Mobile Guidebook for a Scenic Bus Route

Student Name: Cansu Karaboga
Student Number: 170516130
Supervisor: Dr John Colquhoun
Submission Date: 24.08.2018
Word Count: 14663

DECLERATION

This dissertation is submitted to Newcastle University as part of the MSc Computer Science program. Unless stated otherwise, the work belongs to the author.

ABSTRACT

This paper focuses on building a hybrid and cross-platform mobile application that acts as a guidebook for a scenic bus route. The application was built with open-source software and the main technologies used were PhoneGap, Leaflet, HTML, CSS and JavaScript. The paper touches on software engineering life-cycle and principles by following the waterfall model; giving the reader a high-level explanation. Moreover, the paper includes the experiences of the novice developer who built the application. The end product is mostly successful; able to inform the passengers of the bus on upcoming point of interests. However, fails to satisfy some of the requirements such as implementing an offline map.

ACKNOWLEDGEMENTS

First and foremost, I like to express my gratitude to my supervisor Dr John Colquhoun for supporting my interests by allowing me to work with open-source software, for making himself available to answer my questions and for having an encouraging and positive attitude throughout the dissertation period. I also like to thank to the entire open-source community and contributors for their efforts and good work.

Table of Contents

Table of Figures

1 INTRODUCTION

Cumbria Classic Coaches is a small, family owned, and operated bus company. They operate in rural areas in the north of England and run five buses a week with each bus having a unique route. The company currently has a website where they display basic information about themselves, their services and detailed information about the routes they run. However, the company does not own a mobile application. This paper focuses on the journey of building a mobile application for Cumbria Classic Coaches by a novice developer. The mobile application will focus on Route 572, which is one of the five routes operated by the company. Route 572 operates on Wednesdays and runs through Ravenstonedale to Bernard Castle [9]. Route 572 is unique as it caters to both tourists and locals by combining touristic sites and shopping. Therefore, its passengers are consisting of tourists who may be visiting heritage sites as well as locals who may be out for shopping [9]. The mobile application is meant to guide the bus passengers on forthcoming points of interest along the route. The information that is provided on the mobile application is not limited to bus times/schedules. Contrarily, the main focus of the content is on informing the user on upcoming points of interests and providing them with detailed information regarding heritage sites, the history of the villages and towns, natural parks, and anything else that may be of interest to both tourists and locals

1.1 Aim of the Project

This project aims to build a hybrid cross-platform mobile application without any proprietary software for Cumbria Classic Coaches. Given the family and community-oriented nature of the company, this project is intended to be open-source to match with the company values.

1.2 Objectives

The necessary steps that are needed to be taken in order to build this mobile application are as follow:

- Background research will be constructed on Cumbria Classic Coaches, mobile applications, open-source software, and various technologies to build a hybrid cross-platform mobile application.
- Requirements will be established and appropriate techniques will be used to design the system
- Most appropriate software engineering model will be chosen to deliver the best application possible with the resources on hand.
- New tools and languages will be learned in a short period of time to build the application.
- The end product will be tested and evaluated to make sure it conforms to the requirements.

1.3 Expected Outcomes

This project is expected to deliver a fully functioning, hybrid cross-platform mobile application where the user can download it from an application store of their choice. The final product is expected to be available for both Android and iOS. It is also hoped that the application will work on less popular operating systems as well, such as Windows mobile, PureOS and other niche Android-based operating systems. A fully functioning application should detect the user's location and inform them by means of notification and/or with in-app pop-ups that they are approaching to a point of interest. The application should also provide the end user with detailed information such as history, things to do and see on the approaching point of interest. It is expected that the application will be used by people with various abilities and from different age groups. Consequently, the design of the application should be as inclusive and as easy to use as possible. Given the fact that this application will be used in rural England, low signal coverage is expected. Hence, a more advanced application will also be functioning when there is no data coverage or signal. Although not the main goal of the project, it is hoped that this mobile application will help with the business growth of Cumbria Classic Coaches by increasing the company's engagement with the customers and giving the passengers a more interactive bus ride.

1.4 Outline

- Chapter 2 will explain what the mobile application is, introduce the Cumbria Classic Coaches and Route 572 in more detail and present a critical analysis of existing mobile applications. It will also inform the reader on tools and technologies that will be used to build this application and the reasons for choosing them.
- Chapter 3 will focus on the software engineering life-cycle, which will include requirement analysis, design, implementation, testing and maintenance.
- Chapter 4 will present the test results and evaluate them. It will give a critical feedback on the software engineering approach that was adopted for this project. It will also reflect back on the project, suggest future improvements.
- Chapter 5 will explain how well the objectives were met and discuss the valuable lessons that were learned while building the application.

2 BACKGROUND RESEARCH

This chapter will review background material that is relevant to the project. First, the mobile application is introduced, later a focus is given to gaining a deeper understanding of the company and route 572 that the application is to be built on. This is done so that the developer who is not from northern England can get to know the geographical area, the towns and villages that route 572 covers in order to build a realistic application with relevant information. Later, existing mobile applications that are tourism and navigation oriented is reviewed and analyzed. The emphasis was given on both tourism and navigation related applications because Cumbria Classic Coaches is not necessarily a tourism company. To locals, the buses can simply mean a basic means of transportation for weekly mundane grocery shopping [9]. However, to tourists that are using the bus, it is a sightseeing journey where a vintage bus is taking them to picturesque villages and historical places. Hence, the application that will be build for this project should encompass both navigational and touristic functionalities. It is expected that by reviewing existing mobile applications that has similar functionalities to the one that is to be built, there will be a better understanding of available technologies and tools to use, good design principles to adopt and possible functionalities to implement into the application. This chapter also discusses the technologies and tools that have been chosen to build an open-source, cross-platform mobile application. The chapter closes with a reflection on the entire research process.

2.1 Mobile Application

As defined in Cambridge Dictionary, an application software is a software that is designed to do a particular job, or for a particular user [11]. Today, application software may often popularly be referred to as 'application' or 'app'. Similarly, a software program that runs specifically on a mobile phone is called a mobile application or a mobile app [12]. It can be argued that Steve Jobs is the forbearer of mobile apps as we know them today. Back in 1983, in a talk he gave while envisioning an application store, Jobs stated: "…would be like a record store, where software would be downloaded over phone lines." [50]. However, Apple, the technology company co-founded by Steve Jobs was not the first to turn mobile applications into reality. Instead, nine years after the talk Steve Jobs gave, IBM became the pioneer with Simon, often referred as the first smartphone [1]. Simon came with built-in apps such as an address book, calculator, calendar, fax, mail and more. Steve Jobs did initially revolutionize the app world when he introduced the first iPhone in 2007 and App Store later in 2008. With the launch of App Store, for the first time, third-party developers were able to sell software on a mobile device and consumers were able to download software with no more than a few taps. Apple's App Store was seen as a big innovation that was revolutionary when it first came out with approximately 500 apps in the store. Ten years later, apps and app stores are a normal part of our digitalized lives, if not the norm. As of 2018, there are 48.52 million smartphone users in the United Kingdom alone and this number is expected to raise to 53.96 million by 2022 [53]. Only 36% of the world population uses mobile phones as of 2018 and five billion of the world population is expected to own mobile phones just by 2019 [52]. Hence, a steady increase in mobile device usage is expected to be observed throughout the upcoming years. Moreover, 205.4 million apps were downloaded worldwide in 2018 and this number is expected to rise to 258.2 by 2022 [54]. As the statistics suggest, mobile devices and mobile applications are here to stay and be part of humanity's common future. The increasing popularity of mobile usage brings a change of habit among users' internet usage. More and more users prefer to do their internet browsing on their mobile devices instead of their desktop or laptop computers. As stated by Statista, "in 2018, 52.2 percent of all website traffic worldwide was generated through mobile phones, up from 50.3 percent in the previous year." [51]. As a result, it is becoming necessary for companies that like to have an online presence to either have mobile friendly websites or a mobile app if not both. As a matter of fact, pretty much any big-name company already has a mobile app alongside with mobile friendly website. Given the ever-increasing importance of mobile applications in today's market share, it makes sense for small to medium size companies to have mobile presence as well. By having a mobile app as a small to medium size company, one can potentially build and improve brand recognition, stand out from the competition, reach out to younger demographics, cultivate loyalty among existing customers and give users easy access to information by being right at their fingertips [25].

2.2 Cumbria Classic Coaches

Cumbria Classic Coaches Ltd. is based in Bowber Head, Cumbria, Northern England. The company currently owns five single deck buses, two double deck buses and a vintage car [8]. The fleet is made up of all vintage vehicles, the oldest being a 1948 model AEG Regal named “Florence” and youngest being a 1991 model Leyland Olympian called “Big Red”. The entire fleet is available for various purposes, all year around, such as weddings, television and film props, birthday outings, anniversaries, school proms, corporate hospitality, themed and fancy-dress events and mystery tours [9]. However, the main focus of the company is their weekly bus tours that run on a timed schedule, taking both locals and tourists alike to heritage sites, various tourist attractions, shopping points, picturesque villages and towns in North West England [9].

2.3 Route 572

Route 572 stops at six different towns and villages. The journey starts at Ravenstonedale where the Cumbria Classic Coache’s bus depot is located. Ravenstonedale is a picturesque village in Cumbria, part of rural England with the population of approximately 500 people [18]. Golfing, horse riding, cycling, walking the nearby valleys and dales are among the activities that the village offers [43]. St. Oswalds is the only church of Ravenstonedale and may be an interesting visit for history and architecture lovers due to its three-decker pulpit and collegiate pews [43]. The church also has the excavation of Gilbertine ruins on its grounds [43]. There are also cottages, caravans and inns available for visitors to stay in or rent out. Fat Lamb Country Inn and Restaurant and Kings Head are among the available accommodations for overnight guests and Route 572 makes a stop in front of both of them. After Ravenstonedale, the bus heads to Kirkby Stephen. Kirkby Stephen is a market town that lies at the head of the Eden Valley. The town has a population of approximately 1822 people [18]. The market license was granted in 1352, and visitors can still enjoy the lively market every Monday. The town has various local shops, pubs, hotels, restaurants and bistros. Kirkby Stephen is surrounded by natural beauty such as the Yorkshire Dales National Park, North Pennines Area of Outstanding Natural Beauty and many farm lands [58]. The settlement history of Kirkby Stephen goes back to iron-age, so the town is rich with history, offering visitors a historic market town, Lady Anne Clifford Westmorland Heritage trail and the castle of Eden to discover [33]. Brough is a small village in the Eden district of Cumbria and the third location the bus visits on Wednesdays. It is made of twin villages; Church Brough and Market Brough. Church Brough lies on a former Roman road while Market Brough is on a medieval road [54]. Brough Castle which was built during the 11th century is a local tourist attraction. After Brough, bus stops at Middleton in Teesdale. Middleton in Teesdale is a small town with the population of approximately 1200 people, in county Durham [16]. This picturesque town is the first stop of Route 572, that is outside of Cumbria’s boundaries along with Eggleston and Barnard Castle. The town is surrounded by natural beauty, one of such is High Force Waterfall [35]. Eggleston is the fifth place on Route 572 and is in county Durham. Eggleston Hall Gardens, known as the ‘Secret Garden of the North’ is a well-liked tourist attraction in the village [19]. The last stop of the tour is Barnard Castle which is a market town is Teesdale, County Durham. The town is famous with its 12th century castle and has a market on Wednesdays, which is the reason why Route 572 is popular among locals. The Witham Arts Centre and Bowes Museum are among the other tourist attractions in Barnard Castle. The nearby Eggleston Abbey, a 13th century monastery of Premonstratensian can also be visited while visiting Barnard Castle [22]. On the way back, the bus goes from Barnard Castle to Brough via A66 by skipping Eggleston and Middleton in

Tessdale. Durig the summer, the bus passes through Selset reservoir on its way back to Ravenstondale, providing the passengers with a while worthy view.

2.4 Reviewing Existing Applications

This section will review four different mobile application. Once the review is complete, outcomes will be discussed.

2.4.1 Big Bus Tours

Big Bus Tours is the mobile application of the hop-on, hop-off style bus company that caters to tourists. The company has three different tours within London, covering the majority of the city's tourist attractions [6]. The mobile application is available for both Android and iOS. The application promises accurate real-time bus tracking, stop-by-stop overview of routes and push notifications to keep the passengers informed. Hence, functionality-wise, Big Bus Tours mobile app has several characteristics in common with the application that this project aims to develop. For iOS, Big Bus Tours' mobile app has 4.6 out of 5 ratings that was voted by 86 users as of 24.07.2018 [30]. However, the rating numbers goes down to 3.2 on Androids Play Store with 893 voters, giving a clearer representation of user satisfaction [38]. Some of the major complaints include the fact that routes and bus times are not always up to date and the application crushes often.

