Toc100660967)

[2.6. Conclusions 24](#_Toc100660968)

[3. Experiment Design 26](#_Toc100660969)

[3.1 Introduction 26](#_Toc100660970)

[3.2. Software Methodology 26](#_Toc100660971)

[Agile Methodology 26](#_Toc100660972)

[3.3. Overview of System 27](#_Toc100660973)

[3.4. Front End 28](#_Toc100660974)

[Use Case Diagrams 32](#_Toc100660975)

[3.5. Middle Tier 33](#_Toc100660976)

[3.7. Conclusions 33](#_Toc100660977)

[4. Experiment Development 35](#_Toc100660978)

[4.1. Introduction 35](#_Toc100660979)

[4.2. Experiment Development 35](#_Toc100660980)

[4.2.1. Speedometer functionality 36](#_Toc100660981)

[4.2.2. Maps functionality 38](#_Toc100660982)

[4.2.3. Lean Angle functionality 40](#_Toc100660983)

[4.2.4 Saved Instance State 40](#_Toc100660984)

[4.2.5 Convert Kilometres to Miles 41](#_Toc100660985)

[4.2.6 XML Layout 42](#_Toc100660986)

[4.3. Conclusions 42](#_Toc100660987)

[5. Testing and Evaluation 44](#_Toc100660988)

[5.1. Introduction 44](#_Toc100660989)

[5.2 System Testing 44](#_Toc100660990)

[5.2.1. Blackbox Testing 44](#_Toc100660991)

[Test Plan 45](#_Toc100660992)

[5.3. Evaluation 46](#_Toc100660993)

[5.4. Conclusions 46](#_Toc100660994)

[6. Conclusion and Future Work 47](#_Toc100660995)

[6.1. Introduction 47](#_Toc100660996)

[6.2. Conclusions 47](#_Toc100660997)

[6.2.1 Literature Review 47](#_Toc100660998)

[6.2.2 Motorcycle Dashboard Design 47](#_Toc100660999)

[6.2.3 Motorcycle Dashboard Development 48](#_Toc100661000)

[6.2.4 Motorcycle Dashboard Testing & Evaluation 48](#_Toc100661001)

[6.3. Future Work 48](#_Toc100661002)

[6.3.1 System Improvements 49](#_Toc100661003)

[6.3.2. Future Features 49](#_Toc100661004)

[6.3.3. GANTT Chart 49](#_Toc100661005)

[Bibliography 50](#_Toc100661006)

# Table of Figures

[Figure 1 - RSA statistics 9](#_Toc100660910)

[Figure 2 - App's first design prototype 10](#_Toc100660911)

[Figure 3 - GPS Speed Landscape 15](file:////Users/lukedowdall/Desktop/4thYear/FYP/InterimReport.docx#_Toc100660912)

[Figure 4 - GPS Speed Portrait 15](#_Toc100660913)

[Figure 5 - Speedometer 55 Landscape 16](file:////Users/lukedowdall/Desktop/4thYear/FYP/InterimReport.docx#_Toc100660914)

[Figure 6 - Speedometer 55 Portrait 16](#_Toc100660915)

[Figure 7 - AVD of MotorcycleDashboard 18](#_Toc100660916)

[Figure 8 - API Manager 19](#_Toc100660917)

[Figure 9 - Azimuth 21](#_Toc100660918)

[Figure 10 - SmartDrive 23](#_Toc100660919)

[Figure 11 - DynamoDublin 24](#_Toc100660920)

[Figure 12 - Agile Methodology 26](#_Toc100660921)

[Figure 13 - Technical Architecture 28](#_Toc100660922)

[Figure 14 - Horizontal low fidelity 29](#_Toc100660923)

[Figure 15 - Portrait low fidelity 29](#_Toc100660924)

[Figure 16 - First Design Prototype 30](#_Toc100660925)

[Figure 17 - Final Design Landscape 31](#_Toc100660926)

[Figure 18 - Final Design Portrait 31](#_Toc100660927)

[Figure 19 - Use Case Iteration 1 32](#_Toc100660928)

[Figure 20 - Use Case Iteration 2 33](#_Toc100660929)

[Figure 21 - Horizontal prototype design 35](#_Toc100660930)

[Figure 22 - GPS Speedo Code 36](#_Toc100660931)

[Figure 23 - Speedometer 37](#_Toc100660932)

[Figure 24 - Avg Speed 38](#_Toc100660933)

[Figure 25 - Map every 50 metres Figure 26 - Map every 100 metres 39](#_Toc100660934)

[Figure 27 - Lean Angle 40](#_Toc100660935)

[Figure 28 - Metric Switch 41](#_Toc100660936)

[Figure 29 - Layout Land 42](#_Toc100660937)

[Figure 30 - GANTT Chart 49](#_Toc100660938)

# 1. Introduction

## Project Background

There is no debating that motorcycles are the most dangerous mode of transport on public roads. This fact can be shown with the many different tables of statistics published by the RSA.

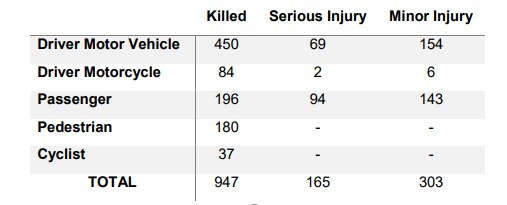


Figure 1 - RSA statistics

During the period of 2008 to 2012, RSA conducted a study of 867 serious collisions on Irish roads. This table shows that motorcyclists have a lesser chance of surviving any serious collisions are there is nowhere near as much protection when driving a motor vehicle.

One of the most harrowing stat shows that of the 1.5% of all registered vehicles are motorcycles but also account for 11.6% of collision fatalities in 2020, this means that motorcyclists are over 28 times more likely to succumb to a collision that any other vehicle (RSA, n.d.). The 1.5% of registered vehicles also does not account for the riders that do not take their motorcycle out during the winter or only once a week. This means that statistic would only be worse if this was accounted for.

