BUSINESS INFORMATION SYSTEMS

Mobile London

A mobile website for planning journeys in London

Nicholas White

25 April 2012

Submitted in partial fulfillment of the requirements for the award of the degree of Business Information Systems of the University of Portsmouth

Copyright

Abstract

This report describes the development of a mobile application, which provides Transport for London's customers with up-to-date information. It researches how best to deliver an application to a wide range of users as a web or native application, and their best frameworks. The developed product uses the NodeJS server technology to connect with APIs provided by TFL and Google, whilst displaying the information with the web framework jQuery Mobile with HTML5 functionality in order to display journey planner, bus stop countdown, live tube and weekend information.

Acknowledgements

I would like to thank my family for their continual support during all of my studies and my girlfriend Zoë Vennard for all of her encouragement. I would also like to thank Richard Boakes my project supervisor, for his indispensible guidance during the development and documentation of this project.

Table of Contents

Acronym Definitions

Below is a list of any different acronyms used within this paper.

AJAX	Asynchronous JavaScript and XML
API	Application Program Interface
CSS	Cascading Style Sheets
CSV	Comma-separated values
DOM	Document Object Model
GPS	Global Positioning System
HTML	Hypertext Mark-up Language
HTTP	Hypertext Transfer Protocol
I/O	Input / Output
iOS	Apple's mobile phone Operating System (iPhone Operating System)
JSON	JavaScript Object Notation
NPM	Node Package Manager
OS	Operating System
SSM	Soft Systems Methodology
TFL	Transport for London
W3C	World Wide Web Consortium
WWW	World Wide Web
XHTML	Extensible Hypertext Mark-up Language
XML	Extensible Markup Language

List of Figures

List of Tables

1 Introduction

This chapter discusses and outlines the project and provides a description of the project, outlining the aims, objectives and any constraints.

1.1 Description

This engineering project's client is any member of the public who uses Transport for London in order to travel. Currently TFL offers a mobile site with use of a Journey Planner, Bus stop countdown and service updates. The mobile site provided by TFL has a basic look and feel, and therefore with the use of APIs, a mobile site using HTML5 and JavaScript can help provide a better option for mobile use. Clients do have the option of using native applications. However, most of these applications only provide one type of usage (such as only Bus stop countdown); rather than a wide range of services that are provided generally for users, while using a desktop computer on TFL's main webpage.

1.2 Aims and Objectives

The aim of this project is to create a new website suitable for mobile use, to provide an easier and more intuitive option for retrieving information from TFL, which is available on a wide range of devices.

The objectives of this project can be broken down into the following:

1. Allow the use of the user's current location as a start or end point
2. Allow users to store favourite travel journeys
3. Check current tube line services
4. Check weekend tube line services
5. Let users check Bus stop countdown information
    - By either Postcode or Current location
6. The application should be available across multiple platforms
7. Provide users with a simple design layout

The name of this application is Mobile London and it is hosted on the website address www.MobileLondon.org. This name was decided upon, as the application will be used strictly within the Greater London area and run generally on mobile devices.

1.3 Project constraints

The limitations for this project, due to building a mobile web app, are that the site should be available for as many different mobile devices as possible. Using a user's current location can also

cause problems as they have to allow the site to have access to their location, and the accuracy of the user's location may not 100%.

Building for a mobile device means that minimising data transferred between the server and the client needed to be looked at, as mobile data services can be costly and bandwidth speeds can be slow.

The application uses APIs to retrieve information that the user requests. This causes a problem as there is no control over the state of these APIs whether they continue to exist in the future or the structuring of how the data is returned to the server. APIs from TFL and Google are used; TFL have in the past terminated their Journey Planner API. However, this risk will always exist with this type of project without having any other access to TFL journey data. The data structure provided with these APIs may also change with Journey Planner API still in beta. This means that maintenance will need to be carried out as it is likely that the structure of the API may change.

Due to the funding of this project, costs needed to be kept at a minimum. To keep this a viable option the use of open-source code and software were used. Amazon Web Services EC2 is currently running a 12 month free tier program. This is used for the hosting of the service, bandwidth, size and CPU has been kept lower than needed for a large scale project. However, for the viability of this project at its current state this was needed.

To complete this project the server technology known as NodeJS was used and with no prior knowledge of this technology, caused the project to take longer. This learning curve needed to be added into the length of the potential build of the project, as was testing of modules to ensure the level needed was gained. The mobile framework jQuery Mobile also needed to be learnt during the build. However, the documentation for this is superior to Node's which therefore took less resource to learn.

2 Literature Review

The literature review was used in order to explore academic articles of different web development techniques; including mobile web standards, web frameworks, server technologies and web technologies.

2.1 Mobile Web Standards

This section looks at some of the mobile web standards that have been set by W3C. W3C's mission statement is "W3C's mission includes ensuring that the Web be available on as many kind of devices as possible. With the surge of powerful mobile devices in the past few years, the role of the Web as a platform for content, applications and services on these devices is increasingly important." (The Web and Mobile Devices).

2.1.1 Bandwidth and cost

Mobile phone providers in the United Kingdom are currently only providing 3G coverage; 4G is currently on trial. However, the area that is covered is only 15 square miles in the area containing Canary Warf and Soho. "Britain's 4G or long-term evolution (LTE) upgrade, expected to begin in earnest in 2013" (Garside, 2011).

This therefore means that the large base of users is still using 3G (or still on 2G depending on the area coverage). 3G Smartphones have been around since 2003 (one of the first was the Nokia 7600); "3 UK was the first operator to roll out a 3G network and services in the UK (March 2003) and has over 7.5 million 3G subscribers as of 3 August 2011." (Telecommunications).

This then means that bandwidth should be considered when building a mobile web application; taking into account how much data needs to be downloaded by the user at each request for speed, as well as cost.

2.1.2 Delivery context

W3C recommend a standard for delivery of content on the web for mobile technologies. This recommendation is in order to provide a 'One Web' experience for all users on all mobile devices. These recommendations can cause conflict for this project as no client-side scripting should be used, as well as CSS1, CSS2 and XHTML (Table 1 - Default Delivery Context). Keeping this project to this standard could hinder the delivery of the data and the user experience. W3C do acknowledge this, "Content providers are encouraged not to diminish the user experience on those devices by developing only to the DDC specification, and are encouraged to adapt their content, where appropriate, to exploit the capabilities of the actual device." (Rabin & McCathie, 2008).