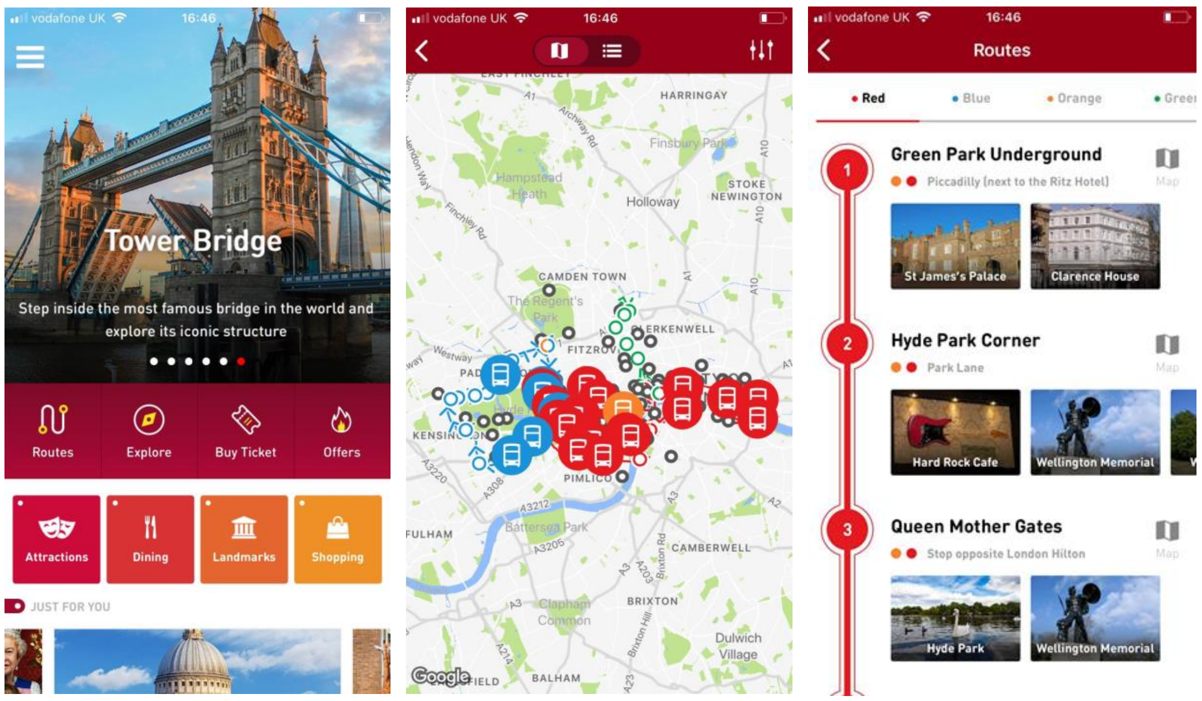


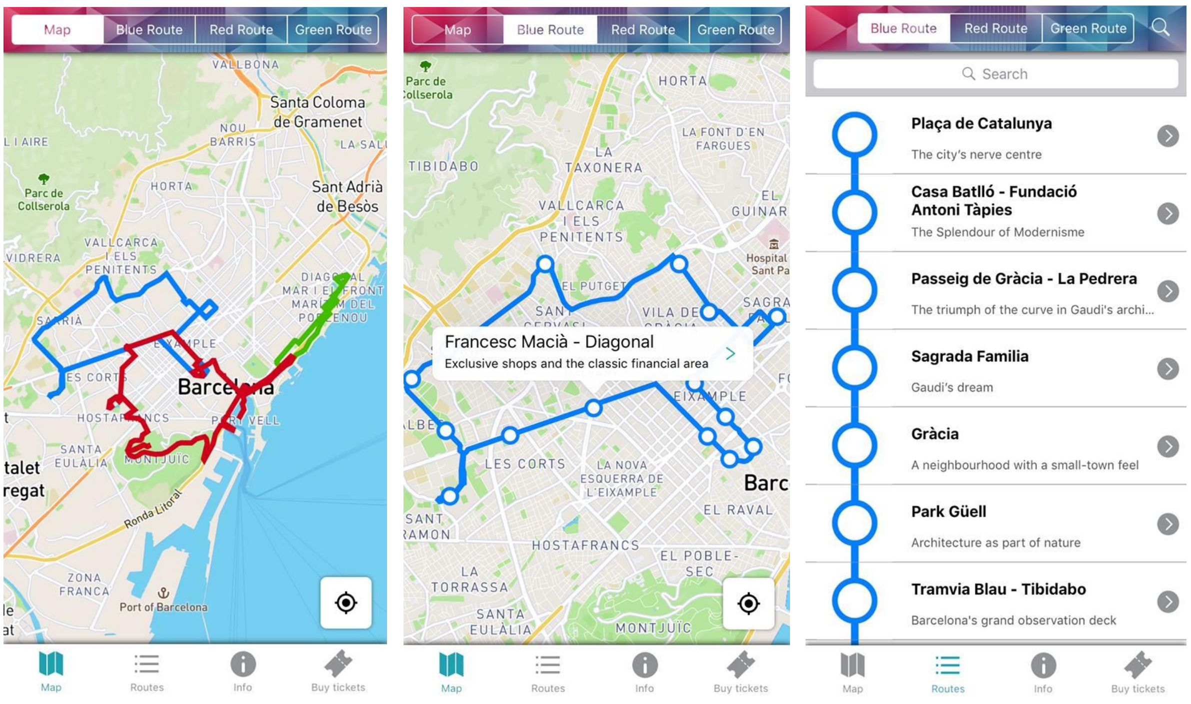
Figure 1: Snippets from the Big Bus Tours. From left to right: Home page, Explore and Routes.

Design wise, Big Bus Tours has a crowded home page that does not fit the screen of iPhone7, so a bit of scrolling is needed to see all the content. With sliding clickable images, options to view where to eat and shop, what to see and do, the home page tries hard to do more than what the actual bus company does. The application tries to behave as a full on tourist guide instead of guiding the bus passengers and giving them information regarding the bus hours, stops and routes. The "explore" button takes the user to the map which is a Google Map.

Again, the user faces a crowded map where stops and routes seem to be meshed up together. “Routes” option gives the user a clear idea about where each different route stops starting from the first stop until the last one. When the user clicks on a stop, surrounding points of interests show up. Overall, the application may not be ideal for novice users, it does not navigate the user and it is not open-source.

2.4.2 Barcelona Bus Turistic

Barcelona Bus Turistic is Barcelona’s official sightseeing bus. The company works in a similar fashion with Big Bus Tours where passengers can hop-on and off the buses wherever they want [4]. Its mobile application of the same name is available for both Android and iOS. As of 20.07.2018, the application has 2.4 stars voted by 30 users on Google Play store and no ratings on Apple’s App Store [29] [40]. One of the common complaints on Play store was regarding unhelpfulness of the application for users stated it was hard to detect bus stops or track the bus with the application [40].



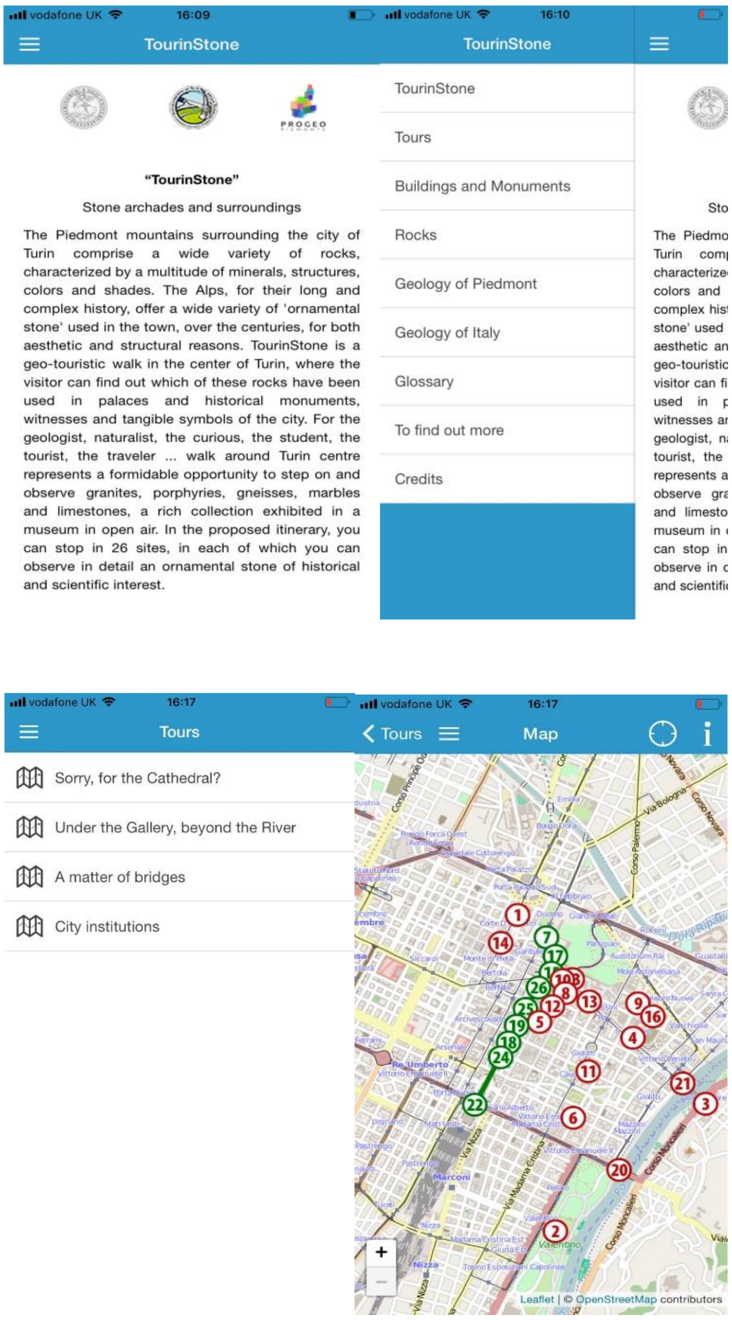
Plaça de Catalunya
The city's nerve centre
Casa Batlló - Fundació Antoni Tàpies
The Splendour of Modernisme
Passeig de Gràcia - La Pedrera
The triumph of the curve in Gaudi's archi...
Sagrada Familia
Gaudi's dream
Gràcia
A neighbourhood with a small-town feel
Park Güell
Architecture as part of nature
Tramvia Blau - Tibidabo
Barcelona's grand observation deck

Figure 2: Snippets from the Barcelona Bus Turistic. From left to right: Home page, Blue Route and Routes.

It was observed that Barcelona Bus Turistic has very clear and neat design. The application opens up to a map, it is a Google Map, and the user can see three routes that are well defined on the map with different colours. When clicked on the separate routes that are placed on top of the screen, the user gets a route specific map with each stop pointed out by a circle. These circles could be clicked on to get further information regarding the name of the stop and point of interests surrounding the stop. Aside from the map views, the application has “Routes” button that functions exactly the same as Big Bus Tours. There also are buttons on bottom of the screen for basic information about the tours and a button to purchase tickets. Overall, it was observed that this application has a better design that does not tire the eye.

2.4.3 TourinStones

TourinStones is an application that combines education, geology and tourism. The application was designed to take the user out for a walk in Turin to show and teach them about rocks and minerals that has been used to build the city up [41]. What makes TourinStones application different compared to the first two applications, other than the obvious fact that it does not involve buses, is that it was developed with open-source software such as Ionic Framework and PhoneGap. Hence, this is a hybrid cross-platform application that is available for both IOS and Android.



"TourinStone"

Stone archades and surroundings

The Piedmont mountains surrounding the city of Turin comprise a wide variety of rocks, characterized by a multitude of minerals, structures, colors and shades. The Alps, for their long and complex history, offer a wide variety of 'ornamental stone' used in the town, over the centuries, for both aesthetic and structural reasons. TourinStone is a geo-touristic walk in the center of Turin, where the visitor can find out which of these rocks have been used in palaces and historical monuments, witnesses and tangible symbols of the city. For the geologist, naturalist, the curious, the student, the tourist, the traveler ... walk around Turin centre represents a formidable opportunity to step on and observe granites, porphyries, gneisses, marbles and limestones, a rich collection exhibited in a museum in open air. In the proposed itinerary, you can stop in 26 sites, in each of which you can observe in detail an ornamental stone of historical and scientific interest.

Figure 3: Snippets from the TourinStone. From left to right: Home page, Main Menu , "Tours" and "Sorry,for the Cathedral?".

TourinStones opens up to an informative home page that is mostly plain text. Based on the developer's observations, the overall look of the application resembles more of a mobile responsive website than a mobile application. There is a menu bar on the top left corner which takes the user to further information on the project, rocks, geology of Italy and tours. When clicked on "Tours", user needs to pick one from the four options. When a decision is made, the user is taken to the map page which is a Leaflet map with OpenStreetMap tiles. However, there is a similar problem to the Big Bus Tours, which is that the map looks crowded. Regardless of which route or tour the user chooses, all the point of interests stays visible on the map, certain points change color to show the user which route they are following.

2.4.4 SASAbus

This is an application that is totally open source, so not just it only uses open-source software, its source code also is available on GitHub [45]. SASAbus is available as a native application on Android and available as a HTML5 web application on other platforms. It uses Leaflet map with OpenStreetMap tiles. SASA offers public transportation services in cities in South Tyrol (Italy). The project is open to contributors. SASAbus has 3.1 ratings on Google Play store that was rated by 520 users [39]. However there are no ratings nor reviews on Apple's App Store [26].

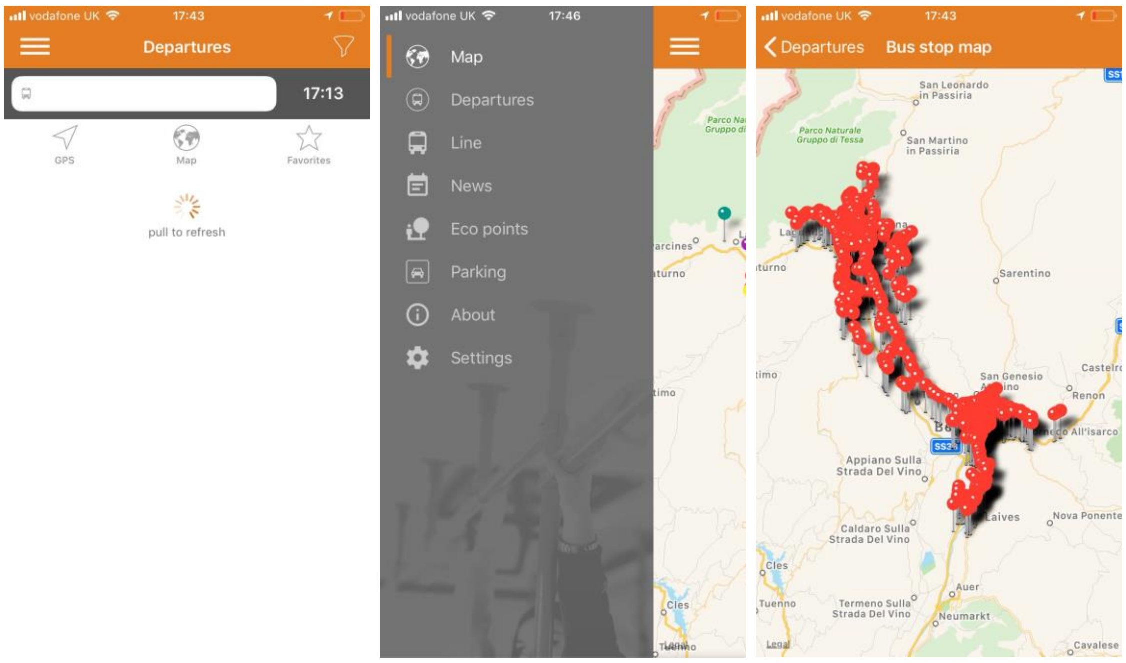


Figure 4: Snippets from the SASAbus. From left to right: Home page, Main Menu and "Bus stop map".

The application has a similar design to TourinStones with a menu button on the top left hand side of the home page and a very busy looking map view for they also decided to drop all the bus stops on the same map view. At the time of this analysis, the home page was not loading on iPhone7 due to an unknown reason.

2.4.5 Findings

Based on the review on existing mobile applications, it has been concluded that it is important to keep the application as simple as possible. Applications are meant to do one thing and do that one thing well. Hence, overcrowding the application is not a good idea [3]. Consequently, busy looking maps, overcrowded home pages are among the things that will be avoided when building the application for this project. Second design downfall that was seen in some of the applications are long menu listings [27]. It may be the best to avoid a menu if possible to give the application more of a mobile look rather than a web look. Overall, all the applications are available for both Android and iOS, which further proves the necessity of building a cross platform application. It was observed that applications that were built for community and by community such as SASAbus, and applications that focused on niche areas such as geo-tourism like TourinStone were both built with open-source software. It was also observed that both open-source oriented applications are using Leaflet map with OpenStreetMap tiles. Also, both commercial applications had a “Routes” section that just gave static information about the route that the bus fallows. Furthermore, based on the user reviews, it is suggested that application accuracy is of high importance to user satisfaction. Android users seem to prefer navigation related applications more so than iOS users because Google Play Store always had more user ratings and reviews compared to Apple’s App Store. This may as well be a natural result of the fact that Android users worldwide are higher in number than iOS users. These observations will be taken into account while building the application intended for this project.