## Project Description

Motorcycle Dashboard is a geographically accurate alternative to a motorcycle speedometer that also allows you to track you route and give you useful information at different points on the interactive map. This app will open in the main page and start working as a speedometer and will not need any user input to start working for the default functionality to work. One of the buttons will let the user to track their trip and view details at different points such as their speed and lean angle. This is done by using the google maps API which has almost all the functionality to do so.

As the user is driving, their speed, current lean angle, average speed and top speed will update while the app is open. The user can also change their units from metric to imperial whenever they please.

Once their trip is over, the user can view their interactive map and select different points plotted and its values at those certain points. This will allow the users to further see where they may be going faster in some sections of the road that they should not be, giving the user more insight into their bad habits.

Graphical user interface, application

Description automatically generated

Figure 2 - App's first design prototype

The main complexity of this whole project will be the implementation of the current lean angle of the motorcycle. There are two ways I can go for the implementation of this, either using the phone’s built in gyro sensors or by using the current speed of the motorcycle, the angle of the bend ahead and using some mathematical formula to calculate the lean angle this way. Using the phone’s built in sensors seems to be the easiest and most obvious choice but I am worried that the vibrations of bumps and the engine may throw off the accuracy of these sensors.

Another complexity in this project was formatting the interface of the android app to facilitate different size and resolution phones that may be using this app. My approach to solving this was to test the app on as many different phones I could get my hands on and test that the design did not get moved around.

## Project Aims and Objectives

My main aim for this project is to have a fully functioning speedometer that is accurate and allows the user to view what speed they were doing at different points on the route thereafter. Through this, I am hoping the user can review their speeds and reflect on where they can improve over the course of the trip.

A goal for this app is to give users the ability to see where they could possibly slowdown in danger areas or on roads that are more prone to accidents and collisions. This can be shown in the “Review Routes” area of the app by utilising the Google Maps API.

Another aim is to have the user’s previous routes accessible from the database and not from the local machine, meaning they have access of the routes on other devices.

* Design different prototypes and receive opinions.

By doing this, I can receive feedback on how the designs and prototypes can be improved upon.

* Utilise the Google Maps API.

This will give the user the familiar view of Google Maps to review their routes.

* Ensure the speedometer readings are accurate.

Make sure that the speed and average speed that is displayed in real time is accurate.

* Implement Top Speed, Average Speed and Lean Angle.

Allow user to view their Top Speed, Average Speed and Lean Angle during their route and after in the map section.

* a

## Project Scope

The aim of this project is to make motorcyclists more aware of the speeds they may be exceeding while driving, not to encourage them to get the highest top or average speed. The app does not condone dangerous or excess of speed on public roads as these activities should be done solely on a race track.

## Thesis Roadmap

### Research

This chapter will show the importance of easing off speed when it comes to collisions involving a motorcycle, for both motorcyclists and other road users. I will be cross-examining and evaluating the various statistics of road collisions published by the RSA and other recognised publishers. This chapter will also show other types of research completed for this project.

### Design

This chapter delves into the methodology chosen for this project and how these choices came to be. I will also be showing detailed use-cases and personas related to the system will be presented. Finally, the designed technical architecture and software testing plans will be discussed.

### Development

This chapter will show each step of the development process for the app in relation to the technical architecture that was portrayed in the Design chapter. Some challenges that arose will also be shown in this chapter.

### Testing and Evaluation

This chapter will show how the testing and evaluation of the app and system was done. I will explain in detail how each iteration of testing was carried out, there will also be the user evaluation’s that were carried out from the testing phase. The system will also be evaluated to see if it accepts Nielsen’s Heuristics for User Interface Design.

### Redevelopment

This chapter outlines some of the development steps taken as a result of the feedback gained from the user evaluation. The changes made and the importance of these changes will be examined.

### Conclusions and Future Work

This chapter will reflect on the entirety of the project and will discuss the conclusions drawn, personal reflections made, and the future work planned for the project.

# 2. Literature Review

## 2.1. Introduction

In this chapter, I will show key areas of research that are imperative to this project. Many of these topics will explore the different aspects of safe driving; different campaigns run by national road authorities; designing a suitable user interface and experience.

## 2.2. Alternative Existing Solutions

There is no question there are many different speedometer apps on the market available to download. A lot of these apps are very outdated or have so much happening on the screen you don’t know where to look, which is not what you want to be doing while driving.

Here are some examples of similar apps:

### GPS Speed – Joao Silviera

GPS Speed (Silviera, 2014) is a very simple and aesthetically pleasing GPS Speedometer app that shows you different values that may be of use to you when driving such as speed, distance, average speed and top speed. The UI of this app is simple and pleasing to the eye and should not distract the user while driving.

This is my favourite app that I found while researching other speedometer apps and was my main inspiration when developing my app. My app functions very like GPS Speed but has implemented key features for motorcyclists in particular. It was also free which means it is available to anyone who may need it to see if their speedometer is accurate or do not have a functioning one.