Usable screen width	120 pixels, minimum
Markup Language Support	XHTML Basic 1.1 [XHTML-Basic] delivered with content type application/xhtml+xml.
Character Encoding	UTF-8
Image Format Support	JPEG GIF 89a
Maximum Total Page Weight	20 kilobytes
Colours	256 Colours, minimum
Style Sheet Support	CSS Level 1 [CSS]. In addition, CSS Level 2 [CSS2] @media rule together with the handheld and all media types (see CSS 2 Media Types).
HTTP	HTTP1.0 or HTTP1.1
Script	No support for client side scripting.

(Rabin & McCathie, 2008)

Table 1 - Default Delivery Context

This artefact was created optimised for an iPhone running the OS iOS5. With this in mind this OS has the capability of running HTML5, CSS3 and client-side scripting, which means that keeping to a 'One Web' standard would not be beneficial to the user.

2.1.3 Page Layout and Content

Mobile web applications must be different as a whole compared to the desktop version (if one exists); taking into account the content populating the page as well as the layout. Limiting the information that is displayed to what the user needs and wants is vital; therefore no extra advertising or extra information about different topics should be added.

One important aspect of the mobile web application that needs to be different is the size of the page that is being displayed. Due to mobile device screens being so small there are two different options to go with this; pagination or scrolling.

Pagination can be handled either server-side or client-side. Client-side, there are two different options. Firstly, having all the information loaded into the DOM and then using JavaScript to allow the client to filter through the information on the page, to the information they require. Another option is to use AJAX, a hybrid using server-side and client-side technology. This then allows the client to fetch data as they filter through the pages, retrieving data from the server on each request. However, this can be a bad experience for the user as they may need to click multiple times in order to get the information that they require.

Scrolling is the other option with small screen devices. For a mobile device, only allowing the user to scroll up and down is best. As with clicking through pages (with pagination), the problem with scrolling is that, it can become tiresome and the user is likely to get bored and stop using the site. A

key point when using this method is to ensure that the most relevant data is higher up the page with less important information at the bottom.

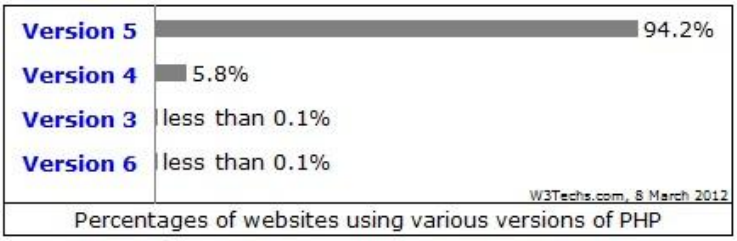
There is no set standard between pagination and scrolling set by W3C. Instead they state that "The balance between pagination and scrolling is partly a matter of taste and partly a matter of necessity. Devices with severe memory restrictions can only have small pages delivered to them." (Rabin & McCathie, 2008).

2.2 Server technologies

This section looks at two different server technologies Node.JS and PHP, in order to come to a conclusion about which server technology to use.

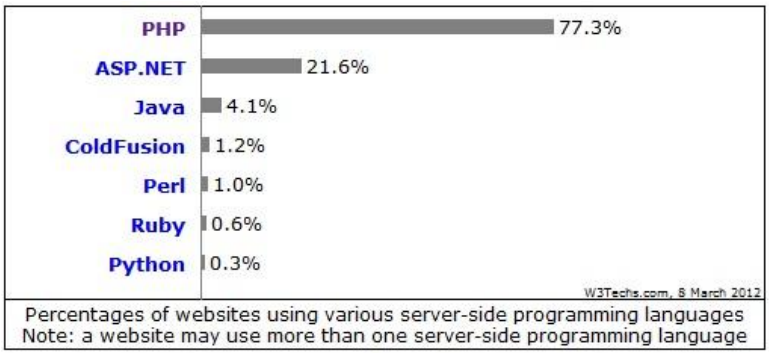
2.2.1 PHP

The server technology PHP was originally created by Rasmus Lerdorf in 1995, since then version 6 of PHP has been released. However, PHP5 is currently the most used version of them all (Figure 1 - PHP Version Statistics). The update to PHP5 was launched on 13th July 2004 and since its original launch in 1995 PHP has become the most widely used server-side language used on the WWW, currently standing at 77.3% with ASP.NET coming in second (Figure 2 - Server-side Language Statistics).



(Usage statistics and market share of PHP for websites, 2012)

Figure 1 - PHP Version Statistics



(Usage of server-side programming languages for websites, 2012)

Figure 2 - Server-side Language Statistics

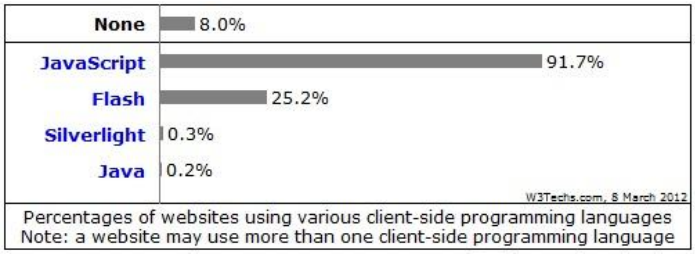
Looking at Figure 2 shows that PHP has a huge market share for server-side programming. These figures do not prove that PHP is the best programming language to use in every situation. It is important to look at which application is going to be built, developers' preferences and how easy it is going to be able to maintain.

PHP as a community on the WWW is huge, with developers providing tutorials and help via different sites (such as Stack Overflow). However, PHP can be a blocking I/O model. This can cause problems when creating real-time applications over the web, with users now expecting to communicate with servers constantly and instantly via a mobile device, desktop or other devices connected to the web. However, this does not mean that all websites need to be a real-time application connecting with users.

"JavaScript is used by 91.7%" (Usage of client-side programming languages for websites, 2012) with JavaScript being the most used client-side programming language currently used on the web, looking at PHP and JavaScript is necessary. Parsing variables from PHP to JavaScript is possible as it is executed on the server-side (using AJAX or JSONP). However, parsing JavaScript to PHP is not possible without the use of AJAX, GET or POST using a form (such as a hidden field). This can therefore be tricky, interacting between what the user does and having the data on the server.