2.5 Technologies

2.5.1 Open Source Software

Open source software and tools were chosen to be used in this project due to cost efficiency, online community support and the philosophy behind it. Consequently, the final version of the project will be open source itself. The finalized version of the code will be uploaded to a public GitHub repository for anyone to make use of it. It is important to define proprietary or closed software first in order to understand open source software. Proprietary software is a software where its source code is closed to the public. Such software is not free of cost and users of it need to abide by the license agreements. The license agreement of a proprietary software restricts the user from sharing, altering or copying the software [24]. Open source software on the other hand, as its name implies, has its source code open to the public. Meaning, the user can not only view the code but alter it, copy it, share it and incorporate it in their software [24]. Contrary to common misconception, an open source software is not always free of charge, although this sometimes can be the case. An open source software owner can also charge the user for services and support. Research shows that there is not necessarily a quality difference between open source and proprietary software [42]. It can be stated that an open source software is as correct, understandable, complete, conscience, portable, consistent, maintainable, testable, usable and efficient as proprietary software [42]. However, open-source software tends to offer the user a better sense of security, because when the source code is open to public, security flaws are detected faster compare to a proprietary software, where a limited number of people work on improvements [13]. Some users may be sensitive about their privacy as well, so using open-source can also mean that they know if the software they are using is collecting any data on them. Another positive aspect of open-source software is free support and community involvement [13]. There are many forums and online communities organized around open-source software and it is common that people generally are generous with their knowledge and time. Consequently, compare to paying for support for a proprietary software, people favor free help from the communities. There are also major differences in ethics and objectives when open-source and proprietary software are compared. The philosophy behind open-source software is highly altruistic. Contributors often donate their free time for a higher cause. The cause is subjective and may change depending on the individual. Open-source developers, contributors and community, in general, enjoy creating something useful for the community, making technology available for all and not just for those with the financial means, and sharing knowledge [5].

2.5.2 Hybrid Cross Platform Mobile Application

At this moment, there are three major mobile platforms in the market that are iOS, Android and Windows. All these operating systems have their own primary programming language. iOS uses Swift and/or Objective-C, Android uses Java and Windows are written in C# [8]. Hence, when an application is written for a specific platform by using the platform specific language, it is called a native application. In another words, an application that was written in C# for Windows cannot be compatible with iOS. As seen in figure 5, it can be concluded that an application that aims to serve the majority and be as inclusive as possible cannot be written for a single platform. Hence, for native programming this will mean knowing all the programming languages and platform specific features. This is not always cost and time effective [10].

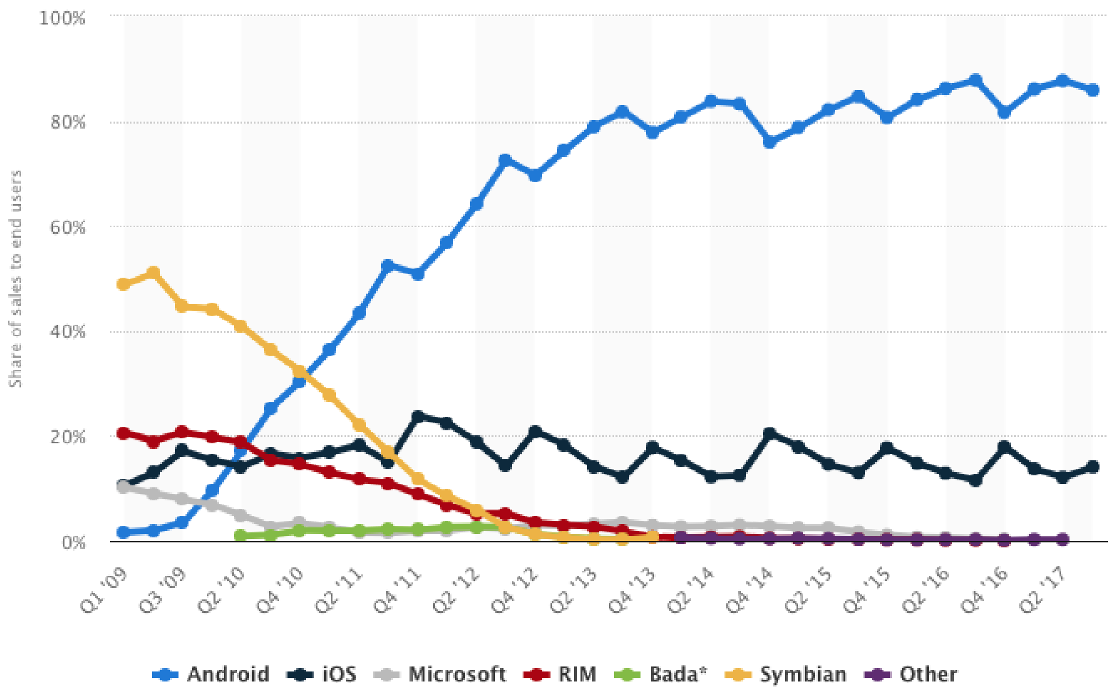


Figure 5: Global mobile OS market share in sales to end users from 1st quarter 2009 to 1st quarter 2018

Adopted From: https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/

Given the time constraints surrounding this project and the fact that both iOS and Android dominate the UK market as seen in figure 6, the application will be built as a hybrid cross-platform application. A hybrid cross-platform application is able to work on all platforms by using specific frameworks to give them a native feel. It is important to differentiate that a hybrid application is not the same as web application. Hybrid applications are not run by a browser. They run on a web container of the device so they have greater access to device specific features through application programming interfaces (API'S) [10]. Essentially, cross-platform development saves time since the code base is created once and compiled for each mobile platform [8]. There are various tools and languages available to build a hybrid cross platform mobile application. However, PhoneGap, HTML5, CSS3, JavaScript, jQueryMobile and Leaflet are the ones chosen to be used in this project.

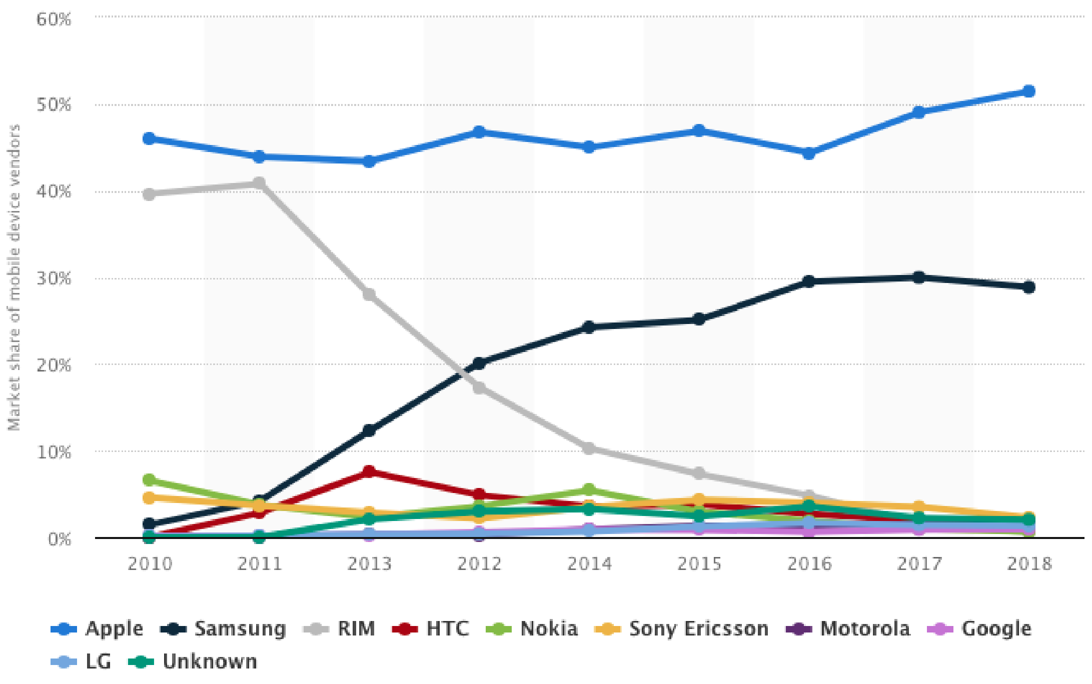


Figure 6: Market share of leading mobile device vendors in the United Kingdom (UK) from 2010 to 2018

Adopted From: https://www.statista.com/statistics/487780/market-share-of-mobile-device-vendors-uk/

➢ PhoneGap

PhoneGap was chosen due to its popularity and open-source framework. PhoneGap is a distribution of Apache Cordova. It was created by Nitobi but later acquired by Adobe in 2011 [37]. As a mobile application development framework, it lets developers build mobile applications by using HTML, CSS and JavaScript. This may be an advantage for front-end developers who are familiar with these languages. It also may be limiting for those who are more comfortable with lower level and object-oriented languages. The framework provides the user with a PhoneGap Command Line Terminal that can create, serve, compile, and run applications, a Desktop App as an alternative to command line terminal, Developer Mobile App which behaves as a mobile emulator, and Phone Gap build which is the compiler [37]. As an alternative to PhoneGap, Ionic framework was considered as it is also popular among the mobile developers. Ionic is open-source and works very much like PhoneGap by using HTML, CSS and JavaScript to create hybrid mobile applications. However, Ionic does not have a desktop app, so it makes terminal usage a must [28]. Due this this reason, Ionic was eliminated because PhoneGap's desktop app appears to be providing ease for novice developers.

- HTML5

  HTML stands for Hypertext Markup Language and is a markup language. HTML5 is an upgrade to the previous versions. This new version of HTML was published in October 2014 by the Wold Wide Web Consortium [59]. HTML5 brought many new functions and attributes with itself. However, for the purpose of this project, only the functionalities that are of interest to this project will be discussed. HTML5 was designed keeping low-powered devices in mind, so it makes a useful language for cross-platform mobile applications [59]. HTML5 offers some useful APIs that enables the developers to use geolocation, storing things locally and support for offline web applications [59]. It is expected that these new attributes will be useful for this project.

- CSS3

  CSS stands for Cascading Style Sheets. It is used as a complimentary to HTML for styling purposes. CSS3 is currently the newest version and it brought certain new functionalities with it such as rounded corners, shadows, gradients, transitions or animations [60]. CSS3 will be used in this project to make the application visually pleasing to users.

- JavaScript

  JavaScript is a scripting language and it will be used to add dynamic component to the application. JavaScript enables the programmer to declare variables, store and retrieve values, create cookies, define and invoke functions, create events and much more [61].

- JQueryMobile

  JQueryMobile is an open source user interface system that is based on HTML5. It was designed to make responsive web sites and apps that are accessible on all mobile devices and desktops [31]. JQueryMobile was built on jQuery and jQuery UI foundation. The system offers Ajax navigation with page transitions and touch events [31].

- Leaflet

  Leaflet is an open-source software that is ideal for creating interactive JavaScript maps for mobile applications. Leaflet is only 38KB which is expected to improve performance by decreasing load time [34]. It is widely used and recommended in the open-source community. Given its well documented library, it is ideal for beginner programmers as well as experienced ones. Leaflet provides its own plug-ins, methods, layers and controls which should be of use to this project. OpenLayer and Polymaps are alternatives to Leaflet and was considered to be used in this project prior to deciding on Leaflet. Both alternatives are open-source and free of cost just like Leaflet. However, Leaflet provided rich set of tutorials to get the developer started and had a very large, well organized library; something the other alternatives did not have. Leaflet was also used in some of the applications that was previously reviewed for this project. Consequently, Leaflet was favored over the alternatives.

2.6 Reflecting Back on Background Research

Throughout the research process, it was a challenge to find peer reviewed academic publications and books on subjects that are related to this project. There has not been much research done on open-source hybrid, cross-platform, mobile development nor there are statistics available telling what tools and technologies are favoured for open-source development. Consequently, no literature review was included in this chapter. The lack of reliable resources on the subject matter led the developer to make observations and rely on online articles.

3 BUILDING THE APPLICATION

Once the background research is done and enough theoretical knowledge is gathered on existing mobile applications, open-source software, hybrid cross-platform applications and tools and technologies to use, the software engineering process can begin. Hence, this chapter will explain the practical aspects of this project such as what software engineering is, software engineering model of choice and steps that were taken to develop the application.

3.1 Software Engineering

Software Engineering is a set of steps that are followed in order to produce a well-documented, maintainable software that meets the expectations. These set of steps may include requirement analysis, design, implementation, testing and sometimes retirement [32]. Software Engineering could be seen as a set of engineering principles that are applied to software development in order to produce a quality software that is delivered on time, within the allocated budget, and with the requirements expected by the customer [32]. Although there are a set of obvious steps to follow in order to produce a healthy software, there are various life cycle models available to choose from. Each model may have distinctive characteristics and approaches. Each model has their positive and negative aspects, and a model should be chosen based on the project and technical skills in hand, because while some models may make the developers life easier, a wrong model may push the project into failure. Two models were considered for this project: evolutionary prototyping and waterfall model. In evolutionary prototyping first prototype is built with zero to very few functionalities at the beginning. This first prototype forms the core of the software. More functionalities are added on top of or around this core. This model was considered as it gives the developer the flexibility to add new functionalities at any given stage. It is also a helpful model if there is a customer involvement, because the prototype can be presented to the customer for feedback after each iteration. This way, the developer is assured that the requirements are being met and necessary fixes can be done early in the life cycle. However, given the fact that a customer does not exist for this project and it is hard to establish a time frame when using the evolutionary prototyping, the waterfall model was chosen.

3.2 Waterfall Model

The waterfall model has a traditional approach towards software engineering and it is one of the oldest models. The model was developed in 1970 by Royce [15]. Waterfall is easy to understand and straight forward. It has a step-by-step implementation, where it reinforces define-before-design, design-before-code approach [32]. Consequently, as its name suggests, there is no iteration in this model. As implied on the figure 7, this model only goes forward; starting with the requirement analysis, fallowed up by design, implementation, verification and finally maintenance. Hence, waterfall model is not suitable for accommodating major changes later in the software life-cycle. Given its rigid linear nature, waterfall does not provide a prototype or a product itself until the end of the software life-cycle. It works well when requirements are known, unambiguous and stable [46]. Consequently, it can be especially risky for long projects where the requirements are not stable or may change. This model works best when used for smaller projects where the requirements are well defined, the team is experienced, and time is limited. However, this model can also be a fast-forward way to produce a software if the time is limited and requirements are well established. Despite its fallback, waterfall model was chosen for this project. The main reason for choosing waterfall is because this is a project that needs to be completed within a very limited amount of time and requirements are well established. Developer is not experienced in mobile development nor using tools and technologies such as PhoneGap and Leaflet, but some familiarity with web technologies such as CSS, HTML and JavaScript exists.