A picture containing text, device, meter, gauge

Description automatically generatedA screenshot of a video game

Description automatically generated

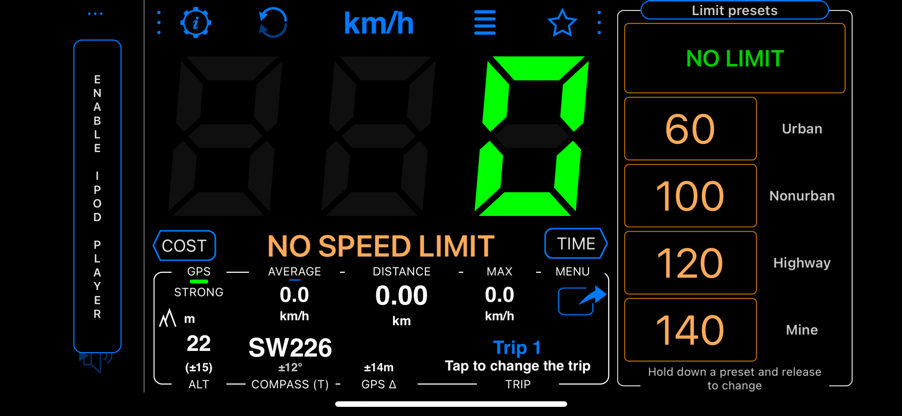
Figure 3 - GPS Speed Landscape

Figure 4 - GPS Speed Portrait

### Speedometer 55 – Stanislav Dvoychenko

Speedometer 55 (Dvoychenko, 2012) is a very informative and sophisticated speedometer app that prioritises functionality over visual aesthetics as seen in these screenshots, there is a lot going on and it may distract the driver.

In my opinion, this app has way too much functionality to be used day-to-day by a user and needs to be more aesthetically pleasing for use.

Graphical user interface, application

Description automatically generated

Figure 5 - Speedometer 55 Landscape

Figure 6 - Speedometer 55 Portrait

## 2.3. Technologies you’ve researched

### Mobile Technology

Android is the most used phone OS in the world today with an astonishing 70.5% of the world’s phone users (Anon., 2021). The language I will be using for the Android development will be Java, which is the go-to programming language when working with Android.

The only reason I did not choose to create my app in IOS is because of all the restrictions Apple have in place for development with their devices. In order for me to have created by Motorcycle Dashboard app, I would have to apply for development licences to have access to various phone sensors including the GPS.

### Java

I am using Java as my main programming language in relation to this app as I have some prior experience using Java to create Android apps and it is also the most used language for Android application development.

A picture containing icon

Description automatically generated

Java was created by James Gosling, Mike Sheridan and Patrick Naughton in June 1991 and was designed with the style and syntax of C and C++. Java is concurrent, meaning it has the ability to run many statements at the same time instead of running sequentially like its C counterpart.

Java is an Object Oriented Programming language and is the second favourite language in the world, second to Python. Java’s OOP element makes it easy to implement new features and also is very easy to use various APIs such as the Google Maps API.

### Android Studio

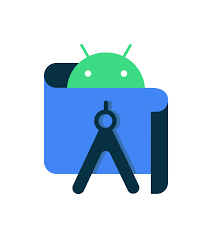
Android Studio is the official Integrated Development Environment for the Android operating system and is designed specifically for Android development. Android Studio makes Android development very easy and allows you to test the app on virtual machines which can help when designing for different devices. The software is also very customisable to whichever layout you prefer, this feature is called the Android Virtual Device (AVD). As you can see in the figure below, it runs the app on the virtual phone with full functionality present.

A picture containing text, electronics, iPod, screenshot

Description automatically generated

Figure 7 - AVD of MotorcycleDashboard

The only problem with using the virtual phone for testing is that I could not test the speedometers functionality as that needs movement, until I found the extended controls section in which I was able to plot a route and change the speed at which it went along the route.



### APIs

An API is an application programming interface, which allows connections between the user’s machine to a remote machine which processes information and can be used alongside applications for different functionality. When it comes to APIs, I will be using is the Google Maps API when plotting the route and allowing the user to view their previous routes interactively using Google Maps.

This API has a lot of functionality when it comes to displaying maps and fits my criteria perfectly when I was designing my app. For my project, I used the maps and markers functionality to display information of different points on the user’s trip. I was worried when it came to implementing it as I thought it would have been difficult and would need to jump through many hoops to get it to function as I would like but it turned out to be very easy to use.

Chart, scatter chart

Description automatically generated

Figure 8 - API Manager

Google’s API console is very easy to use and understand, it also gives people who want to test using an API on a small scale such as this system to be used for free.

### GPS Speedometers

GPS Speedometers are a very accurate way to calculate speed in comparison to physical speedometers you would find in a vehicle as they can over time, become less accurate and may cause you to be over the speed limit when you think you are not. GPS Speedometers calculate the distance between two GPS locations and use this distance and the time taken between the two points to calculate what speed you are going, because of this, GPS Speedometers are more accurate in higher speeds in comparison to lower speeds as there is a greater distance between the two GPS locations that it is using calculating the speed from.

## 2.4. Other Research you’ve done

### Lean Angle Calculation

Calculating the lean angle was one of the tougher functionalities to implement as since 2013, Android Studio depreciated the ability to access the phones standalone tilt sensors. Because of this, the only way to get the rotation of the phone was to calculate Azimuth, which is the angular measurement in a spherical coordinate system.

To do this, I had to calculate Azimuth by using the phone’s accelerometer and magnetometer. This also proved to be an issue for me when testing as the phone I was using to test my app was made without a magnetometer, which made the lean angle impossible to calculate and would be the same for any older phones that also do not have a magnetometer.