2.2.2 Node.JS

The Node.JS (also known as Node) platform was originally created by Ryan Dahl in 2009 and has been built on Google Chrome's JavaScript runtime "Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices." (node.js). JavaScript as a server-side programming language makes sense as the language client-side is used exponentially on the internet, currently on 91.7% of sites (Figure 3 - Client-side Script Statistics). This shows how much developers now use JavaScript and already knowing the programming language means that they do not need to learn a whole new language but in return a new concept.



(Usage of client-side programming languages for websites, 2012)

Figure 3 - Client-side Script Statistics

Due to Node being event-driven rather than handling incoming traffic with multiple processes at once, means that the server can run at a high speed. Node uses JavaScript programming which is now widely used alongside HTML and CSS for creating websites. With Node using JavaScript it allows the developer to use the same language client and server side. This can have a huge advantage, as it passes information between the two. The Node community is growing vastly with developers contributing modules, through a system that Node uses, NPM. This allows developers to download modules easily on their server to implement and use.

As the JavaScript language uses callbacks, it allows developers to wait until the server is ready to run a function and retrieve the necessary data from the callback. “Rather than handling incoming traffic with many different processes (à la classic Apache) or threads (modern Apache, as well as some other servers), Node.JS handles all incoming connections in a single process and a single thread.” (At the Forge - Node.JS, 2011) This makes Node a non-block I/O model which in return helps the application to be more lightweight and efficient.

Node allows for a faster server for such things as real-time web applications where multiple users are interacting with the server at any given time, “Node’s evented I/O model freed us from worrying about locking and concurrency issues that are common with multithreaded async I/O.” (Allamaraju, 2011). This way, paradigm for servers is a new concept compared to those which have been around and used widely on the internet by websites. Due to this, it forces developers to re-think the way they build applications for Node compared to other languages such as PHP. Rather than being able to fire functions one after the other, Node means that code needs to be smaller, nested and forces the code to use the callback features of the JavaScript language, to ensure that any variables or functions that need to first of all run, are completed.

Node as a web server currently has no set standards for most parts. Instead, Node relies on modules to be created by the developers using the server technology. This in return could be seen as a bad thing for developers to then use, as multiple frameworks may be used and need to be tested, depending on what the developer requires the server to do. One web framework however, does seem to be more widely used than the others. ExpressJS (see http://expressjs.com/). This web framework uses JADE which is a template engine used with Node to output HTML from the server.

Node as a server can be configured to have multiple custom events. These events can be defined by the developer based on data parsed from the client to the server, using an I/O socket. Using this approach it allows apps to be built using both client and server side technology. This is unlike other

programming languages such as PHP, which does not allow client-side code to execute server-side code.

Using single threads for the I/O model, Node prevents concurrency running code sequentially; so that developers no longer need to worry about locking of data. However, due to this when developing with Node code, it requires a lot of computation to be written asynchronously.

2.2.3 Node.JS against PHP

Due to the vast amount PHP is used on the internet for creating websites, Node is compared to PHP which is not necessarily the correct thing to do, as they are both different technologies. However, due to the amount PHP is used this comparison can be understood.

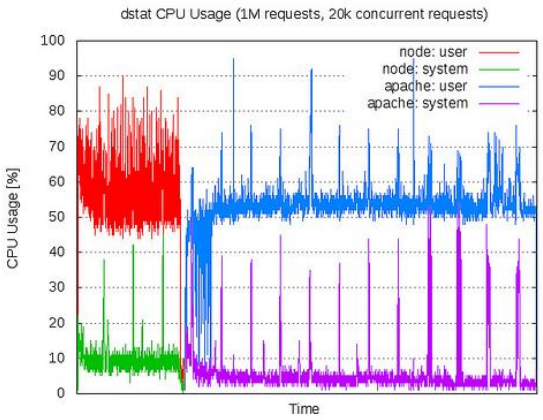
Benchmark tests have been completed in order to compare Node and PHP. Node as expected, is faster in both benchmark tests run by two different sites (Table 2 - Node.js Benchmarks) and (Figure 4 - CPU and memory usage comparison) shows that Node is faster.

Platform	# requests/sec
PHP (via Apache)	3187.27
Static (via Apache)	2966.51
Node.js	5569.30

(Synchros Interactive, 2010)

Table 2 - Node.js Benchmarks

These tests provide statistics showing that Node is faster than PHP. In addition to this, Node allows developers to run the same code client and server side, and developers to be able to initiate server code client-side with the use of sockets. This therefore makes Node a much better server technology to use for real-time systems than PHP, particularly if speed is of a vast importance for the site.



(Maciej)

Figure 4 - CPU and memory usage comparison

2.3 Web and native apps

Since the introduction of Smartphones, web and native apps have grown exponentially with the number of Smartphones used all around the world.

2.3.1 Native application

Native apps are phone specific meaning that the app can only be used on a certain type of phone (such as an iOS application is specific to Apple's iPhone). Apps are required to be written in various coding languages depending on OS'.

Company	3Q11 Market Share	3Q10 Market Share
Android	52.5	25.3
Symbian	16.9	36.3
iOS	15.0	16.6
Research in Motion	11.0	15.4
Bada	2.2	1.1
Microsoft	1.5	2.7
Others	0.9	2.5