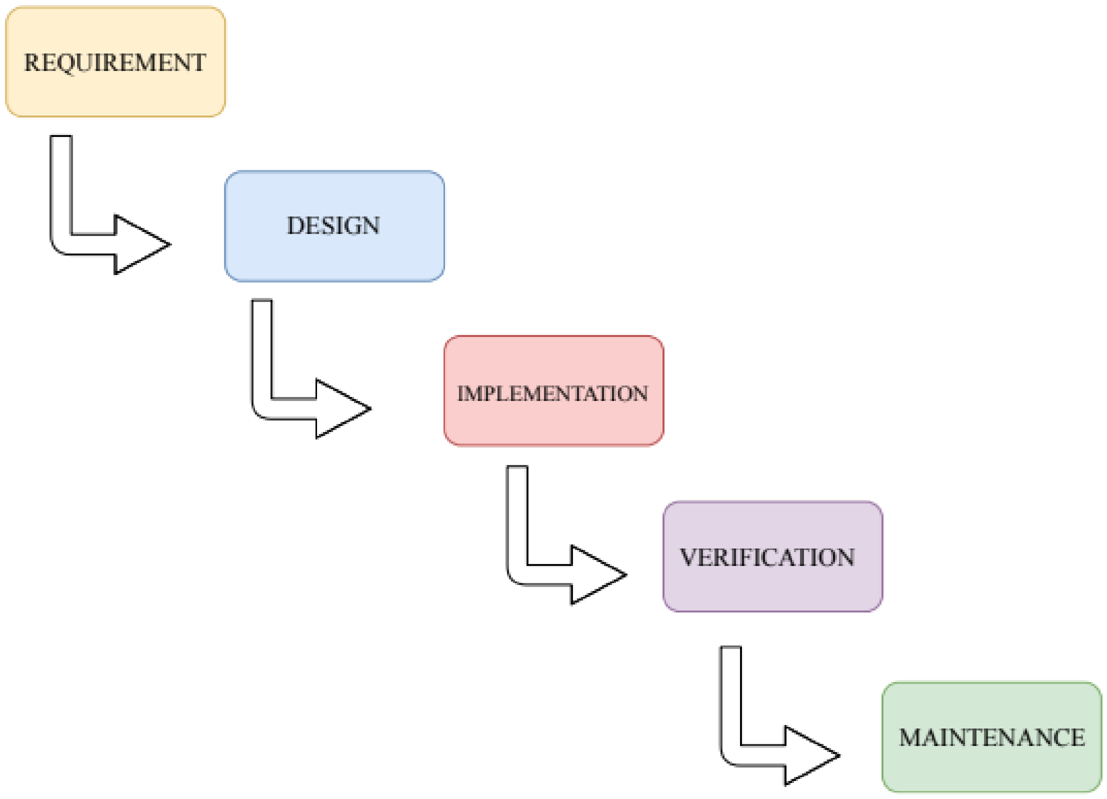


Figure 7: Waterfall Model.

3.3 Requirement Analysis

It is important that one knows what they are building before starting to code and in very simple terms this is why requirement analysis exists. In essence, requirements are a list of functionalities to implement in order to create the application software. It is “the scaffolding around which you will hang your design” [15]. It is crucial to understand the product requirements. If this first step of the software life cycle is done incorrectly, the entire finished product will be wrong especially when using the waterfall model. A good requirement analysis translates to fewer errors and clean design [15]. Requirements are typically divided into two as functional and non-functional requirements. Functional requirements are the features that the end user will see and be able to use. It answers what the program does. They are the external behaviour of the program. Non-functional requirements are concerned with the inner workings of the program such as defining any constraints, performance expectations, timing constraints, memory usage, file access privileges and security concerns [15]. Non-functional requirements answer how or how well the program does what it does. There are few techniques and activities to pin down the requirements of a new project that is to be built from scratch. Traditionally, an important step towards requirement analysis would be having a healthy communication with the customer to understand what it is that they want. However, in this project there is no real customer involvement, so no interview was constructed. Prototyping also is a common tool to use towards requirement analysis but this approach does not fit well with the waterfall model, so it was not used.

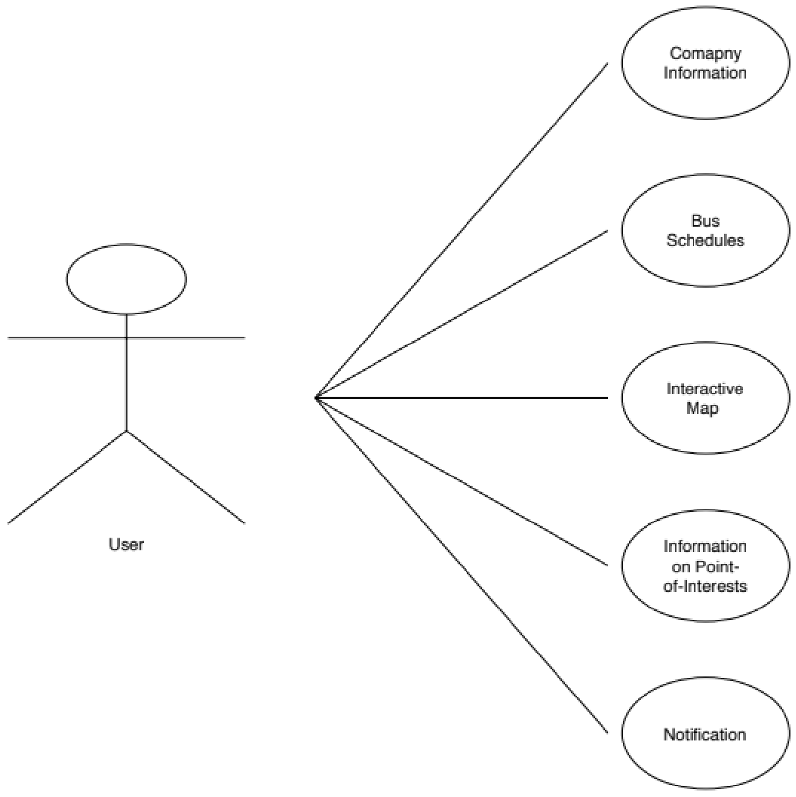


Figure 8: Use-Case Diagram.

The first step towards requirement analysis for this project was identifying the aim as discussed in the first chapter. The second step was understanding the possible needs of a future user by conducting a background research on existing mobile applications as discussed in the second chapter. Lastly, as seen in figure 8, a use-case diagram was produced to gain better understanding of interactions between future users and the system to be built.

	FUNCTIONAL REQUIREMENTS	IMPORTANCE
1	**Interactive Map** User will be presented with a map to interact with.	High
2	**Detect Location** The user will be able to see their current location on a map. Their location will be marked and this mark on the map will move as the user moves.	High
3	**Points of Interests** User will be able to click on the marked point of interests which will lead them to gather more information on that point of interest.	High
4	**Bus Schedule** Users will have access to bus arrival and departure times.	Moderate
5	**Location Access** User will give or deny access to their location	High
6	**Company Information** User will be provided with detailed information about Cumbria Classic Coaches	Low
7	**Notifications** User will receive notifications when they are getting closer to a point of interest	High
8	**Access to Information** User should still be able to see all the point of interests, bus schedules and company information if they deny access to their location	High

	NON-FUNCTIONAL REQUIREMENTS	IMPORTANCE
9	**Absence of Internet Connection** User should ideally be able to use the app as it is regardless of internet connection. If this cannot be achieved, user should at least be able to use all the basic functionalities of the map in the absence of internet connection. These basic functionalities are including all functionalities that the application provides, minus the location tracking and notifications.	Moderate
10	**Absence of Cell Signal** Same as stated above in Absence of Internet Connection.	Moderate
11	**Privacy** The system should respect user privacy by not sharing user location with third parties or storing it permanently.	High
12	**Graphical User Interface (GUI)** User will be presented with an easy to navigate GUI which they can interact with. GUI will consist of clear buttons, a map, and relevant text.	High
13	**Operating Systems** They system should be available for Android, iOS and Windows operating systems.	High

3.4 Design

Once the system requirements are established, the next step in the product life-cycle is design before going into the implementation phase. The design phase exists to provide a clear set of instructions for those have to implement it, test the implementation and maintain it later on. In other words, design is the blue print for the code. The approach taken for this phase was to take the requirements on hand, divide them into smaller components to have a better understanding of their behaviours and how they may interact with each other. This was done by using models. Models are simplified descriptions and visual representations of the system to be built based on the requirement analysis.

3.4.1 Architectural Design

Architectural design is a creative process, so the activities within the process varies depending on the type of system one aims to build [48]. Since, this project aims to develop a mobile application, appropriate models were chosen accordingly. Architectural pattern, as described by Sommerville is “a stylized, abstract description of good practice, which has been tried and tested in different systems and environments”. Model-View-Controller is a widely used pattern especially in web systems for its proven success. As seen in figure 9, Model-View-Controller was chosen for this project to have a high-level overview of the system before moving into more specific architectural designs.

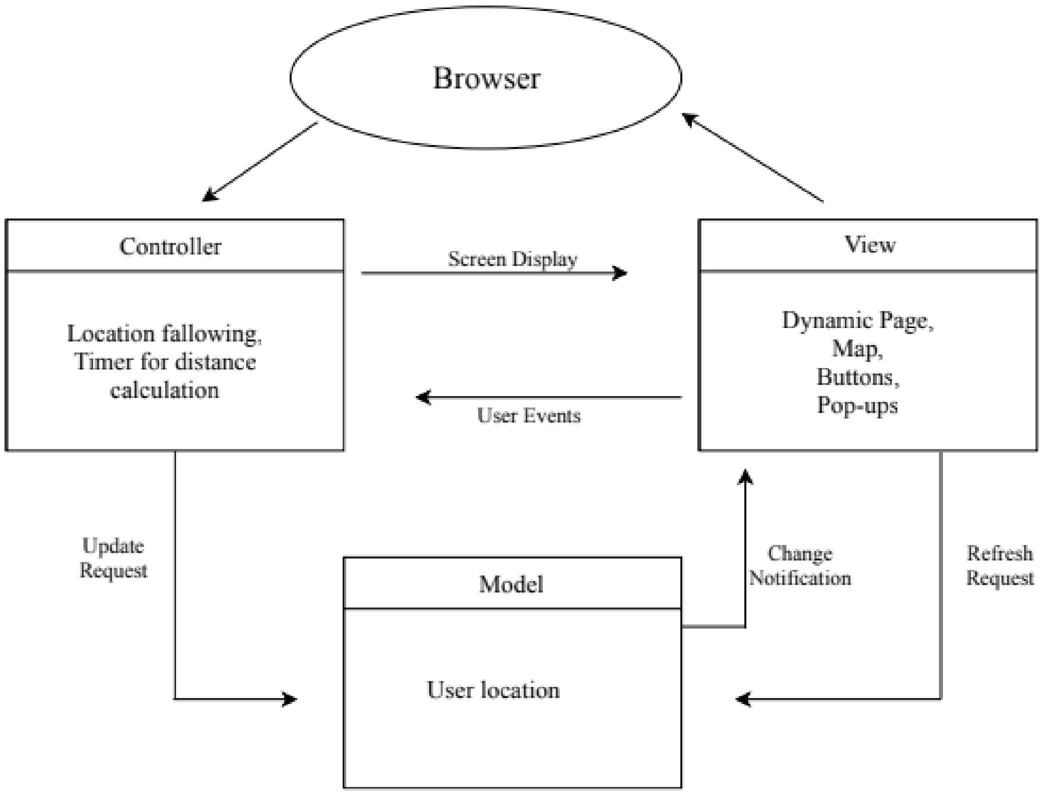


Figure 9: Model-View-Controller.

By using Model-View-Controller, related actions were logically grouped together. Low coupling was also achieved which will enable the system to be maintained and improved easily. For example, adding new buttons to the view will not effect the data in the model. By creating this pattern, it was also understood that there would not be a need for database for this application. User location will be stored in real time as long as the application is in use. Based on the location that is being held, the view is expected to change based on the pattern. There is also a direct correlation between distance calculation and the user location that is being stored.

3.4.2 Navigation Design

Once the system architecture was established, the second approach taken in the design phase was defining navigation pathways for the user. For this project, there is no user hierarchy hence a single navigation pathway will apply to all users. To establish these navigation pathways, navigation semantic unit (NSU) was used. By doing so, it became easier to see how the user will be navigating the application. As seen in figure 10, user is presented with three different pathways that all correlate to a page, and each page has access to each other. It was also outlined which page will consist of what information and event.

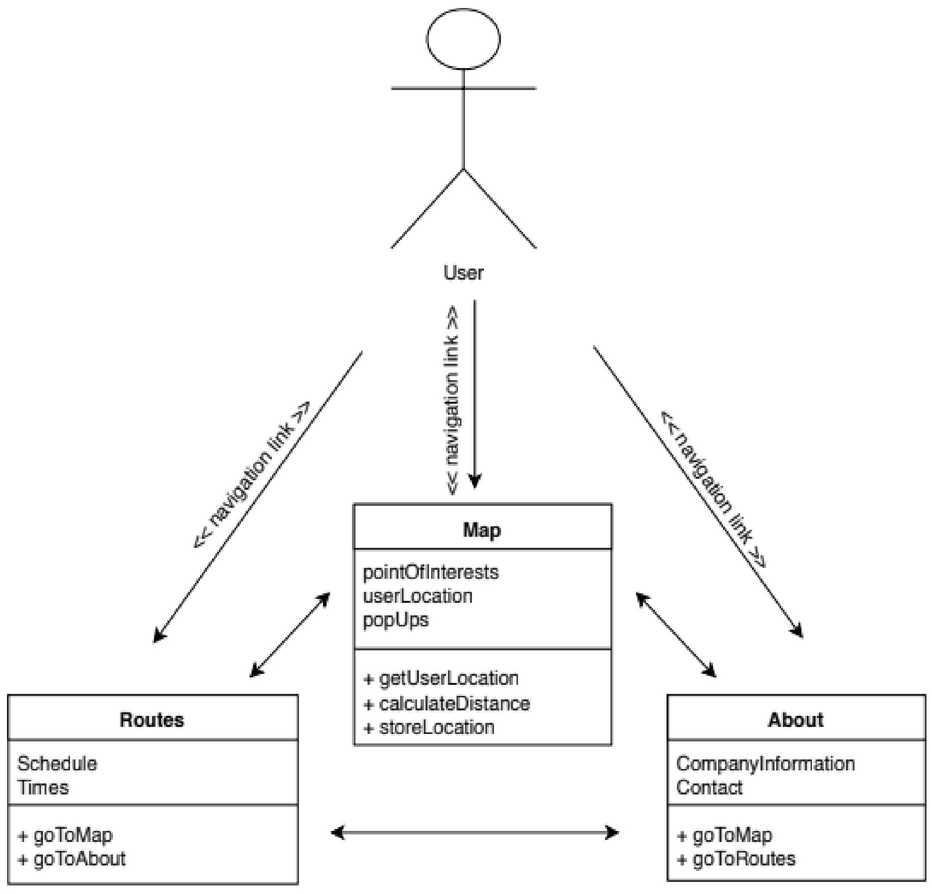


Figure 10: Navigation Semantic Unit.