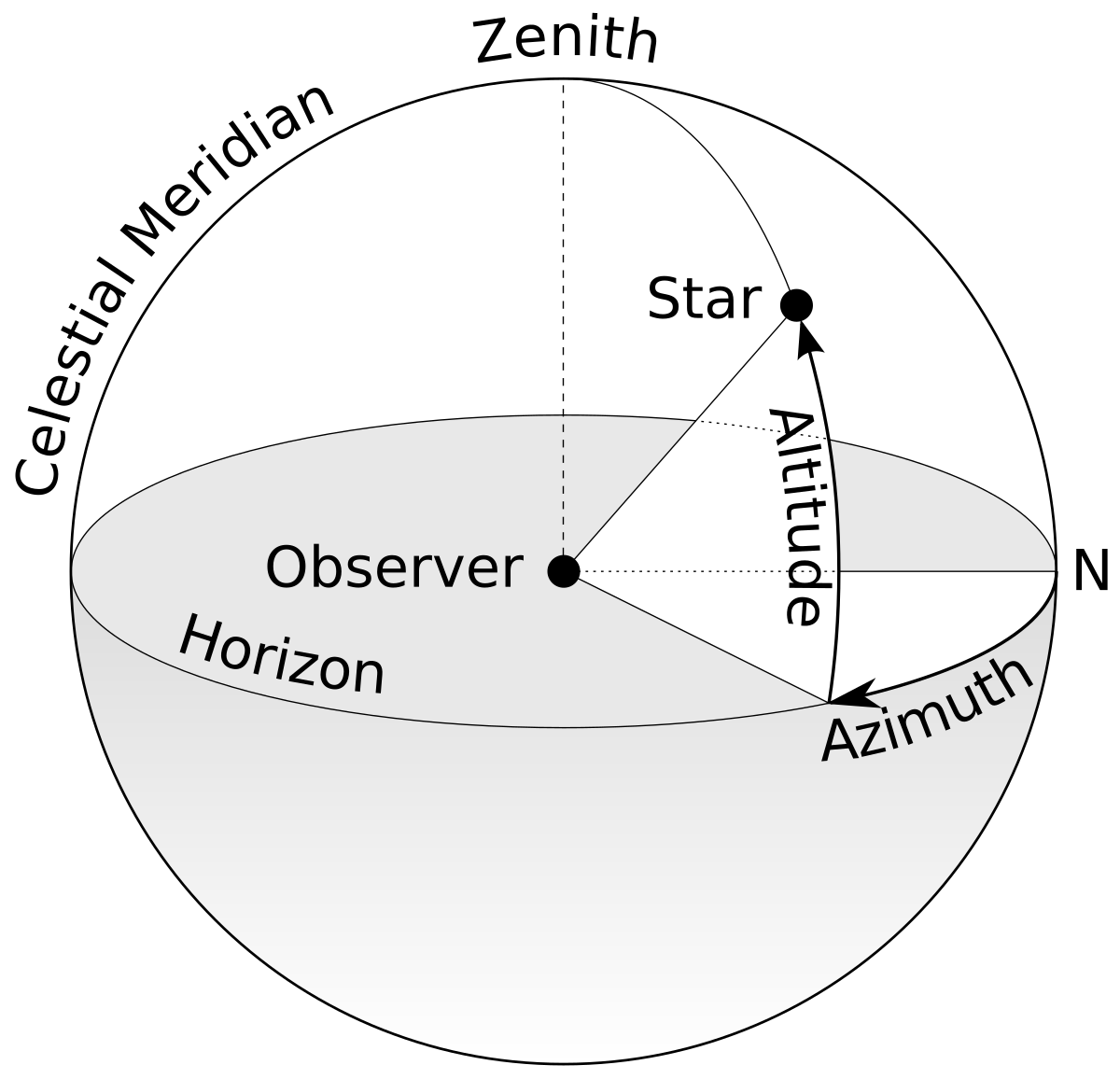


Figure 9 - Azimuth

## 2.5. Existing Final Year Projects

I have researched two previous Final Year Projects that have similarities with my project. These two projects also have geographic location plotting and speedometer functionality.

### SmartDrive: Exploring the Use of GPS in Order to Track Speed and Promote Safe Driving – Richard Meyer

The goal for this project is to provide a user with real-time feedback and a comprehensive breakdown of how they are driving (Meyer, 2021). The user is alerted while driving if their speed is more than the posted limit on the road they are on, this makes the user more aware of their speed and promotes slowing down, a result being a lesser chance of an accident occurring.

This app also had a similar map feature in comparison to my app and showed the route the user took. The main difference between my app and Richard’s is that the user can view different information of specific points on my app while also seeing what route the user took.

Richard used a full stack Progressive Web App architecture which gave him the ability to use it on both mobile phones and on a desktop. When I was first designing and deciding on what architecture I had considered this approach but decided against it as there is no reason this app will be used on a laptop or desktop as that would defeat the purpose of a speedometer app.

This app also had an interesting feature as it would light up the screen to alert the user that they are driving over the speed limit and to entice them to slow down, I had thought of adding this feature to my own app but decided against it as motorcyclists should not be distracted from what is ahead for even a split second.

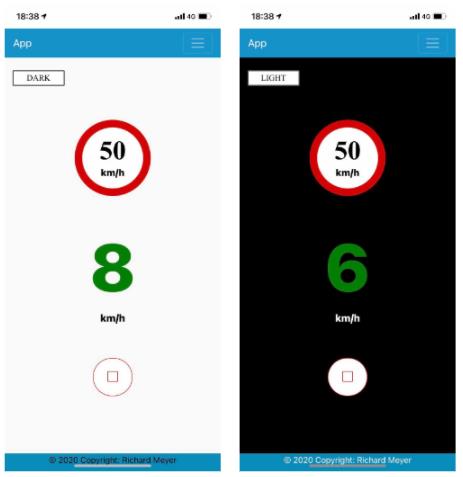


Figure 10 - SmartDrive

### Dynamo Dublin: Real-Time Bus Arrival Predictions – Anton Medves

The objective of this project is to develop a real-time bus arrival prediction model using GTFS- R data (Medves, 2021). This project allows the user to give an estimate on how long their public transportation will take to arrive to their stop. Anton does this by using the geolocation of the public transportation and how long it usually takes from that location to the stop.

Graphical user interface, text, application

Description automatically generated

Figure 11 - DynamoDublin

## 2.6. Conclusions

Through all of my research stated above, I gained a lot of valuable information and knowledge on what technologies and architecture would suit my project. I also gained a lot of knowledge from previous student projects and how they overcame problems in their system. With all of this information and knowledge, I started my design stage with it all in mind.

# 3. Experiment Design

## 3.1 Introduction

In this chapter, I will be explaining 4 separate sections, ranging from Software Methodology; the sequential approach that will be taken for the overall development of this system which includes the app, database and other likewise aspects, Overview of System; A full description of all aspects of the system, Other Sections and Conclusions.