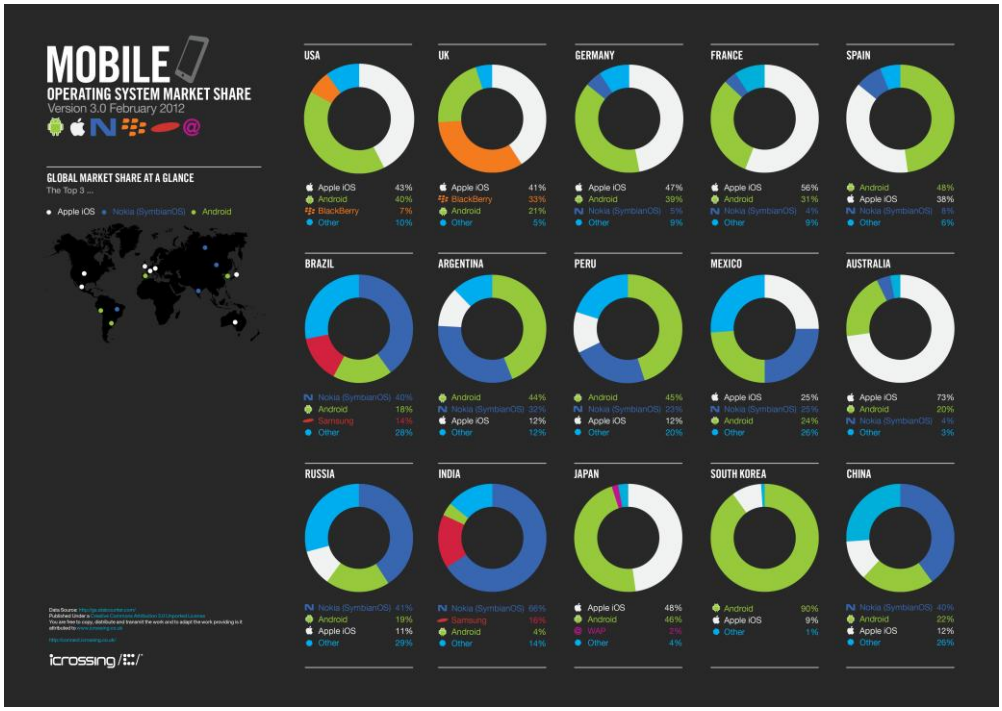
(Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011; Smartphone Sales Increased 42 Percent, 2011)

Table 3 - Worldwide Smartphone Sales to End Users by Operating System in 3Q11

What OS should a business create their app for? Based on the market share (Table 3 - Worldwide Smartphone Sales to End Users by Operating System in 3Q11) the Android OS currently controls just over half the market. However, this does not necessarily mean that this is the correct OS for their app. One major consideration an organisation needs to look at is their demographic. For example, if the app is for Transport for London, it is only necessary to look at the United Kingdom, rather than the world. When looking at the United Kingdom only (Figure 5 - Mobile Operating System Market Share) we can see that actually iOS controls most of the market share with 41% whereas Android only controls 21% of the market share.

Due to the vast number of OS' that users may be using, is creating a native application necessarily the best option? The usage of apps by users is huge. Apple recently announced that over 25 Billion apps (25 Billion app countdown) have been downloaded whilst running a competition to find the user who downloaded the 25 billionth app.

There are many different types of apps being created for all different OS'. Some businesses are creating the same applications for many different OS' such as WhatsApp, which is a cross-platform instant messaging service. This app connects five of the main OS' together to allow users to send pictures, locations and text. This is a prime example of a native app that has had to be produced for different OS'. To combat this, using a mobile development framework such as Adobe Systems PhoneGap, allows developers to write HTML5, CSS3 and JavaScript. This approach for WhatsApp could have saved development time and money, creating one app that works across all Platforms.



(Lyons, 2012)

Figure 5 - Mobile Operating System Market Share

2.3.2 Web application

Web applications can be a great approach for an organisation, as a first step to give access to users via a mobile device. The reason for this is that mobile devices can all render HTML and now with the push towards HTML5 (Figure 6), Smartphones are growing as a standard.

Creating a web application utilising HTML5, CSS3 and JavaScript gives developers the option of using PhoneGap to create an initial Android, BlackBerry, Windows Mobile and iOS application and then to build a complete native app in the future.

(W3C HTML5 Logo)

Figure 6 – HTML 5 Logo

With the introduction of HTML5 it now allows developers to use Geo-location capabilities to receive users' current location, Local storage to store data connected to the website, and Local session storage to store data specifically to the current session, which will end once the browser is closed. HTML5 full capabilities are not necessarily compatible with every mobile device such as the Notifications API. This API is currently only supported by BlackBerry's tablet running the OS 2.0+, while the Network Information API is currently only compatible with Android's browser. Due to this it all depends on what the application requires from the user and what features are needed.

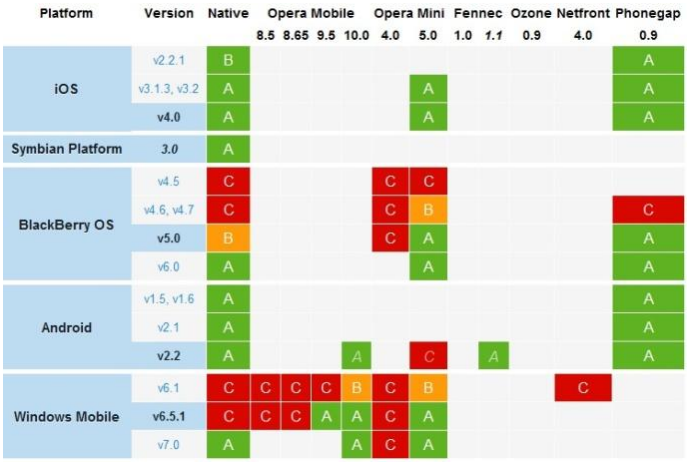
For mobile web application development there is the option of using pre-created mobile frameworks. This can make it easier to develop for multiple web browsers maximising the amount of mobile devices that are available. These different mobile frameworks use HTML5, CSS3 and JavaScript.

- jQuery Mobile
- Sencha Touch
- M-Project
- NimbleKit
- Wink

2.3.2.1 jQuery Mobile

This mobile framework is a part of the jQuery Project which has also created jQuery, jQuery UI, QUnit and Sizzle. It has been set up to work alongside jQuery allowing developers to use the same functionality as they would find building desktop sites using jQuery. It has been built to work with HTML5 and CSS3.

A main objective of this project is to allow developers to create cross-platform web applications "jQuery mobile framework takes the 'write less, do more' mantra to the next level: Instead of writing unique apps for each mobile device or OS, the jQuery mobile framework allows you to design a single highly-branded web site or application that will work on all popular Smartphone, tablet, and desktop platforms." (jQuery Mobile).