3.4.3 User Interface Design

As this project aims to develop an application, it is important to consider user interface design as well. User interface is the way the user and a computer system interact through the graphical user interface or in more simple terms; through the screen of their phone. To have a successful user interface design, this project worked towards having an interface design that is easy to use, easy to learn, easy to navigate, intuitive, consistent, efficient, error-free, and functional [49]. To achieve these, Jakob Nielsen's heuristics principles were followed. Consequently, the user interface will follow platform conventions and will not re-invent the wheel. The buttons, images and overall look of the application will adopt familiar looks that has been used by many other applications [36]. This way, it is hoped that the user can feel a sense of familiarity towards the system and any potential learning time is either totally eliminated or decreased. The system will have clear exit or back buttons when necessary to give the user control and freedom over the system. Nielsen suggests error prevention is important and this can either be achieved by giving needed error messages to the user or totally eliminating the error-prone conditions all together [36]. Moreover, according to Nielsen’s design heuristics, it is important to stay away from information clutter [36]. Hence, the application will aim to have a simple and clean look that delivers the end-user only what they need or asked for.

3.4.4 Unified Modelling Language

Unified Modelling Language (UML) is a general-purpose modelling language that is widely used in the software engineering to help aid the design process. UML was used in this project to further aid the design process. Activity diagram which is a diagram in UML was used to capture and visualise the dynamic aspect of the system. Activity diagrams are often used to represent the activity flow of the system. For this project, activity diagram played an important part establishing how the system should work when user gives access to their location or denies it.

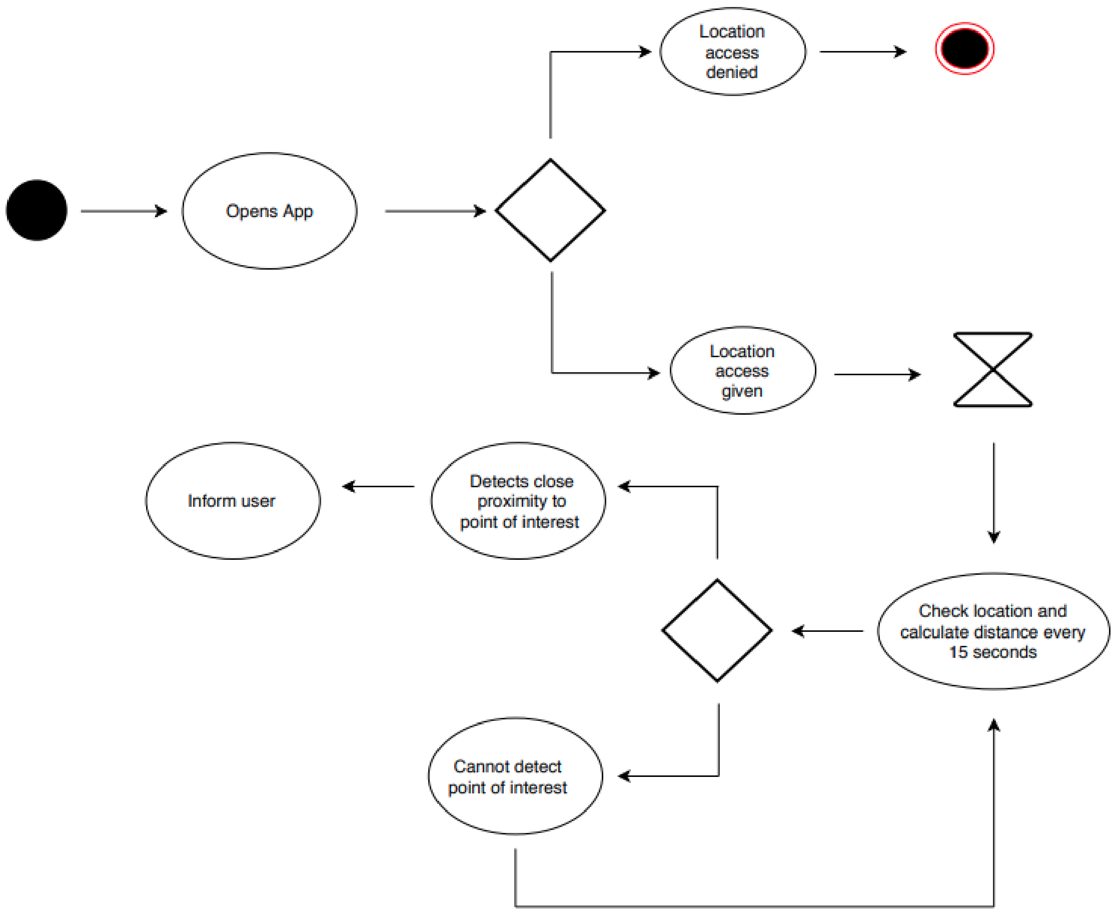

Location access denied
Opens App
Location access given
Inform user
Detects close proximity to point of interest
Check location and calculate distance every 15 seconds
Cannot detect point of interest

Figure 11: Activity Diagram.

As seen in figure 11, when the user denies giving access for their location, the system cannot calculate the distance between the user's location and any point of interest, hence the user will not be receiving any information when they come close to a point of interest. However, they should still be able to benefit from the system by using the map available and information it will provide. When user gives permission for the system to reach and store their location information, it is designed that the system will check the user's location and calculate the distance between all the points of interests to see if the user is coming closer to any of them. If the system detect that the user is getting closer, it informs the user regarding the approaching point of interest. If the system cannot detect any approaching point of interest, it continues to check. Another UML diagram that was used in the design process was sequence diagram. Sequence diagrams are used to capture the interactive behaviour of the system. They provide how the events of the system will take place in a sequential order. Hence, its purpose is not that different than the activity diagram. However, it was hoped that by having more design models and diagrams, any possible design error would be minimized. As seen in figure 12, sequence diagram had provided a deeper look into how the program should be storing the user location and use it to do distance calculations.

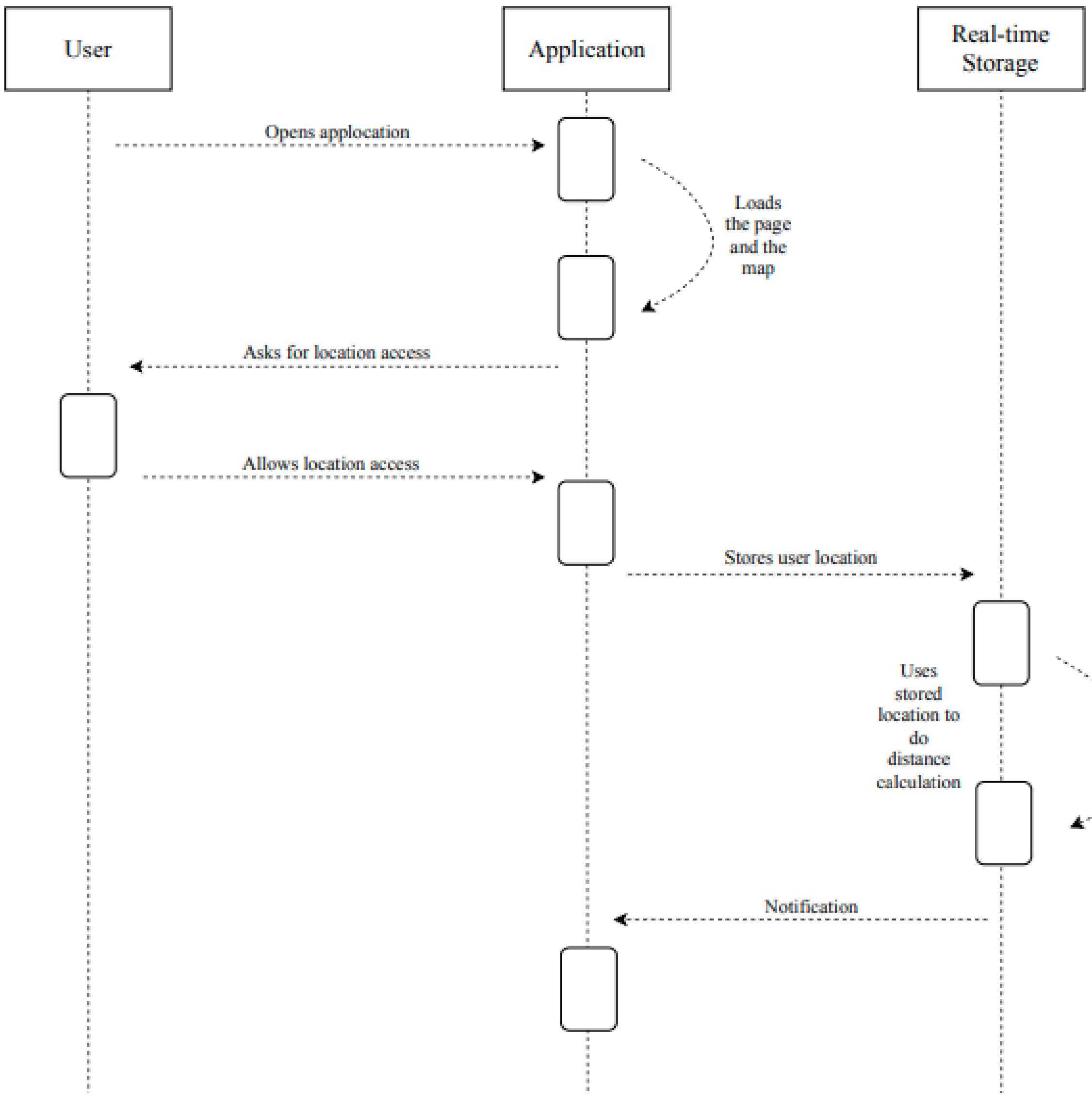


Figure 12: Sequence Diagram.

3.5 Implementation

Once the requirements are well established and design process is completed, one can move on to the implementation phase. Implementation phase is where the actual coding takes place and the product starts to turn into a reality from an idea. This section will discuss the implementation phase, provide high-level explanation of the code and touch on the benefits and limitations of the used tools. The visuals of the finished product can be seen below in figure 13 and figure 14.

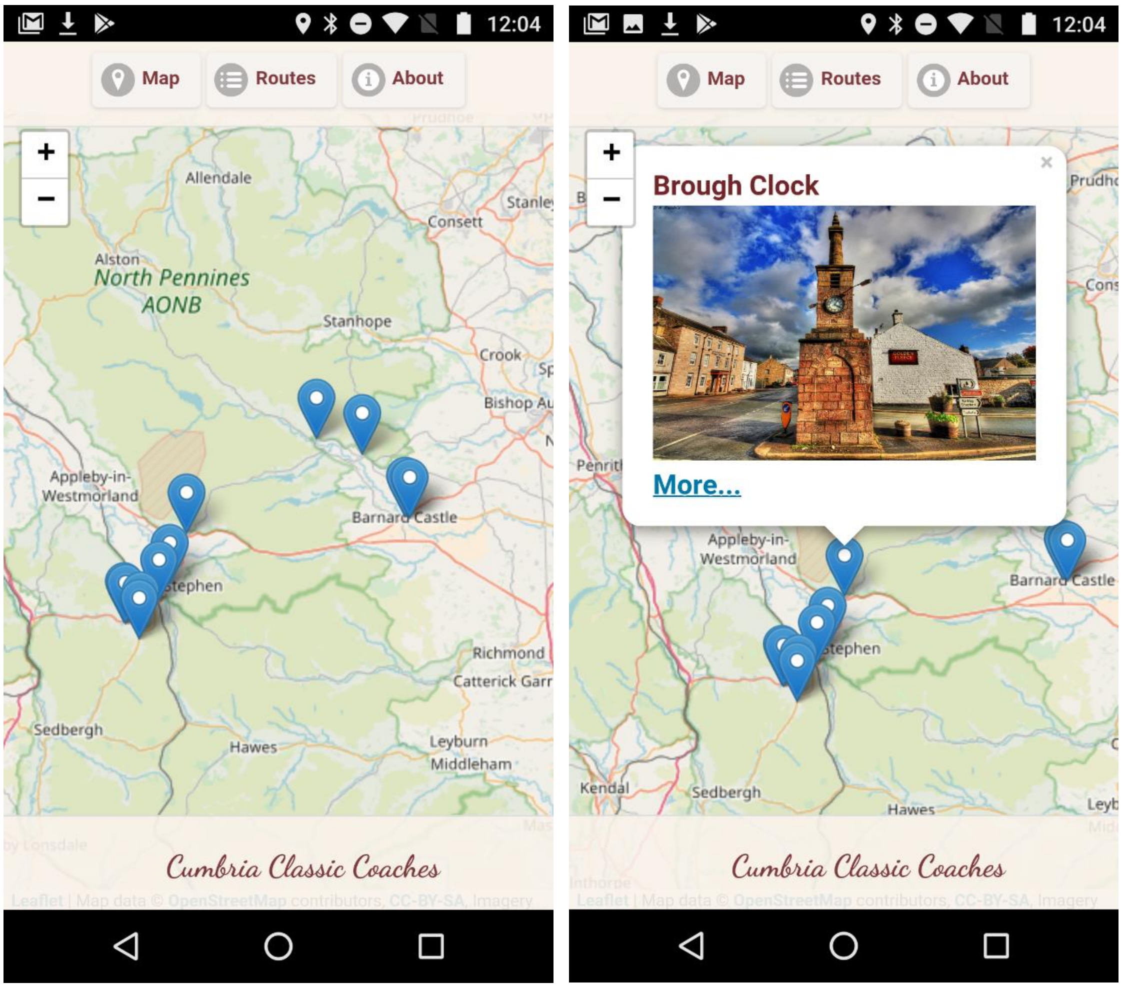
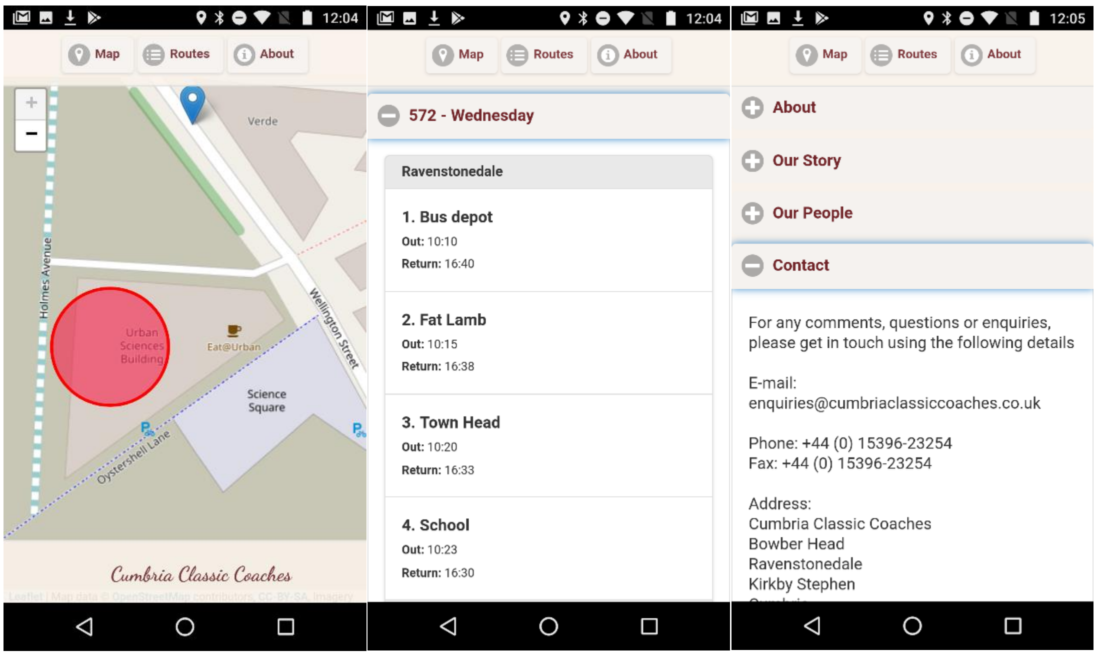


Figure 13: Screenshots of the end product. From left to right: Home page which is the “Map”, a marker pop-up.