## 3.2. Software Methodology

### Agile Methodology

Agile Methodology uses an incremental approach, meaning it is flexible and different stages of development can be revisited further down the line instead of using a methodology such as a waterfall where once one stage is completed, it should not be revisited, giving Agile Methodology a lot more flexibility.

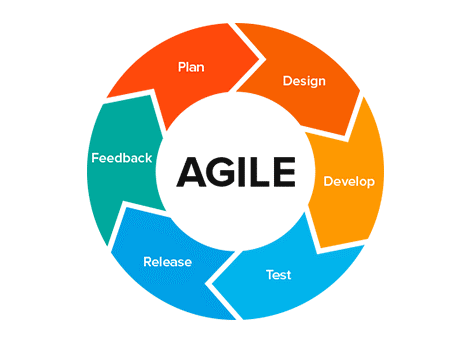


Figure 12 - Agile Methodology

Agile Methodology allows for testing to be carried out during the whole development stage and allows for bugs to be caught and fixed more efficiently. This type of development is more user focused and means user feedback can be implemented much more frequently instead of at the end.

It is important to ensure that when using Agile methodology that the project does not deviate from the original plan.

I believe that an Agile Methodology would suit this project and more specifically, using a Scrum Framework. This focuses more on sprints as there will be many features added to this project during its lifetime in development.

## 3.3. Overview of System

The current development approach for this project is as follows:

1. Design and develop a basic functioning GPS speedometer app.
2. Design a new feature.
3. Develop new feature.
4. Test the new feature.
5. Repeat steps 2-4.

This development approach worked very well by using an iterative design and development strategy and follows the Agile Methodology.

As seen in the diagram below, I have portrayed the technical architecture that will be applied to the system which is a three tier system consisting of the Presentation, Middle and Data layer.

Diagram

Description automatically generated

Figure 13 - Technical Architecture

## 3.4. Front End

The systems UI (User Interface) is located in the presentation layer. This layer strictly deals with the user and how they interact with the system without having direct access to the middle of data layer.

Paper Protypes were created for the first iteration of what a low fidelity design of the app may look like. These prototypes were very basics, showing the core functionality of the dashboard in action.

The goal for the main page is to make it as basic and not distracting to the user as they drive. In comparison to the Speedometer 55, I do not want to fill up the page with unnecessary information that may focus the drivers attention off the road. Some design personalisation such as night mode and different colour themes are optional to the user.

A picture containing text, whiteboard

Description automatically generated

Figure 14 - Horizontal low fidelity

A white piece of paper with writing on it

Description automatically generated with low confidence

Figure 15 - Portrait low fidelity

These low fidelity prototypes give the basic design on what I wanted the app to look like and as seen as the first design prototype and the final design, I have kept most of low fidelity design in mind and have added the max lean angle for left and right but have also removed the tripometer as that is a future design I intend to implement.

A picture containing text, electronics, iPod, screenshot

Description automatically generated

Figure 16 - First Design Prototype

Graphical user interface, application

Description automatically generated

Figure 17 - Final Design Landscape

Graphical user interface, application

Description automatically generated

Figure 18 - Final Design Portrait

With the final design, I have implemented all the functionality I wanted at this stage and am very happy with the final outcome of what users will be using. There is still plenty of space on the home page for future features such as the tripometer.

### Use Case Diagrams

Diagram

Description automatically generated

Figure 19 - Use Case Iteration 1

Diagram

Description automatically generated

Figure 20 - Use Case Iteration 2

## 3.5. Middle Tier

The middle tier is what will be the communication between the users inputs and how that gets translated into the core functionalities which then go to the database in the data layer. Some of these functionalities include plotting the users points on the map and also recording their speed and lean angle.

## 3.7. Conclusions

In this chapter I looked into the design of the system and different development approaches when it comes to design. Overall, this chapter gave me good insight into the process of designing a system and the different aspects of how it is done, while also giving me the knowledge of making use diagrams which proved to be very helpful in the development stage as a point of reference

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Priority** |
| Speedometer | Fully functioning and accurate speedometer | HIGH |
| Lean Angle | Fully functioning and accurate lean angle display | HIGH |
| Map Plotting | Plot the locations along the route that can be accessed after the ride | HIGH |
| Average speed | Show user their average speed for the current trip | MEDIUM |
| ‘Code Red’ | Emergency functionality for user to contact specified contacts in case of urgency | MEDIUM |
| Fuel level | User can enter their average fuel consumption (Km/L) and the app will notify them if they may be low on fuel | MEDIUM |
| Top speed | Show the user their current top speed on that trip | MEDIUM |
| Change Colour Theme | Allow user to change the colour theme of the app | LOW |
| Night Mode | Darker display for when riding at night | LOW |

# 4. Experiment Development

## 4.1. Introduction

This chapter will outline the process of the development of the system and each iterative stage. Outlined will be the key stages of development and also the challenges that were faced.

## 4.2. Experiment Development

The first step in beginning the experiment development process was to set up a backup or repository in case of loss of progress and also to share work. I set up a public GitHub repository and have been committing regularly to ensure no loss of progress. It is also possible to link Android Studio with GitHub to ensure fluid commits without hassle.

Graphical user interface, application

Description automatically generated

Figure 21 - Horizontal prototype design

I used XML code to design and format the horizontal design as such, but this is only the first of probably many design but I do intend to keep it simple and not distracting. I decided to make the first design prototype in landscape as this is how I normally have my phone mounted on my motorcycle and also seems like the most popular for other motorcyclists.

### 4.2.1. Speedometer functionality

Graphical user interface, text, application, Teams

Description automatically generated

Figure 22 - GPS Speedo Code

The Speedometer was the core functionality of the app so I ensured that I took my time and made sure it was in perfect working condition before continuing on with other features like Top Speed and Avg Speed since both of these features need the speedometer to be accurate.