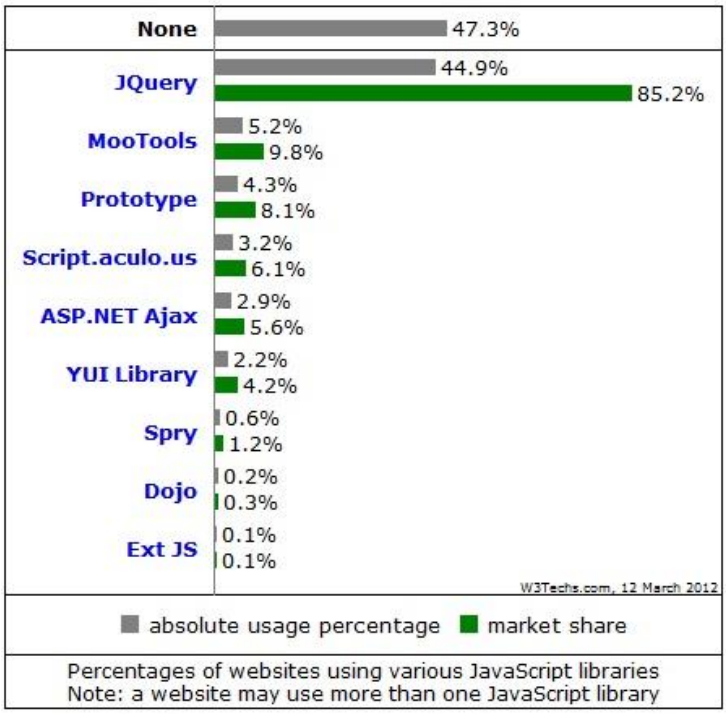
(Original graded browser matrix)

Figure 7 - Mobile Graded Browser Support

Figure 7 shows that using jQuery Mobile creates a cross-platform mobile web application that works across multiple OS' as found in Table 3. jQuery Mobile can be used for developing with PhoneGap;

this can therefore mean that a web application can be created as well as a native app that has extra features.

The use of the jQuery library, which powers jQuery Mobile currently, has the highest market share on the internet (Figure 8 - Usage of JavaScript libraries for websites). Looking at this using jQuery Mobile as a mobile framework can be highly beneficial for developers. Future developers are more likely to therefore know jQuery as a library, which means it should be easier for other developers to learn jQuery Mobile and maintain the site in the future.



(Usage of JavaScript libraries for websites, 2012)

Figure 8 - Usage of JavaScript libraries for websites

2.3.2.2 Sencha Touch

Another mobile framework is Sencha Touch (currently running version 2). This mobile framework uses HTML5 and JavaScript to create the mobile web application. “Sencha Touch 2, a high-performance HTML5 mobile application framework, is the cornerstone of the Sencha HTML5 platform. Built for enabling world-class user experiences, Sencha Touch 2 is the only framework that enables developers to build fast and impressive apps that work on iOS, Android, BlackBerry, Kindle Fire, and more.” (Sencha Touch 2, 2012).

```
<script type="text/javascript">
        new Ext.Application({
                launch: function() {
                        new Ext.Panel({
                                fullscreen: true,
                                html: 'Hello World!'
                        });
                }
        });
</script>
```

(Pearce, 2011)

Figure 9 - Sencha Touch Hello World

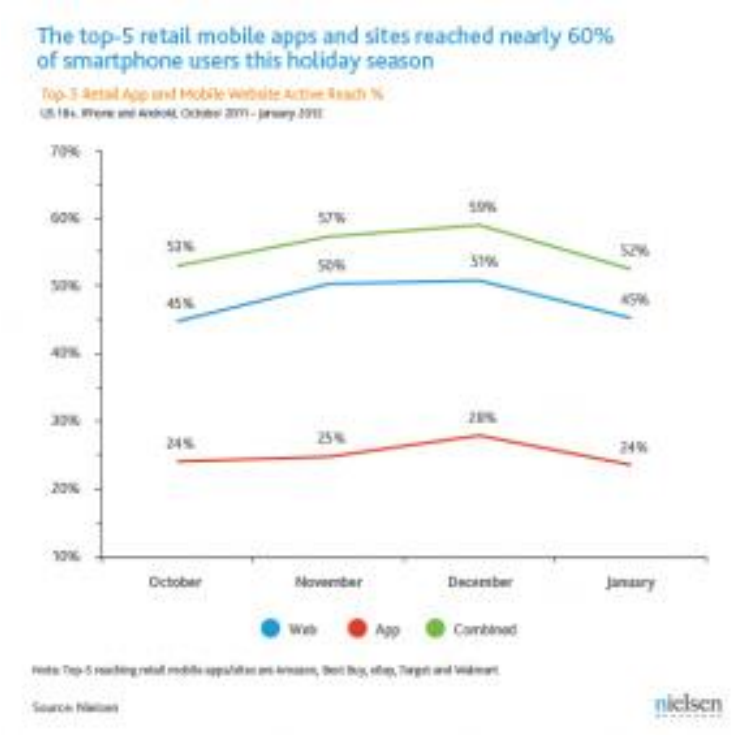
JavaScript (Figure 9 - Sencha Touch Hello World) is used across the WWW by such a vast number of sites that this can still be a good fit for developers. Rather than writing HTML, only JavaScript is written and Sencha Touch renders the page using HTML5, based on the ExtJS framework.

Using Sencha Touch makes it easier to develop native apps when using HTML5, as Sencha Touch offers 'Native Packaging' which allows further interaction with a user's mobile device, such as the use of the camera. This allows developers to create a web application with basic functionality using the same code packaged as an application, to put on the Apple or Android store and allow more functionality if required.

2.3.3 Web or native app

But what is best for an organisation to have, a web or native application? It is best for an organisation to have both options for their client. Every individual has their own preference between the two, depending on the OS, their age, how they interact with it etc.

Initially creating a web application for economic and speed purposes would be the best idea, providing the application to more users, whilst using a mobile framework that creates a web and native app. As it can be seen in Figure 10 web apps are currently used more by users than native apps. This therefore provides an argument that simply ensuring that a mobile site using web framework such as Sencha Touch can be more beneficial. However, these statistics are for M-Commerce which could therefore mean that it depends on what the app is for.



(Luden, 2012)

Figure 10 - M-Commerce Web and native app %

2.4 Methodologies

The aim of this section is to identify possible methodologies to be used for the build of the artefact, in order to be able to come to a conclusion on the most viable option for this type of project.

2.4.1 Soft Systems Methodology

SSM was originally created by Peter Checkland, it is a "systemic approach for tackling real-world problematic situations" (Checkland, 1999).