572 - Wednesday

Ravenstonedale

1. Bus depot
Out: 10:10
Return: 16:40

2. Fat Lamb
Out: 10:15
Return: 16:38

3. Town Head
Out: 10:20
Return: 16:33

4. School
Out: 10:23
Return: 16:30

About

Our Story

Our People

Contact

For any comments, questions or enquiries, please get in touch using the following details

E-mail:
enquiries@cumbriaclassiccoaches.co.uk

Phone: +44 (0) 15396-23254
Fax: +44 (0) 15396-23254

Address:
Cumbria Classic Coaches
Bowber Head
Ravenstonedale
Kirkby Stephen

Figure 14: Screenshots of the end product. From left to right: location detection, “Routes” and “About”.

3.5.1 Phone Gap Desktop

The first step forward during the implementation stage was to download PhoneGap's desktop app via PhoneGap's official website. The desktop app is available for both Windows and Mac operating systems. PhoneGap uses the Cordova library and is completely open source. Once PhoneGap was downloaded, the project file was created via the PhoneGap's desktop app. A unique id is required at this stage. This id is needed to deploy the application because both App Store and Play Store requires unique ids to publish any application. The file system that is created by PhoneGap's desktop app contains a "www" folder that has an index.html file and folders for images, JavaScript and CSS. It also contains the config.xml file which will be discussed in detail later on this section, the plugin folder that contain all the PhoneGap plugins, and it contains hook and platform folders. These latter folders are used internally, so at the beginning of the coding process, the only thing that the developer needs to be concerned about is the "www" folder.

3.5.2 Atom

All the coding for this project was done on Atom, the text editor. Atom was developed by GitHub so it also is open source. Atom was chosen due to various reasons. This text-editor provides the user with smart autocomplete, find-and-replace, multi-pane views, and file system browser. Moreover, Atom has embedded Git controls and a lot of add-on packages that helped the developer write the code. Some of the packages that were used during coding are atom-beautify. This package makes the code more readable by automatically re-organizing the parts that are messy for HTML, CSS and JavaScript. Another handy package that eased up the coding process was JavaScript-Snippets. This package saved a lot of time by completing the common JavaScript syntaxes and reduced the chance of any syntax error.

3.5.3 HTML and jQuery Mobile

The first step forward building the app was writing the HTML code with jQuery Mobile. jQuery Mobile library was used to give the traditional html a more mobile look that users are familiar from native apps. To incorporate jQuery Mobile, the content delivery networks (CDN) as seen in the below code snippet was implemented in the head right after the meta tags.

```
<link rel="stylesheet"
href="http://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.css" />
<script src="http://code.jquery.com/jquery-1.11.1.min.js"></script>
<script src="http://code.jquery.com/mobile/1.4.5/jquery.mobile-
1.4.5.min.js"></script>
```

The CDN's first line represents the jQuery stylesheet, next line is for the core JavaScript library and the third line is the mobile library. Alongside with CDN, meta tag was set up as seen below:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0,
maximum-scale=1.0, user-scalable=no" />
```

It is important to set viewport meta tag as such to tell the browser how to display the page zoom levels and dimensions, otherwise many mobile browsers will use a “virtual” page width that is around 900 pixels and that is not an ideal for small mobile screens (jQuery Mobile). After jQuery Mobile was fully installed, it was used to create jQuery Mobile pages within a single HTML page. This was done by using HTML5 doctypes and jQuery’s library. A regular jQuery Mobile page structure comes in three sections that are: header, content and footer. An example of this could be seen in the below code snippet.

```
<div data-role="page">
	<div data-role="header">...</div>
	<div role="main" class="ui-content">...</div>
	<div data-role="footer">...</div>
</div>
```

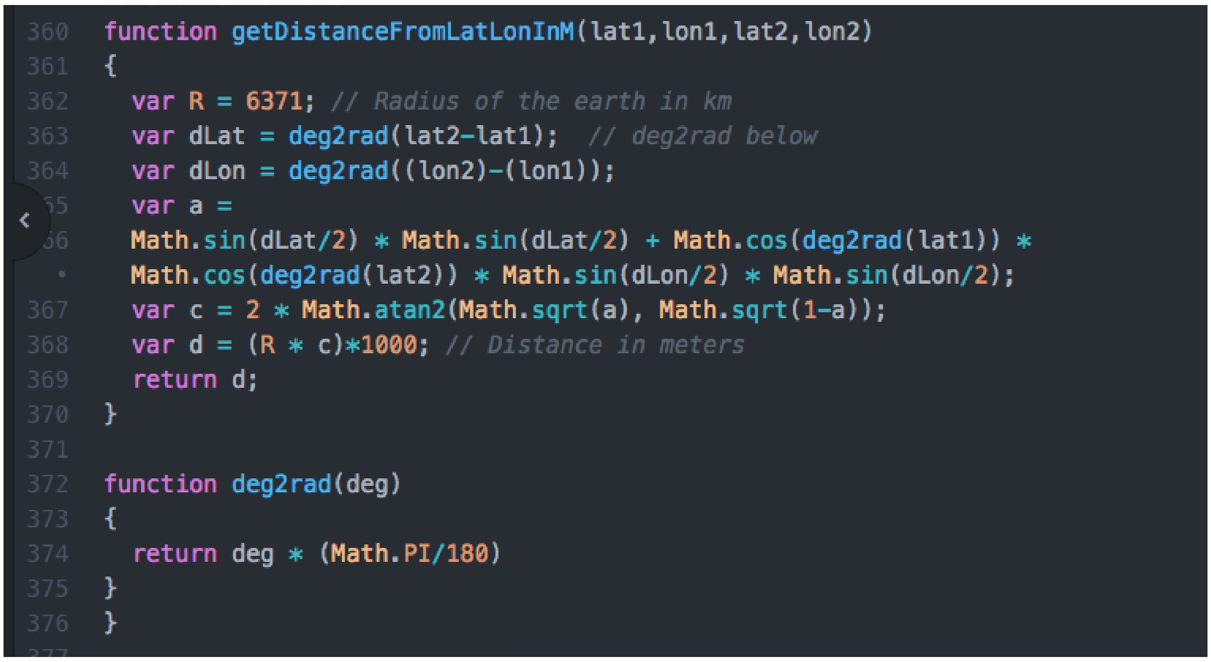
All three pages were created with the same style. jQuery Mobile’s help was also taken to create the header buttons and the collapsible buttons.

3.5.4 CSS

Once the majority of the HTML5 code was written, CSS was used to style the app. Matching the application color to the Classic Cumbria Coaches’ website’s color was important. Hence, a pick-color tool was used to detect the exact color code that was used to build the company’s website. jQuery Mobile is not as welcoming towards styling as plain HTML, so the developer was constrained on this and could not do as big of a change on the button shapes and colors as hoped for.

3.5.5 JavaScript and Leaflet Library

There are multiple functionalities that were achieved by combining JavaScript with Leaflet’s library. To set Leaflet up, the map’s CSS and JavaScript file was included in the head section of the HTML file. Once this was done, the map was initialized in the JavaScript file. The map was set to be displayed with Open Street Map tiles and the default map view contains and centers all the point of interests that Route 572 passes by. All the point of interests were pointed out by using the “marker” element from Leaflet’s library. When user gives permission access to their location, the location is circled in red by the “circle” element in Leaflet’s library. A crucial part of the program was to capture the users location in real-time and reflect this with the circle on the map, so the user knows where they are or what is their proximity to any of the markers on the map. This was achieved by storing users’ latitude and longitude information as a variable. Once the user location was detected, the watch attribute was set to true. This enabled the program to continuously watch and retrieve user location instead of detecting it once and not checking it again. However, just setting up the watch attribute is not enough. To reflect the renewing location on the map, the red circle that was associated with the user location is being redrawn with the “redraw()” method. This way, the user gets to see a red circle on the map that moves when they move. Another important functionality that was implemented with JavaScript is the Haversine’s formula which could be seen in figure 15. This formula calculates the distance between two points on a sphere given their longitudes and latitudes. It was used to calculate the distance between a point of interests and a user’s location.



```
function getDistanceFromLatLonInM(lat1,lon1,lat2,lon2)
{
  var R = 6371; // Radius of the earth in km
  var dLat = deg2rad(lat2-lat1);  // deg2rad below
  var dLon = deg2rad((lon2)-(lon1));
  var a =
  Math.sin(dLat/2) * Math.sin(dLat/2) + Math.cos(deg2rad(lat1)) *
  Math.cos(deg2rad(lat2)) * Math.sin(dLon/2) * Math.sin(dLon/2);
  var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
  var d = (R * c)*1000; // Distance in meters
  return d;
}