Before running the actual function to initialise the speedometer, the app checks if the necessary permissions are granted and if not, it prompts the user to allow these permissions. There are also statements that check if these permissions have been granted and will return a message to the user saying these are not granted and will prompt the user to allow them.

Here, I have coded the speedometer functionality by accessing the user’s current location and using the built in phone function “location.getSpeed”. Android has a many functions like this that can help such as distanceTo and many others. Once the user starts moving, the digital speedometer changes in real time with what speed the user is going.

Icon

Description automatically generated

Figure 23 - Speedometer

I also added a switch which allows the user to change from KM/H to MP/H. This switch essentially runs the speed in kilometres through a calculation that converts it to miles.

public float getSpeed() {  
 float nSpeed = super.getSpeed() \* 3.6f;  
  
 if(!this.getMetricUnits()) {  
 //metres/sec to mph  
 nSpeed = super.getSpeed() \* 2.23693629f;  
 }  
  
 return nSpeed;  
}

Calculating the top speed was very simple to implement by running an if statement in the updateSpeed function and checking if the current speed is bigger than top speed variable and thus reassigning the top speed variable and changing the text on the screen.

if (nCurrentSpeed > TopSpeed) {  
 strTopSpeed = strCurrentSpeed;  
 TopSpeed = nCurrentSpeed;  
 TV\_TopSpeed.setText("Top speed: " + strTopSpeed + " km/h");  
}

In order to calculate the average speed of the motorcycle, I used a list and added each current speed iteratively while the motorcycles speed was greater than zero in order to exclude any deviation while the motorcycle is at a stop. The average is then calculated by taking the average of the avg\_speed\_list and thus displaying it onto the home screen.

Graphical user interface

Description automatically generated with low confidence

Figure 24 - Avg Speed

if (nCurrentSpeed > 0) {  
 AvgSpeedList.add(nCurrentSpeed);  
 Double average = AvgSpeedList.stream().mapToDouble(val -> val).average().orElse(0.0);  
 Formatter fmt4 = new Formatter(new StringBuilder());  
 fmt4.format(Locale.*US*, "%5.0f", average);  
 strAverage = fmt4.toString();  
 TV\_AvgSpeed.setText("Avg. speed: " + strAverage + " km/h");  
}

### 4.2.2. Maps functionality

The aims of the map and route taken functionality were to show the user their route once it is completed and be able to view every point along the route and to show details of speed and lean angle at those specific points. To do this, I created two array list, one that contains each map marker and another to contain the information.

To do this, I made an if statement to add a marker and information of speed and lean angle every 50 metres. This distance can be changed by the user in the settings page in case they would like information more frequently or less frequently through the settings page. The reason I put this option in is because these markers would take up a big chunk of phone memory if the user was to take a much longer trip which may in turn crash the app and all of the routes current data.

As you can see between these two screengrabs, you can see the difference between setting markers every 50 metres to every 100 metres.

Map

Description automatically generatedMap

Description automatically generated

Figure 25 - Map every 50 metres Figure 26 - Map every 100 metres

Once the user is finished with their journey, they can view their map and their progress by clicking one of the markers, it will display information at that certain point. This function was very easy to implement because of the simplicity of using the Google Maps API.

for (int j=0; j<MainActivity.*GlPointers*.size(); j++) {  
 options.position(MainActivity.*GlPointers*.get(j));  
 options.title(MainActivity.*GlDesc*.get(j));  
 options.snippet("someDesc");  
 mMap.addMarker(options);   
}

### 4.2.3. Lean Angle functionality

Calculating the lean angle of the motorcycle was always going to be one of the most difficult aspect of this project and was especially difficult since Android Studio depreciated the ability to access the phone’s tilt sensors in 2013. In order to get the phones degree of tilt, I would have to calculate Azimuth with the phone’s accelerometer and magnetometer. This essentially uses a compass to see what direction the phone is pointing towards and uses the accelerometer. I was able to do this with this function:

if (event.sensor.getType() == Sensor.*TYPE\_ACCELEROMETER*)  
 mGravity = event.values;  
if (event.sensor.getType() == Sensor.*TYPE\_MAGNETIC\_FIELD*)  
 mGeomagnetic = event.values;  
if (mGravity != null && mGeomagnetic != null) {  
 float R[] = new float[9];  
 float I[] = new float[9];  
 boolean success = SensorManager.*getRotationMatrix*(R,I,mGravity,mGeomagnetic);  
 if (success) {  
 float orientation[] = new float[3];  
 SensorManager.*getOrientation*(R, orientation);  
 azimuth = orientation[1]; // orientation contains: azimuth, pitch and roll  
 }  
}

Once I implemented Azimuth as the lean angle, it worked perfectly when testing with the AVD extended controls which allowed me to change the axis of rotation on the X axis.

Graphical user interface

Description automatically generated

Figure 27 - Lean Angle

### 4.2.4 Saved Instance State

I encountered an issue when rotating the phone from portrait to landscape and vice versa, as changing orientation would reload the activity and would reset the top speed and average speed. I found save instance state to be very essential for this reason.

protected void onSaveInstanceState(Bundle outState) {  
 super.onSaveInstanceState(outState);  
 outState.putFloat("float\_value", TopSpeed);  
 outState.putString("string\_value", strAverage);  
}  
  
@Override  
protected void onRestoreInstanceState(@NonNull Bundle savedInstanceState) {  
 super.onRestoreInstanceState(savedInstanceState);  
 TopSpeed = savedInstanceState.getFloat("float\_value");  
 strAverage = savedInstanceState.getString("string\_value");  
  
 TV\_TopSpeed.setText("Top speed: " + TopSpeed + " km/h");  
  
  
 if (strAverage == null) {  
 TV\_AvgSpeed.setText("Avg. speed: " + "0" + " km/h");  
 } else {  
 TV\_AvgSpeed.setText("Avg. speed: " + strAverage + " km/h");  
 }  
  
}

Now, when the app’s orientation is changed, onRestoreInstanceState restores the variables and text of top speed and average speed.