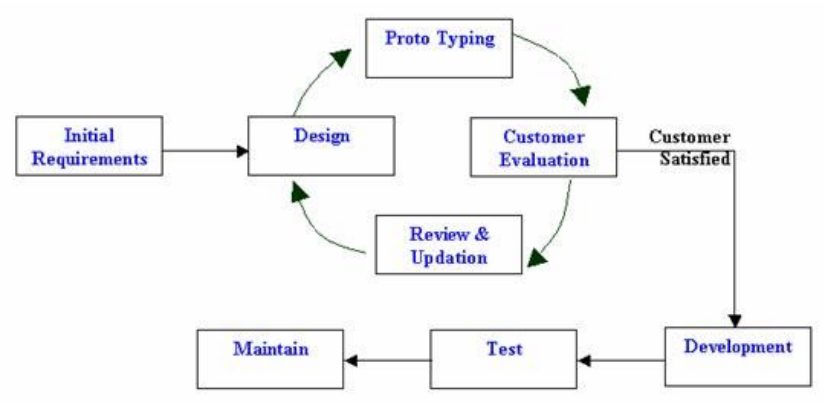
SSM is generally broken down into 7 different steps. These are:

1. "Identifying the problematic situation that it is desired to intervene in
2. Researching the situation and building a 'rich picture' (interpretive representation) of it
3. Selecting perspectives and building 'root definitions' (key processes that need to take place within the desired system)
4. Developing a conceptual model of the change system
5. Comparing the model with the real-world situation
6. Defining the changes to be implemented
7. Taking action" (Lester, 2008)

This methodology requires an endless cycle, as the idea is that no problem is ever solved. You can only have a resolution, which for a project building an artefact in this manner would cause no artefact to be finished and built, causing a problem for this project.

2.4.2 Prototyping

"Software prototyping is somewhat similar. It produces a "throw-away" solution that is designed for the sole purpose of verifying user functionality and for demonstrating capability." (Bowman, 2009). A visual representation showing a schematic representation of the prototype methodology can be seen in Figure 11.



(The Concept of Proto-Typing in the System Development Process)

Figure 11 - A schematic representation of proto-typing

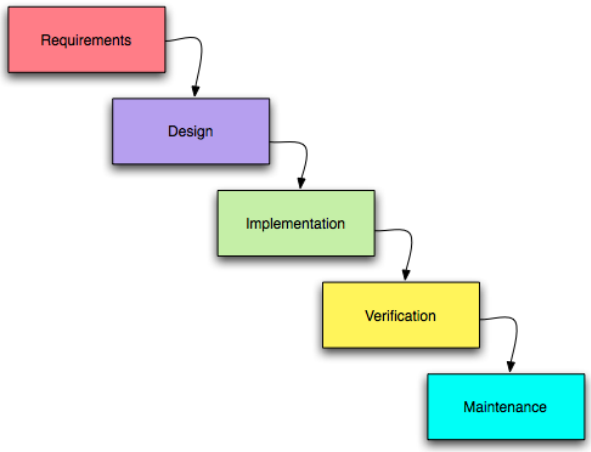
This methodology allows the artefact to be created to an initial stage (a prototype) and then tested against potential users who have no link with the project. A prototype does not need to be a fully functional piece of software (for example) the use of wireframes "also known as a page schematic or screen blueprint, is a visual guide that represents the skeletal framework of a website" (Brown, 2006), can be used for this stage which allow users to click through to check that the navigation, layout and functionality of the site meets their needs.

Wireframes can be used as a paper representation or using software (such as Flairbuilder). The general difference between these two approaches is that when using a paper based system a user needs to be guided through; whereas using some software, a full click navigation system can generally be used. This can help with the viability of testing, as anyone would therefore be able to test the wireframes anywhere in the world, unguided.

2.4.3 Waterfall Model

This model has five different stages; Requirements, Design, Implementation, Verification and Maintenance (a visual representation of this can be seen in Figure 12).

The methodology (when unchanged) has a set model expecting each stage to be completed sequentially. Using this can cause problems for the build of a project, as one stage may take longer than expected. For example, if further requirements were found once the implementation stage had been completed, it would mean that the entire design stage would need to be restarted rather than adding the further requirements on in the future.



(Smart)

Figure 12 - Waterfall model

2.5 Demographic

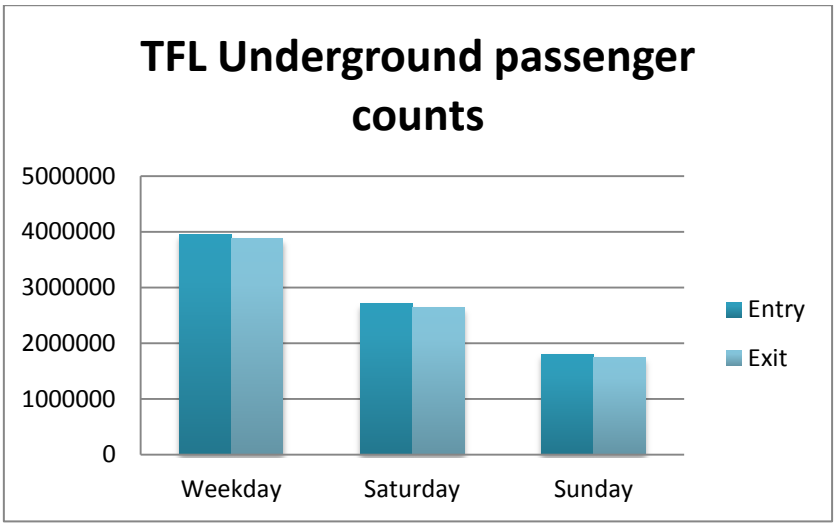
This project aims to deliver all TFL's customers with up-to-date information supplied from TFL using a Smartphone device. To ensure the viability of the project and its usefulness, it is essential to look at the potential market figures.

In order to look at the potential market two different statistics have been analysed, TFL's underground (tube) and buses within England (broken down into London). The reason for this is that the journey planner for travelling around London will generally use tube, buses and train. However, it wasn't possible to accumulate train statistics for London at this time.