function deg2rad(deg)
{
  return deg * (Math.PI/180)
}
}
```

Figure 15: Code snippet from the original source code showing JavaScript implementation of the Haversine's formula.

Another helpful JavaScript method that was used in this project is "setInterval ( )". This function enables a certain part of the program to run in time intervals. This was needed for the project because without the intervals, the program checks the distance between the user and the point of interests only once when the application was first opened. In real life, the program should calculate and check these distances every so often as long as the application is open and running. By setting the time interval to five seconds, the code that does the distance calculation runs once every five seconds. This way, it was assured that the user will get alerts when they approach to a point of interest as long as their app is in use.

3.5.6 Debugging and Developer Tools

Since a user-centric program was being developed with web-based technologies, it was important for the developer to check how the code was reflecting itself visually. This was needed to see if the code was doing what it was precisely written to do so. Two major tools were used for debugging. First of these was Google's Chrome web browser. Under the developer tools of Chrome, one can pick to see the code in mobile view and even choose a specific device they like to debug on such as iPhone7 or Nexus. Moreover, if a particular device is not readily available, the developer can just create a custom device by adding the name and dimensions of the desired device to start debugging their code. Chrome also provides the developer with a console. By using "console.log()" method in JavaScript file, the developer was able to check the correctness of the methods that were created, to see if they were returning the right values. Moreover, adding stopping point to your code is also possible with Chrome's developer tools. The second tool that was used for debugging and to check periodically how the code was reflecting itself on a mobile device was PhoneGap's developer mobile app. When the project was created with PhoneGap's desktop app, a server address was provided. This server address helps to establish the connection between the written code and the PhoneGap's developer mobile app. This way, developer can see how the code behaves on an actual mobile device. PhoneGap's mobile app had proven itself useful during the implementation phase because given hardware and software differences, not everything that seems to be working on a browser, worked on mobile. Hence, the mobile app was used for lower level debugging. However, this useful tool was not always available to the developer which slowed down the debugging process greatly due to the fact that the PhoneGap Developer App has been removed from the iOS App Store. This issue is discussed in length in later sections.

3.5.7 Configuration File

PhoneGap applications are configured using a config.xml file and it is important for submitting the application to application stores. There are certain mandatory attributes that needs to be set up inside the widget element of the configuration file. First one of these attributes is id. This is the unique identifier of the application. Second attribute is the version attribute that sets the version number of your app. Also, the name of the application, description of the application, information about the author are among the attributes that are not mandatory but are good practice to include them. Application icons and splash screen are among the things that are set within the configuration file. For security, the developer can limit access to certain uniform resource locators (URL). Allowed platforms can be set up in the configuration file as well. Plug-in implementations were also done within the config.xml file. Two of the important plug-ins used were Cordova's whitelist plug-in which activates any link that is present within the code and geolocation plug-in, which as the name implies, enables the application to reach device location.

3.5.8 Failed Attempts

Certain attributes and functionalities of the application failed to be implemented due to lack of expertise and time limitations. Building an application that partially worked during signal loss or lack of internet connection were among the requirements. Implementing an offline map to the application was going to satisfy this requirement. An offline map originally wanted to be implemented by using Mobile Atlas Creator with a logic developed by David Rust-Smith [44]. As implemented by Rust-Smith, desired map tiles are chosen from the Mobile Atlas Creator and then these tiles are downloaded in a compressed file. This file with the map tiles then be saved in to the main project file to be later called with JavaScript in absence of internet connection. Mobile Atlas Creator did not have the original Open Street Map tiles, so an alternative tile was going to be used for this mission. The system was designed to call stored tiles when the internet connection was off and call Leaflet map when connection was on. The exact steps that were created by Rust-Smith were carefully followed. However, after several attempts, offline map implementation with Mobile Atlas Creator was aborted for the stored map tiles were not loading and reasons for it were unknown. Second attempt to implement an offline map was with a Leaflet plugin: TileLayer.Cordova. This plugin is maintained by Greg Allensworth [2]. It was designed specifically for Cordova/PhoneGap applications. TileLayer.Cordova adds tile caching onto local device storage, switching between offline and online mode. Unfortunately, the developer was not able to implement this solution due to lack of experience with plugin use. Since this was going to be a third-party plugin, it was not as straight forward as implementing Cordova/PhoneGap's own plugins. Given the time shortage, it was decided not to spend any more valuable time on creating an offline map and instead focusing on getting all the main functionalities work in ideal conditions, where internet connection and cell signal is present. Similar challenges were faced when implementing push notifications into the application.

Push notifications are only possible when using PhoneGap's command line interface. The command line interface is no different than PhoneGap's desktop app in essence. It helps the user create a PhoneGap project. The desktop app was chosen for this project given its simplicity and the developer's desire to save time by eliminating time needed to learn terminal commands for PhoneGap. However, this decision was made without realizing certain elements of the PhoneGap, such as push notifications, which were available only when command line interface was used. This problem was realized mid-way through the implementation phase. Attempts were made to see if the push notification implementation was possible without using the terminal, but progress was coming slow, so the developer decided to drop push-notifications. Instead, in-application pop-ups and vibration was implemented to inform the user on the upcoming point or interests.

3.5.9 PhoneGap Build

PhoneGap Build was used in order to combine and convert all these web based technologies into mobile application. To do this, the code written was uploaded into a public GitHub repository, later this repository link was called from the free PhoneGap build account. PhoneGap Build offers paid accounts that come with private repository use, but this was not necessary, for this project is planned to be an open source project as well. Once PhoneGap receives the files via GitHub repository, it uses webview inside a native app to convert the files into a mobile application.

3.6 Verification and Testing

3.6.1 Verification

Verification was used to evaluate whether the application produced satisfies the conditions imposed by the requirement specifications [47]. Hence, the aim of verification is to make sure the program that was produced is the right one and that no requirement of high importance was bypassed during the implementation phase. Verification is a static practice, meaning the program is tested without executing the code [47]. This is a human based method, so no tools were used for verification. The testing model that is used for verification is called static testing. The first step towards verification was reviewing the code with a checklist to compare the end product to the requirement specifications. A technical review was also conducted to check if the code was written in accordance to technical specifications and standards. Validation techniques are needed to make sure a quality software was produced. Although verification is important, it is not enough by itself. Tests should be performed to make sure the software works as intended.

3.6.2 Testing

Testing is perhaps the most stressful part of building a program with the waterfall model. If a big flaw in the program is detected, fixing it will be both costly and time consuming for the developer would already be at the end of the software engineering life-cycle. Various tests were performed for this project including unit testing and manual testing.

3.6.2.1 Unit Testing

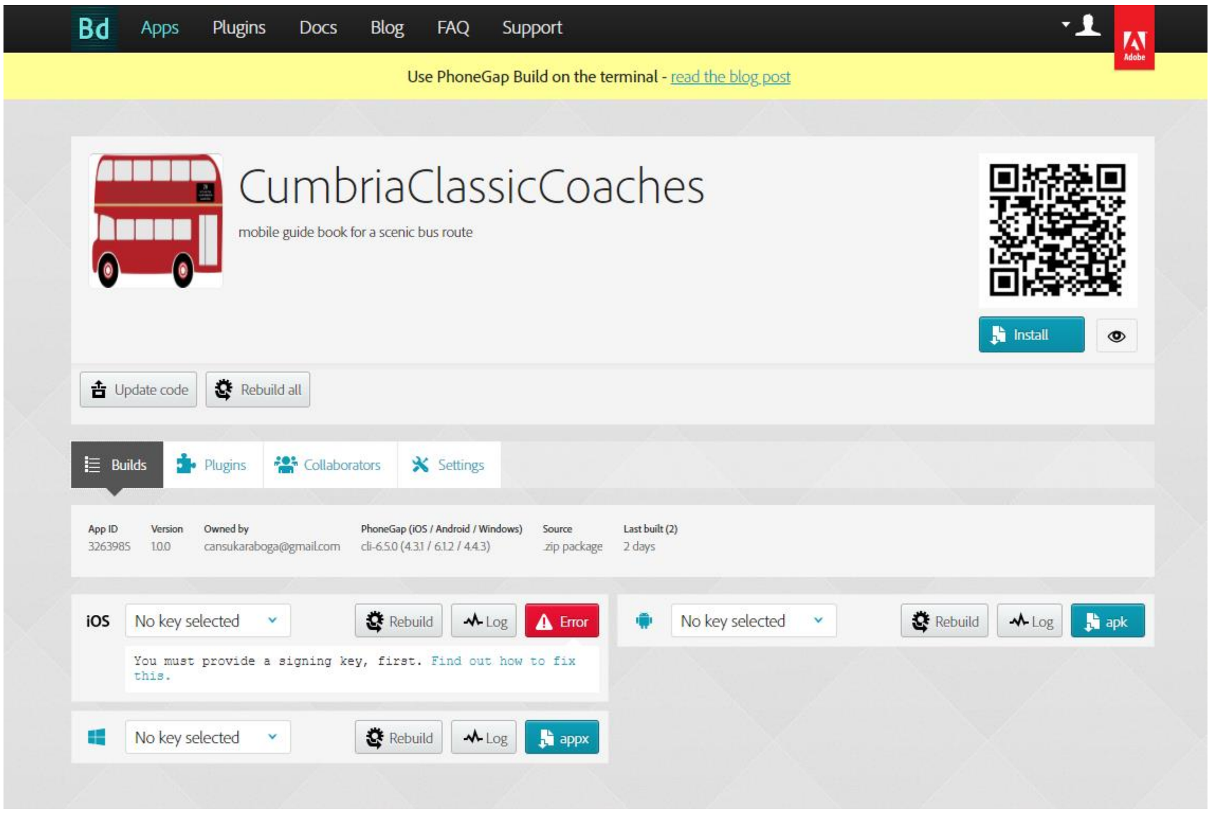
In unit testing, individual components of the software are tested. These are usually small components such as a method or a library. The purpose of unit testing is to make sure the small units of the code works as intended in hopes that everything combined will work well as a whole [21]. Unit testing was performed in parts of the code where the developer thought was critical. One of such parts was the method that was calculating the distance in meters using Haversine's formula. To test this method, the developer created special test cases by giving the method latitudes and longitudes of familiar locations, where the distance between the locations were already known and the unit tests were expected to return the right value.

3.6.2.2 Manual Testing

Manual testing is done without using any automation tools, where the tester plays the role of the end user to confirm that all the features of the application are working as planned [21]. Manual testing for this project was carried out multiple times in various scenarios with different configurations to assure correctness of the test results. Below table gives a brief overview of the tests completed before going into more details.

#	TESTS
1	**Desktop** Used Google Chrome's developer tools. Was carried out prior to file compilation
2	**PhoneGap's Developer App** Was carried out prior to file compilation
3	**Visiting Test Locations in Newcastle** Was carried out after the files were compiled into a .apk file
4	**Testing on the Route 572** Was carried out after the files were compiled into a .apk file

The first manual test was done on a browser before the code was assembled into a mobile application. Developer added some test locations to the code. These test locations were in close proximity to developer's work location which is Newcastle-upon-Tyne. The distance range given was minimum 50 meters and maximum 500 meters. Hence, if one of these test locations fell within the range of the developer's work location, an alert was expected to pop up on the browser, informing the developer that a test location is within the range. Second manual test was also performed before the files were compiled together into a hybrid app. The second test involved using the PhoneGap's developer mobile application. For this, a Samsung s7 with Android Oreo operating system was used. The overall functionality of the application was observed including static elements of the app, location detection, map markers, real-time location watching and alerts. Any difference detected between the ways the application functioned on desktop browser versus PhoneGap's developer mobile application was observed and noted down to debug. Later tests were done once the files were assembled together with PhoneGap build. However, these tests were only able to be carried out for Android. All the major operating systems such as Android, iOS and Windows require special keys as well as membership to a paid developer account in order to sign an application with the key and upload it to the related application store. However Android and Windows provide ease of testing by letting the developers create a debug.apk for Android and debug.appx for Windows machines. This way, one can test an application on an actual mobile phone even in absence of signing keys and memberships. Unfortunately, the latter is not the case with iOS. Apple does not let developers create debug.ipa files to be tested on an actual iPhone without a signing key which could only be attained once a paid membership is attained. As seen in figure 16, iOS shows an error, which means the file is not available for download via PhoneGap build. Consequently, further tests were not able to be carried out for iOS and Windows given the developer did not have an Apple developer account and did not have access to a Windows phone.



App ID	Version	Owned by	PhoneGap (iOS / Android / Windows)	Source	Last built (2)
3263985	1.0.0	cansukaraboga@gmail.com	cli-6.5.0 (4.3.1 / 6.1.2 / 4.4.3)	.zip package	2 days