### 4.2.5 Convert Kilometres to Miles

In order to convert Kilometres to Miles, I ran a check to see if the change units switch was on and if so, it ran through a function that converted it mathematically.

public float getSpeed() {  
 float nSpeed = super.getSpeed() \* 3.6f;  
  
 if(!this.getMetricUnits()) {  
 //metres/sec to mph  
 nSpeed = super.getSpeed() \* 2.23693629f;  
 }  
  
 return nSpeed;  
}

This can be changed at any point during the trip by the user when they please by pressing the Metric switch.

Graphical user interface, application

Description automatically generated with medium confidence

Figure 28 - Metric Switch

### 4.2.6 XML Layout

The XML Layout was challenging to format correctly for all different devices shapes and sizes so I created a variant of the portrait layout and made a ‘land’ XML layout which is called when the phone is put into the horizontal position. This means I was able to create independent layouts for portrait and landscape without worrying about the format being jumbled up and buttons being in the wrong place.

Graphical user interface, application

Description automatically generated

Figure 29 - Layout Land

## 4.3. Conclusions

Having implemented all of the key features, I have achieved my goal for creating a fully functioning speedometer that has extra key features for motorcycles. The development phase of my project has taught me so much about both Java and working with the Google Maps API. I have really enjoyed this phase and has solidified the fact of Java being my favourite language to use in development.

|  |  |  |
| --- | --- | --- |
| Feature | Days (estimate) | Days (actual) |
| Speedometer | 7 | 5 |
| Maps | 10 | 6 |
| Settings | 2 | 1 |
| Top and Average Speed | 1 | 2 |

# 5. Testing and Evaluation

## 5.1. Introduction

In this chapter, I will explore the method of testing this system, the timeline of testing and also the findings of the system testing.

## 5.2 System Testing

### 5.2.1. Blackbox Testing

The main method of testing I used was to ask groups of fellow motorcyclists to use the app and give their thoughts and also if they have any problems or bugs in the append took their findings into consideration in the development of my app.

This was a way of Blackbox testing as this user group did not know the structure or design of the app as they would only be interacting with the app and not how it is run. This gave insight into aspects such as potential bugs, interface problems such as formatting for different devices and also accuracy and performance if the speed of the speedometer app does not correlate to the speedo on their motorcycle.

I have a group of motorcycle friends that I see every Thursday evening and Sunday morning for a group ride. I asked for whoever had an Android phone if they would test my app and report back any bugs they encountered or any ideas on how to improve on the app. There were about 15 people who agreed and downloaded the app to use and of those 15, I got frequent responses from about 10 about bugs and suggestions.

To give the test group access to the app, I uploaded the APK file onto a google drive folder and shared the link with the test group. They were able to download the app through the APK with a click of a few buttons. I also added the test checklist to ensure all features worked. I was also able to upload updated versions of the app and notify them that they can download it through the previous link I sent them.

One of the suggestions that I received was to change the max lean angle into two separate right and left max’s. By doing this, the user is able to see which side they are more comfortable leaning and my entice them to be more conscious of what side is more confident for them.

I also gave each user this checklist below to ensure all of the core functionalities work properly and all came back to me with passes.

## Test Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Test No.** | **Test Description** | **Expected Outcome** | **Pass?** |
| 1 | Does the phone app load when the icon is selected? | The app will load properly and bring the user to the login screen. | Pass |
| 2 | Does the app close when the app is no longer being used? | The app will shut down properly when the app is unused. | Pass |
| 3 | Request device location | The app should request the device’s location when using the app. | Pass |
| 4 | Are route points being plotted correctly? | The points on the route should be accurate and correct. | Pass |
| 5 | Does the information display on each point? | Speed and Lean Angle should display on each point. | Pass |
| 6 | Speedometer working properly? | The speedometer is accurate and shows real-time speed. | Pass |
| 7 | Lean angle working properly? | The lean angle is accurate and shows real-time angle. | Pass |
| 8 | Top speed and Average speed work properly? | Top speed and Average speed should change respectively during the trip. | Pass |

## 5.3. Evaluation

Evaluation is a key component for the system and is as important as testing is to the system. Without evaluation, testing a system would be useless as nothing would come of testing and finding and improving issues found during the testing phase.

Most of the evaluation will be the comparison of the Motorcycle Dashboard against other similar speedometer apps and will see the pro’s and con’s between them.

## 5.4. Conclusions

In this chapter, I outlined the testing and evaluation that took place over the lifetime of this system. Both testing an evaluation were carried out by the projected users of this app, motorcyclists, using Blackbox testing.

# 6. Conclusion and Future Work

## 6.1. Introduction

This chapter will go into detail of the different issues encountered so far with the project and also the issues that are most likely to be a problem in the future. I will also be detailing the future work and what things are planned to be done at what stages.

## 6.2. Conclusions

### 6.2.1 Literature Review

The first aspect discussed in the literature review section was alternative solutions to my problem. While there were a number of applications that solved my problem, none of them were catered towards motorcyclists. These apps also had their own flaws in design and functionality. Some apps also cost money, which should not be the case as these apps are not, in my opinion worth it and should be free for all to use.

I also researched different technologies that would in turn benefit me in the development of this project, some of which I already had previous experience with and some of which were brand new to me and so this helped me greatly to experience new technologies and have a better understanding on how to use them. I learned through delving deeper into Android Studio that I could use extended controls to test my apps geolocation functionality instead of having to test it physically by going out on my motorcycle.