Figure 13 shows that there are approximately 4 million people using the tube on a weekday. However, it will need to be taken into account that these are entries and some people travelling may enter more than one tube station on a single day. This number provides us with approximately 4 million potential users. Due to using a cross-platform website this shouldn't narrow this number as much as only providing an iOS application (which would potentially be 41% of 4 million coming to 1.64 million).

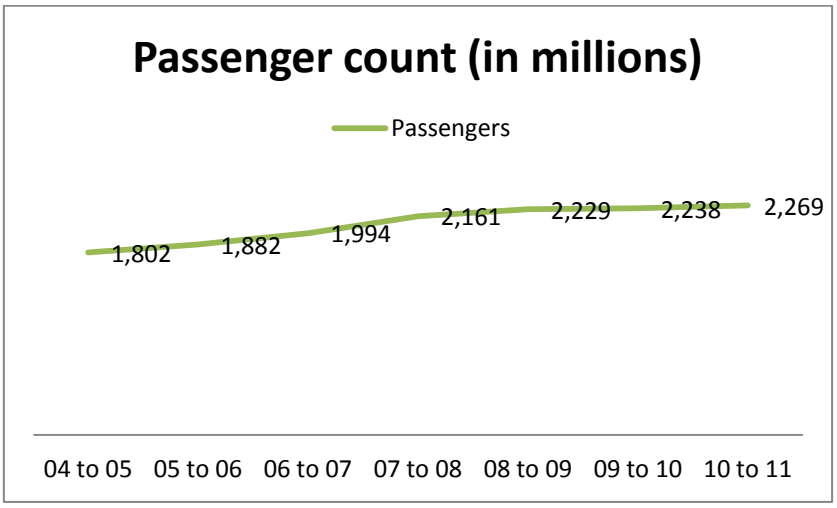
As seen in Figure 14 bus passengers have increased each year within London between 2004 and 2011, by approximately 400,000 passengers. Some of these users may contribute to some of the

statistics in Figure 13 however, not all will do so. This also provides us with potentially 2.3 million (approx.) users of the bus stop countdown functionality for this project.



(Guidelines and Support)

Figure 13 - TFL Underground passenger counts



(Quarterly bus statistics q4. 2011, 2011)

Figure 14 - Bus statistics 2011

3 Artefact Design

This chapter looks at the design of the final artefact, identifying the design methodology, requirements, wireframes and the proposed solution.

3.1 Design Methodology

Since reviewing some of the methodologies available in the Literature Review 2.4 Methodologies, it was decided to use the prototype methodology (as reviewed in 2.4.2 Prototyping). The reasoning for this was that it should help provide a solution best fit for the potential users of this artefact.

In order to create a prototype wireframes were used to create the layouts, once printed they were tested and the navigation of the artefact and potential functionality evaluated. After testing, results were used to amend the wireframes where changes in the design, or the layout of the system needed to be changed.

3.2 Requirements

The requirements of the final artefact needed to be outlined before build. The requirements can be found in Table 4; all of the requirements have been ordered by importance. This was required due to the time constraints for the build process where it was not possible to complete some of them.

3.2.1 Importance

	Level of Importance
Journey planner	1
Bus stop countdown	2
Live tube status	7
Weekend tube status	8
Favourites	4
Cross-platform	5
Simple design	3
Data size	6
Server speed	9

Table 4 - Artefact requirements importance

Table 4 sets the order in which the project build was carried out by starting with the journey planner and finishing with the speed of the server.

To do this the functionality required has been looked at to see what would be most beneficial to the potential end clients. The 'Journey Planner' has been deemed more important than other features due to the potential number of users. Whereas the bus stop countdown functionality may have fewer users, it is however, more important than favourites and cross-platform due to the number of applications (as previously stated), that only provide one piece of functionality. As the favourites are

beneficial for the users of both the journey planner and bus stop countdown, it therefore seems a natural fit for these to go hand in hand.

The cross-platform should be important for the build of the project, as a main gain for this application compared to those already created and on the market, is that they only work with one OS. Making this application cross-platform expands the potential market share to multiple devices, increasing the potential number of users.

3.2.2 Journey Planner

The journey planner should allow users to search by postcode, current location or train / tube station. Users should have the possibility of searching for future or current date and times with the option of setting either arrival or departure times.

Each search should display all routes returned from TFL, displaying the following details:

- Departure time
- Arrival time
- Estimated duration of travel
- Amount of Inter-changes (times need to change public transport)
- Detailed information about each individual routes

3.2.3 Bus stop countdown

The bus stop countdown should allow users to search for a bus stop by either postcode or their current location; providing the nearest ten bus stops to the location and displaying the bus stop countdown information.

Bus stop information should display the following information:

- Distance between starting and bus stop location
- All possible bus routes from the bus stop
- Bus, destination and length of time needed to wait

3.2.4 Live tube status

This feature should display the current up-to-date information from TFL about the current status of the tube whether the line is closed, open or has any possible delays. The data should be displayed to the user using a simple navigation preventing users from needing to use search facilities.

3.2.5 Weekend tube status

The weekend tube status feature should be displayed the same as 3.2.4 Live tube status however, displaying TFL's information for the coming (or current) weekend; whether the line is closed, open or if the user should expect any delays.

3.2.6 Favourites

Users of the journey planner application should have the facility to save favourite bus stops, in order to receive up-to-date countdown information provided by TFL; as well as favourite journeys. These journeys should allow the user to save the origin and destination of their journeys without needing to register or sign into the website.

3.2.7 Cross-platform

The main clients for this artefact are going to be ones living within London, and the United Kingdom. Therefore looking at Figure 5 the main OS' that those clients will be using are iOS, BlackBerry and Android. These three OS' should be used as a basis for making the artefact cross-platform.

3.2.8 Simple design

One of the issues that the current TFL mobile site has, is the design as simple as it may be, is not aesthetically pleasing as it could be. The simple design also uses scrolling in areas where it is not necessarily needed. Figure 15 shows TFL's journey planner design. The form shows some information that is not necessarily always needed by the user, such as the time and date details.