```
You must provide a signing key, first. Find out how to fix
this.
```

Figure 16: PhoneGap build account, showing iOS error

Hence, the third test was done by downloading the application to MotoG5, which runs on Android. In this scenario, the developer walked passed all the test locations within the city to see if alerts were received for every each one of them. Location detection and live tracking were also observed to see how well the application functioned when the user is moving on foot. The fourth and final test was done by a volunteer at Cumbria Classic Coaches to see if the application was going to work as it worked for the test locations in Newcastle. However, no result was obtained from this final test. Hence, it is not known how the application worked and behaved when tested on a moving bus.

3.7 Maintenance

Maintenance includes all the modifications and updates that are done after the delivery of the software. It is the last phase of the software engineering life-cycle. It also is the longest phase, which lasts until the software is too old to be maintained. Life expectancy of a software depends on how fast the technology will evolve and how maintainable the original source code was. This is why it is important to write clean code with good commenting. It can extend the life of the software. There are various reasons why a software may need maintenance: Client may ask for changes or additional functionalities, hardware changes may lead to mandatory software changes, software may need to adapt to new technologies and problems may occur after deployment which will require fixing. For open-source software, the responsibility to maintain the code base rests mostly upon the volunteer contributors. A software should be retired if the cost of maintenance is becoming higher than the cost of a new software. For this specific project, there will not be any efforts towards maintenance.

4 RESULTS and EVALUATION

This chapter will discuss the results obtained from testing and elaborate on the failed requirements. Later, the chapter will evaluate how the entire software engineering process went and reflect back on various challenges that were faced and mistakes that were made during this process. The chapter will close with ideas on future improvements on the existing software.

4.1 Results Obtained

The first two set of tests that were done before compiling the files were mostly successful. The developer did receive alerts for the closest test location during the first test, which was the desired outcome. There were few problems when the files were tested on PhoneGap's developer app. It appeared that the footer on the home page was disappearing and JavaScript alerts were being received too soon, before the page was fully loaded. Both of these problems were solved with small alterations. The third test was successful as well. The developer was able to see her location on the map in the form of a red circle and this red circle moved as the developer moved. There was no latency. The in-app pop-up and vibration were activated each time the developer came nearby a test location. The developer did not receive a pop-up for a location that was already visited before as intended. Overall, majority of the requirements were satisfied. Below table gives a brief overview of the test results before going into more detail.

	FUNCTIONAL REQUIREMENTS	IMPORTANCE	TEST RESULTS
1	**Interactive Map** User will be presented with a map to interact with.	High	Successful
2	**Detect Location** The user will be able to see their current location on a map. Their location will be marked and this mark on the map will move as the user moves.	High	Successful
3	**Points of Interests** User will be able to click on the marked point of interests which will lead them to gather more information on that point of interest.	High	Successful
4	**Bus Schedule**	Moderate	Successful

	Users will have access to bus arrival and departure times.		
5	**Location Access** User will give or deny access to their location	High	Successful
6	**Company Information** User will be provided with detailed information about Cumbria Classic Coaches	Low	Successful
7	**Notifications** User will receive notifications when they are getting closer to a point of interest	High	Partially Successful
8	**Access to Information** User should still be able to see all the point of interests, bus schedules and company information if they deny access to their location	High	Unsuccessful

	NON-FUNCTIONAL REQUIREMENTS	IMPORTANCE	TEST RESULTS
9	**Absence of Internet Connection** User should ideally be able to use the app as it is regardless of internet connection. If this cannot be achieved, user should at least be able to use all the basic functionalities of the map in the absence of internet connection. These basic functionalities are including all functionalities that the application provides, minus the location tracking and notifications.	Moderate	Unsuccessful
10	**Absence of Cell Signal** Same as stated above in Absence of Internet Connection.	Moderate	Unsuccessful
11	**Privacy** The system should respect user privacy by not sharing user location with third parties or storing it permanently.	High	Successful
12	**Graphical User Interface (GUI)** User will be presented with an easy to navigate GUI which they can interact with. GUI will consist of clear buttons, a map, and relevant text.	High	Successful
13	**Operating Systems**	High	Partially Successful

	They system should be available for Android, iOS and Windows operating systems.		

Requirements seven and thirteen were marked partially successful. Notifications were marked partially successful due to the fact that the system notifies the user by in-app pop-ups and vibration and not actual push notifications. As discussed in the previous chapter, it was known from the implementation phase that this requirement was not going to be satisfied as well as originally hoped for. Push-notifications when combined with in-app pop-ups and vibration would have made the application more professional. However, the app still does inform the user, so requirement seven was partially satisfied. Operating Systems were marked partially successful due to the fact that the later tests were only done on Android. Hence, it is proved that the application works on Android. However, no proof exist for iOS or Windows. On the other hand, this application was developed with PhoneGap and web technologies. This means there is no chance that the code written will not be compatible with iOS or Windows. At most, some alterations will be needed for the config.xml file, which should not take a long time. For more niche operating systems, the code written can be deployed as a web app since such operating systems usually do not have an application store of their own. Requirement nine and ten had been unsuccessful prior to testing phase as explained in the previous chapter. Their importance level were moderate, but achieving these could have improved the overall quality of the application greatly. Requirement eight had also been unsuccessful which came as a surprise since during the first and second tests, it was successful. The system did ask the users permission to reach their location when it was tested on desktop and PhoneGap's developer app. However this functionality was lost when the files were compiled. During the third test, the user was asked if they wanted to download the application as it needed access to their location. If the user said no, they simply could not download the app. Hence, the end user was not given any option to use the system without giving access to their location.

4.2 Is Open-Source Beginner Friendly?

During the development process of this application, a lot of problems had risen that were believed to be open-source software related which led to the question: Is open-source software beginner friendly? One of the challenges experienced was the insufficient documentation by Leaflet and PhoneGap. While Leaflet provides enough material to get one started with a basic map, resources become scarce and undetailed when one needs to implement a map with advance functionalities. One such example is the lack of code examples regarding Leaflet methods. Originally distanceTo() method from the Leaflet library was intended to be used instead of the Haversine's formula. However, despite many attempts, the developer could not manage to call the method. Hence, more examples on how to implement Leaflet methods would have saved both time and lines of code. This problem that was faced with Leaflet libraries may also be due to the developer's lack of experience in building complex systems with JavaScript. However, throughout the coding process, writing JavaScript code by itself was not challenging for the developer. Contrarily, when Leaflet did not seem to be working, problems were attempted to be solved with pure JavaScript. Consequently, it is assumed that Leaflet's lack of detailed explanation on how to use the libraries must have been the problem. PhoneGap proved itself to be even more problematic than Leaflet. The biggest problem that was caused by PhoneGap was unavailability of PhoneGap Developer mobile app for iOS. Originally the app was available for Android, Windows and iOS. However, due to changes in Apple's own strict measures around the guidelines, the PhoneGap Developer app was removed from the App Store. Aside from the problems with iOS, the app was not available for all Android phones either, for there were compatibility issues with some of the hardware. The developer had scarce access to the PhoneGap Developer app given the available mobile phones for the project were an iPhone 7 that runs on iOS and a MotoG5, which's hardware was not compatible with the app. This was unfortunate because the PhoneGap Developer app could have make debugging faster and easier. The exact same piece of code that worked on a web browser did not always work on mobile. An ideal debugging process would have involved checking all the functionalities first with a browser on a laptop and once everything was working on the browser, checking the code once again with the app. Even though PhoneGap stated that they were working on the problem on a blog post on 25 June 2018, no progress had been made during the time frame of this project [7]. On the same blog post they offered a temporary solution to the problem which was to fork the open-source code of the PhoneGap Developer app from GitHub and build it using PhoneGap Build. This perhaps is a logical solution but only for people with paid Adobe accounts. As a low budget project, the free PhoneGap Build account that was used for this project did not let the developer compile projects that exceeds 50MB which PhoneGap Developer app does. Overall, having scarce access to PhoneGap Developer app surely hindered the overall project flow. Documentation of PhoneGap was also problematic especially for the config.xml file. The guide provided on the official website of PhoneGap build regarding config.xml file was misleading. A basic template was given with the essential properties such as i.d., version, description, author and platform needed in the config.xml file to get the app working [14]. However, this example was missing the important information regarding plug-in use depending on the functionalities included in the application. For example, certain plug-ins were needed to be included in the config.xml file if the app used geolocation or had any links. A lot of valuable time was lost due to this because when compiling was done without these plug-ins, the application was coming out faulty and it was hard to pin down what was causing this problem given the developer was assuming all the necessary steps were followed correctly. Moreover, finding help in general was challenging. Online open-source communities were not as helpful as expected them to be. It was not always possible to get an answer for a question posted. It was also observed that a lot more people preferred developing on native platforms and using Google Maps, so finding someone who had experience with PhoneGap and Leaflet was rare. There were also more tutorials and videos available on native platforms and usage of Google Maps on platforms such as Youtube and Udemy. As a result, a lot of time was spent on learning and comprehending instead of building the system.

4.3 Waterfall versus Evolutionary Prototyping

Waterfall model was used to build this application. However, desired results were not obtained. The mobile application did not meet some of the requirements and stayed well below expectations. This was partly due to mistakes that had been done during requirement analysis and design phases, but also due to the problems caused by Leaflet and PhoneGap as mentioned previously. To fix some of these mistakes, the developer went back to the source code to add additional code or correct existing ones. There was a period of rapid iterations between the design, implementation and testing phases after the waterfall model was completed. Such a practice surely does not have a place within the waterfall model. It is thought that evolutionary prototyping should have been picked instead of waterfall model for this project as originally considered. Even though there was no customer, the developer should have played the role of the customer by testing and evaluating the application after each prototype. This way, the problems that were faced with config.xml file and offline map implementation could have been realized early on and could have been fixed while there was sufficient time. Following the waterfall model gave the developer a false sense of assurance. It was thought that everything was going well until problems started manifesting themselves during the implementation phase and later during testing phase. When such problems were realized, it was already midway through or towards the end of the time available to finish this project. Consequently, evolutionary prototype could have given the developer a more realistic sense of the progress that was being made. This way, time on hand could have perhaps been used better to create more complete software that successfully contains all the functionalities.

4.4 Future Improvements

Although the current application works, it is not good enough to be deployed to any application store for public use. Improvements should be made in order to increase the quality of this application. The very first step forward to improve the quality of this software would be testing it on iOS and Windows to see if everything works as it works on Android. If not so, necessary alterations should be made on the config.xml file. Once it is known that the software works on all three operating systems, the next big improvement would be implementing actual push notifications and eliminating in-app alerts and vibrations. This will give the application a more professional look while giving the user the freedom to not focus on the application all the time but still get notified when they are approaching a point of interest. Another important improvement would be implementing an offline map. This way the user can still benefit from the map even in the absence of internet connection or signal loss. Moreover, the application should be improved to give the user the option to allow or deny the application's access to their location. This could possibly be done by adding a navigation button to the map. This way, the user will be tracked and notified only when they tap on the navigation button. Since the application was aimed to work in the absence of cell signal and internet connection, the information provided on the point of interests could be made as static as possible. At the moment the application directs the user to related links if they desire to learn more about an approaching point of interest. Providing carefully summarized information on each point of interest in HTML format will give the user undivided access to such information even in the absence of internet connection.

5 CONCLUSION

A hybrid, cross-platform and open-source mobile application was built for this project. The application was based on Cumbria Classic Coaches' route 572. The goal was to build an application for bus passengers travelling on route 572 so that they get notified prior to arriving to a point of interest and learn more about these points if they wish to. The application was built with non-proprietary software such as PhoneGap, Leaflet and various web technologies by a novice developer. After conducting tests, it was concluded that the application satisfied most of the requirements. Hence it was relatively successful but not good enough to be deployed to an application store and made available for public use.

5.1 Meeting the Objectives

All the originally stated objectives were mostly met. The first objective involved a throughout research on everything related to the project. Developer fallowed a step by step approach to do the research, starting from theoretical elements and finishing with more practical aspects of the project. Hence, the research started with gaining a deep understanding on the Cumbria Classic Coaches, mobile applications and open source software. Later, with the information gathered on mentioned theoretical elements of the project, the developer researched necessary tools and technologies needed to build the software. The second objective was to establish the requirements in order to design the software prior to implementation. Design was successful within itself and developer used the most appropriate design techniques for web technologies. However, requirement analysis could have been more successful. A more detailed requirement analysis could have been done on the non-functional requirements. This way, perhaps some of the implementation failures such as offline map implementation could have avoided. The third objective was to pick a software engineering model to deliver the best results. It had been concluded that waterfall model could have been replaced with evolutionary prototyping for better results. However, it is hard to be 100% sure whether or not evolutionary prototyping would have brought more success compare to waterfall model. Consequently, the third objective was neither successful nor unsuccessful. The fourth objective was rather personal. The developer was a novice and unfamiliar with mobile development. New skills were needed to be acquired in a short period of time. This was challenging for the developer especially given the problems faced with open source software as mentioned on the previous chapter. Although not as well as originally hoped for, new tools and languages were learned and a semi successful application was built on time. The final objective was to test and evaluate the final product. This was met by constructing verification analysis and manual tests on the final product.

5.2 Lessons Learned

A lot of valuable lessons were learned as a result of this project. The very first lesson was how hands on practice differed greatly compared to the theory. Developer's expectations were very high at the beginning of the project. New technologies that needed to be learned looked very easy and straight forward during the background research. However, when it was time for implementation, it was realized how challenging it was to especially work with APIs and Plugins. It was also a surprising discovery to learn about how quickly such APIs and Plugins can be deprecated. Learning more about JavaScript and its capabilities was a lesson within itself. The developer was more comfortable with back-end development with Java prior to starting this project. Hence, working with a scripting language came forward as unorganized and unstructured at first. There was also a doubt towards how much could have actually achieved with a scripting language. However, reading Leaflet's library, finding out thousands of open-source API and Plugin code on GitHub brought more appreciation towards JavaScript and how much can be achieved with it. A greater understanding was also gained by building a system from beginning till the end. Developer had never built an entire system prior to this project, so it was a valuable experience to learn how many tools and technologies needed to build a complete system. Working with maps was especially a challenging experience for the developer and as a result a broader understanding and appreciation was developed for the popular navigation apps that it used on a daily life but often taken for granted.

REFERENCES

[1] Aamoth, D (2014). [online] time.com. Available at: http://time.com/3137005/first-smartphone-ibm-simon/ [14.07.2018].

[2] Allensworth. G [online] github.com. Available at: https://github.com/gregallensworth/L.TileLayer.Cordova [01.08.2018].

[3] Babich, N. (2018). [online] smashingmagazine.com Available at: https://www.smashingmagazine.com/2018/02/comprehensive-guide-to-mobile-app-design/ [25.07.2018].

[4] barcelonabusturistic.cat [online] Available at: https://www.barcelonabusturistic.cat/en [24.07.2018].

[5] Baytiyeh & Pfaffman, 2010. Open source software: A community of altruists. Computers in Human Behavior, 26(6), pp.1345–1354.

[6] bigbustours.com [online] Available at: https://www.bigbustours.com/en/london/london-routes-and-tour-maps/ [24.07.2018].

[7] blog.phonegap.com, (2018). Update on the PhoneGap Developer App (iOS). [online] Available at: https://blog.phonegap.com/update-on-the-phonegap-developer-app-ios-99e07e3309dd [06.08.2018].

[8] Constantin et al., 2016. Comparison of Hybrid Cross-Platform Mobile Applications with Native Cross-Platform Applications. Journal of Computer Science and Control Systems, 9(2), p.24.

[9] Cumbriaclassiccoaches.co.uk [online] Available at: https://www.cumbriaclassiccoaches.co.uk [12.07.2018].

[10] Delia, L. et al., 2015. Multi-platform mobile application development analysis. Research Challenges in Information Science (RCIS), 2015 IEEE 9th International Conference on, 2015(June), pp.181–186.

[11] dictionary.cambridge.org. [online] Available at: https://dictionary.cambridge.org/dictionary/english/application-software [14.07.2018].

[12] dictionary.cambridge.org. [online] Available at: https://dictionary.cambridge.org/dictionary/english/mobile-application [14.07.2018].

[13] Dhir, S., 2017. Adoption of open-source software versus proprietary software: An exploratory study. Strategic Change, 26(4), pp.363–371.

[14] docs.phonegap.com. Configuring. [online] Available at: http://docs.phonegap.com/phonegap-build/configuring/ [06.08.2018].

[15] Dooley, J.F., 2017. Software Development, Design and Coding, Berkeley, CA: Apress.

[16] Durham.gov.uk, (2002). [online] Available at: https://www.durham.gov.uk/media/3616/Teesdale-Local-Plan-Chapter-5-Population-and-Housing/pdf/TeesdaleLocalPlanPopulationAndHousing.pdf [12.07.2018].

[17] Eden.gov.uk, (2010). [online] Available at: https://www.eden.gov.uk/media/1691/ravenstonedale-area-profile.pdf [12.07.2018].

[18] Eden.gov.uk, (2011). [online] Available at: https://www.eden.gov.uk/media/1953/edb-area-profile-kirkby-stephen.pdf [12.07.2018].

[19] Egglestonhallgardens.co.uk, [online] Available at: http://www.egglestonhallgardens.co.uk/ [12.07.2018].

[20] Egglestonparishcouncil.org, [online] Available at: http://www.egglestonparishcouncil.org/history.html [12.07.2018].

[21] Ekambaram, V., Sharma, V. & Rajput, N., 2016. Mobile application testing. In Mobile Application Development, Usability, and Security. IGI Global, pp. 25–53.

[22] English-heritage.org.uk, [online] Available at: http://www.english-heritage.org.uk/visit/places/egglestone-abbey/?utm_source=Google%20Business&utm_campaign=Local%20Listings&utm_medium=Google%20Business%20Profiles&utm_content=egglestone%20abbey [12.07.2018].

[23] Foster, E., 2014. Software Engineering, Berkeley, CA: Apress.

[24] Gaff, B.M. & Ploussios, G.J., 2012. Open Source Software. Computer, 45(6), pp.9–11.

[25] Haselmayr, M (2014). [online] forbes.com. Available at: https://www.forbes.com/sites/allbusiness/2014/11/17/heres-why-your-business-needs-its-own-mobile-app/#715b78fd327f [15.07.2018].

[26] html5.sasabus.org [online] Available at: http://html5.sasabus.org/en/#0 [25.07.2018].

[27] Huffingtonpost.com [online] Available at: https://www.huffingtonpost.com/entry/when-building-a-mobile-app-avoid-using-a-hamburger_us_59b0adcae4b0c50640cd649b [25.07.2018].

[28] ionicframework.com [online] Available at: https://ionicframework.com/ [01.08.2018].

[29] itunes.apple.com [online] Available at: https://itunes.apple.com/us/app/barcelona-bus-tur%C3%ADstic/id1238753459?l=es&ls=1&mt=8 [24.07.2018]

[30] itunes.apple.com [online] Available at: https://itunes.apple.com/gb/app/big-bus-tours-interactive/id590746945?mt=8&ign-mpt=uo=4# [24.07.2018].

[31] jquerymobile.com [online] Available at: https://jquerymobile.com/about/ [24.07.2018].

[32] Kumar, G. & Bhatia, P.K., 2014. Comparative Analysis of Software Engineering Models from Traditional to Modern Methodologies. Advanced Computing & Communication Technologies (ACCT), 2014 Fourth International Conference on, pp.189–196.

[33] Kirkby-stephen.com, (2018). [online] Available at: https://www.kirkby-stephen.com/ [12.07.2018].

[34] leafletjs.com [online] Available at: https://leafletjs.com/index.html [24.07.2018].

[35] Middletonplus.org.uk, [online] Available at: http://www.middletonplus.org.uk/ [12.07.2018].

[36] Nielsen, J., 2000. Designing web usability, Indianapolis, Ind.: New Riders.

[37] Phonegap.com [online] Available at: https://phonegap.com/ [23.07.2018].

[38] play.google.com [online] Available at: https://play.google.com/store/apps/details?id=com.bigbustours.bbt&showAllReviews=true [24.07.2018].

[39] play.google.com [online] Available at: https://play.google.com/store/apps/details?id=it.sasabz.android.sasabus [25.07.2018].

[40] play.google.com [online] Available at: https://play.google.com/store/apps/details?id=fr.smartapps.bbt.na&showAllReviews=true [24.07.2018].

[41] progeopiemonte.it [online] Available at: http://www.progeopiemonte.it/multimedia/ [25.07.2018].

[42] Raghunathan et al., 2005. Open source versus closed source: software quality in monopoly and competitive markets. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on, 35(6), pp.903–918.

[43] Ravenstonedale.org [online] Available at: http://www.ravenstonedale.org/ [12.07.2018].

[44] Rust-Smith, D [online] davidrs.com. Available at: http://davidrs.com/wp/phonegap-3-0-leaflet-offline-maps/ [01.08.2018].

[45] sasabus.org [online] Available at: http://sasabus.org/home [25.07.2018].

[46] Shah, U., 2016. An Excursion to Software Development Life Cycle Models: An Old to Ever-growing Models. ACM SIGSOFT Software Engineering Notes, 41(1), pp.1–6.

[47] Sharma, L (2017). [online] toolsqa.com. Available at: http://toolsqa.com/software-testing/difference-between-verification-and-validation/ [31.07.2018].

[48]   Sommerville, I., 2011. Software engineering 9th ed., International., Harlow: Addison-Wesley.

[49]   Pressman, R.S., 2005. Software engineering : a practitioner's approach 6th ed., London: McGraw-Hill.

[50]   Soundcloud.com, (1983). [online] Available at: https://soundcloud.com/mbtech_1434547055436/talk-by-steven-jobs-idca-1983 [14.07.2018].

[51]   statista.com. [online] Available at: https://www.statista.com/statistics/241462/global-mobile-phone-website-traffic-share/ [14.07.2018].

[52]   statista.com. [online] https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/ [14.07.2018].

[53]   statista.com. [online] Available at: https://www.statista.com/statistics/553464/predicted-number-of-smartphone-users-in-the-united-kingdom-uk/ [14.07.2018].

[54]   statista.com. [online] https://www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads/ [14.07.2018].

[55]   Thisisdurham.com, [online] Available at: https://www.thisisdurham.com/explore-durham/durham-towns/barnard-castle [12.07.2018].

[56]   visitcumbria.com [online] Available at: https://www.visitcumbria.com/evnp/brough/ [12.07.2018].

[57]   visitcumbria.com [online] Available at: https://www.visitcumbria.com/evnp/ravenstonedale/# [12.07.2018].

[58]   Visiteden.co.uk, (2018). [online] Available at: http://www.visiteden.co.uk/explore-eden/the-eden-valley/kirkby-stephen [12.07.2018].

[59]   w3.org [online] Available at: https://www.w3.org/TR/html5/introduction.html#background [23.07.2018].

[60]   w3schools.com [online] Available at: https://www.w3schools.com/css/default.asp [23.07.2018].

[61]   w3schools.com [online] Available at: https://www.w3schools.com/jS/default.asp [23.07.2018].