### 6.2.2 Motorcycle Dashboard Design

In the first stages of design, I look through many different design methodologies and decided that an Agile methodology would suit my project the best. This gave me a lot of flexibility in terms of development and testing and is the main reason why I chose it. Also since my application was built up with features, feature driven development was beneficial by breaking down the project into smaller pieces that could be developed and implemented one at a time.

The design of the user interface was shown through the different stages of low, medium and final prototypes and gave feedback from each iteration from users on how it was improved from each iteration.

### 6.2.3 Motorcycle Dashboard Development

In this section, I discussed the development process over the lifetime of the project and many of the different features I developed and implemented. This part of the project was the most interesting for me. I went into detail of the various challenges and ways of developing the more difficult features such as the GPS speedometer, lean angle and the plotting of data points along the route taken by the user.

### 6.2.4 Motorcycle Dashboard Testing & Evaluation

In this section, I explained the various methods of testing that took place to ensure the app was satisfactory by the users and also that it was fully functioning as intended. Through creating test plans, I was able to gain valuable information from the test group on many aspects such as bugs and any future improvements that they would like to see in the application.

After the testing phase, my application had improved enormously through the improvements found by the test group and proved to be a very integral part of the development process.

## 6.3. Future Work

One of the many benefits of working with an Android application is the enormous amount of features that I can implement further down the line without much hassle at all. Motorcycle Dashboard has a solid ground for making future improvements and features to be added and also refinements to the current features. The potential growth for my application is explained in two sections below; System Improvement and Future Features. Here I will discuss the future improvements that can be made to existing features and the possible additions that can be made to the application.

### 6.3.1 System Improvements

The main improvement I would consider doing would be the maps

### 6.3.2. Future Features

### 6.3.3. GANTT Chart

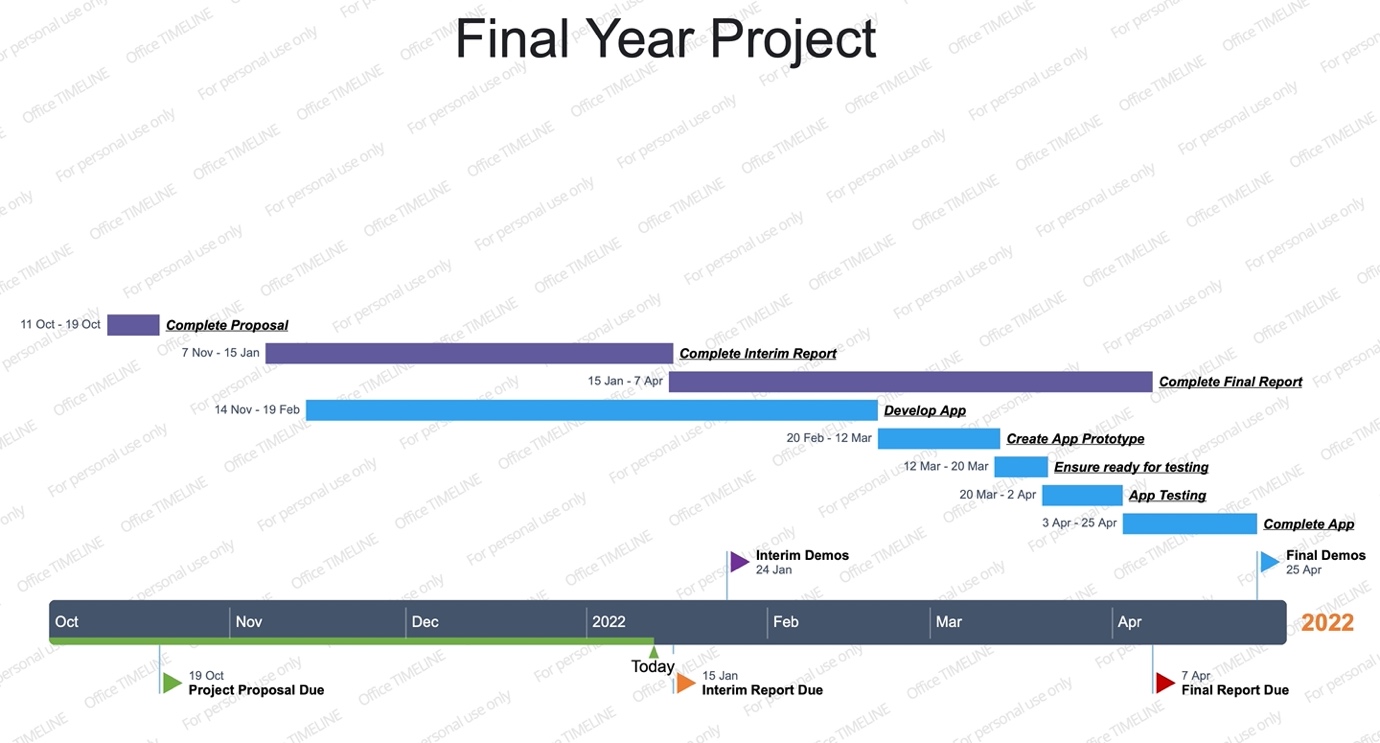


Figure 30 - GANTT Chart

# Bibliography

Medves, 2021. *Dynamo Dublin - Real-Time Bus Arrival Predictions,* Dublin: s.n.

Meyer, 2021. *SmartDrive: Exploring the Use of GPS in Order to Track Speed and Promote Safe Driving,* s.l.: s.n.

RSA, n.d. *RSA.ie.* [Online]   
Available at: https://www.rsa.ie/road-safety/campaigns/motorcycle-safety  
[Accessed 20 December 2021].

RSA, n.d. *RSA.ie.* [Online]   
Available at: https://www.rsa.ie/news-events/news/details/2019/05/01/motorcyclists-reminded-to-ease-off-the-throttle-as-speeding-a-factor-in-half-of-motorcyclist-deaths  
[Accessed 20 December 2021].

Silviera, 2014. *GPS Speed,* s.l.: Joao Silviera.