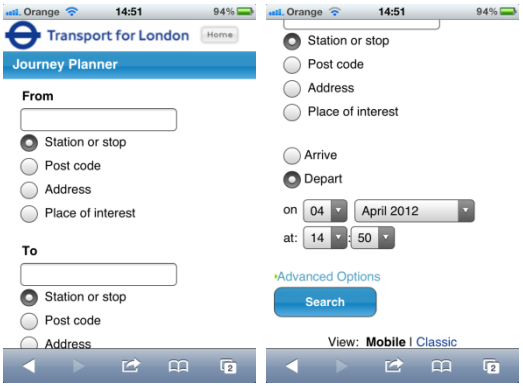


Figure 15 - TFL Journey Planner Design

An option to combat this problem would be to use an accordion type system, which hides some information to the user on the load of the page. However, with the use of an action button the hidden data will be displayed.

3.2.9 Data size

It can be assumed that potential users will generally be using the application with a 3G connection. With this in mind client-side code should be kept to a minimum, whilst using minified versions of code where possible (such as jQuery), displaying to the user only what is relevant.

3.2.10 Server speed

Users expect information to be displayed as quickly as possible (if not instantaneous) from the internet “1.0 second is about the limit for the user's flow of thought to stay uninterrupted, even though the user will notice the delay” (Nielsen, 1993). Therefore connection speeds between the server and other application servers as well as processor speed, needs to be considered. With the high number of potential users the server should be able to deal with this load, as well as potential bottlenecks during peak times (rush hours).

3.3 Proposed Solution

The proposition for this artefact is to create a cross-platform journey planner application. This therefore means that a native application is not ideal, as multiple artefacts would need to be created. Using a mobile framework is thus needed, due to the functionalities used within this application being equipped with HTML5, CSS and JavaScript. This therefore narrows the selection between Sencha Touch 2 and jQuery Mobile.

When choosing between Sencha Touch 2 and jQuery Mobile, it is necessary to look at both time constraints and previous knowledge. Previous knowledge of jQuery provides an easier understanding of jQuery Mobile due to the same foundation for code working alongside HTML, whereas Sencha Touch is based on a JavaScript framework, where no previous knowledge exists. This in return means that building a Sencha Touch 2 artefact will generally take longer and due to time constraints of this project, using the jQuery Mobile framework will be a better option.

Using the jQuery Mobile framework with this proposed solution causes the application to be hosted on the web, which requires a server technology to be chosen which best suits this type of application. The only previous knowledge of server technologies currently possessed is PHP. However, the other option since doing the literature review is to use NodeJS, and since the JavaScript language is already possessed, the learning curve will be the use of this language as a server and the asynchronous code structure. Due to the speed of NodeJS being a factor that is greatly needed for users, this learning curve needed to be factored in within the time constraint for this project.

3.4 Workflow

The workflow a user will find when using the artefact can be seen in Figure 16. This workflow has been used as a basis for creating the wireframes for the prototype, showing how users navigate around the site using the different functionality across the application.

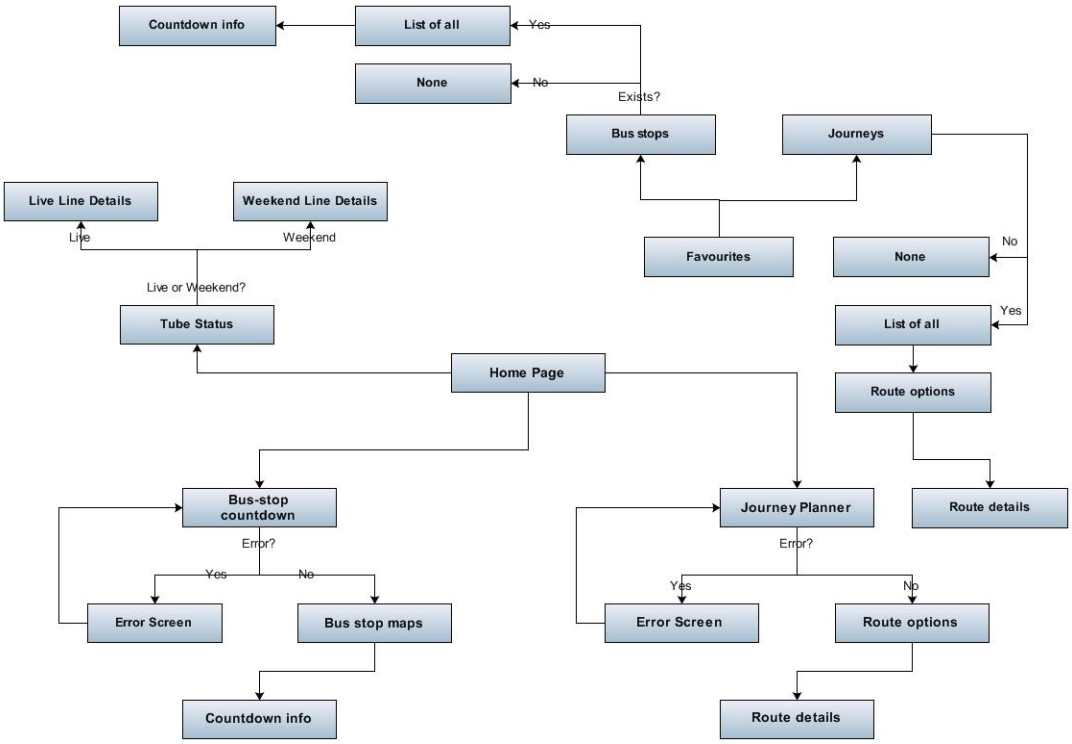


Figure 16 - Workflow

3.5 Wireframes

Since the proposed solution is using jQuery mobile, wireframes were built with this in mind, keeping to the same style as jQuery Mobile (see Figure 17 - Wireframes Part 1 and Figure 18 - Wireframes Part 2). The wireframes were tested by potential users to gain an insight into what they expected from the navigation of the site as well as the information, to be displayed. The testing section of these wireframes can be found in 4.7.1 Prototype Testing; initial wireframes used can also be found in the appendices (7.1 Initial Wireframes).

The testing showed that users found the application was simple to use and that the navigation was as they expected. One issue raised with the bus stop countdown however, was that the user might not necessarily know exactly what bus stop they want to look at, by name. This issue has since been amended as demonstrated in Figure 17, the nearest ten bus stops are shown via a map indicating where their estimated location is (via Postcode or Geo-location). This has been done to enable the user to be able to visually look at their location and the bus stops surrounding them from a bird's-eye view.

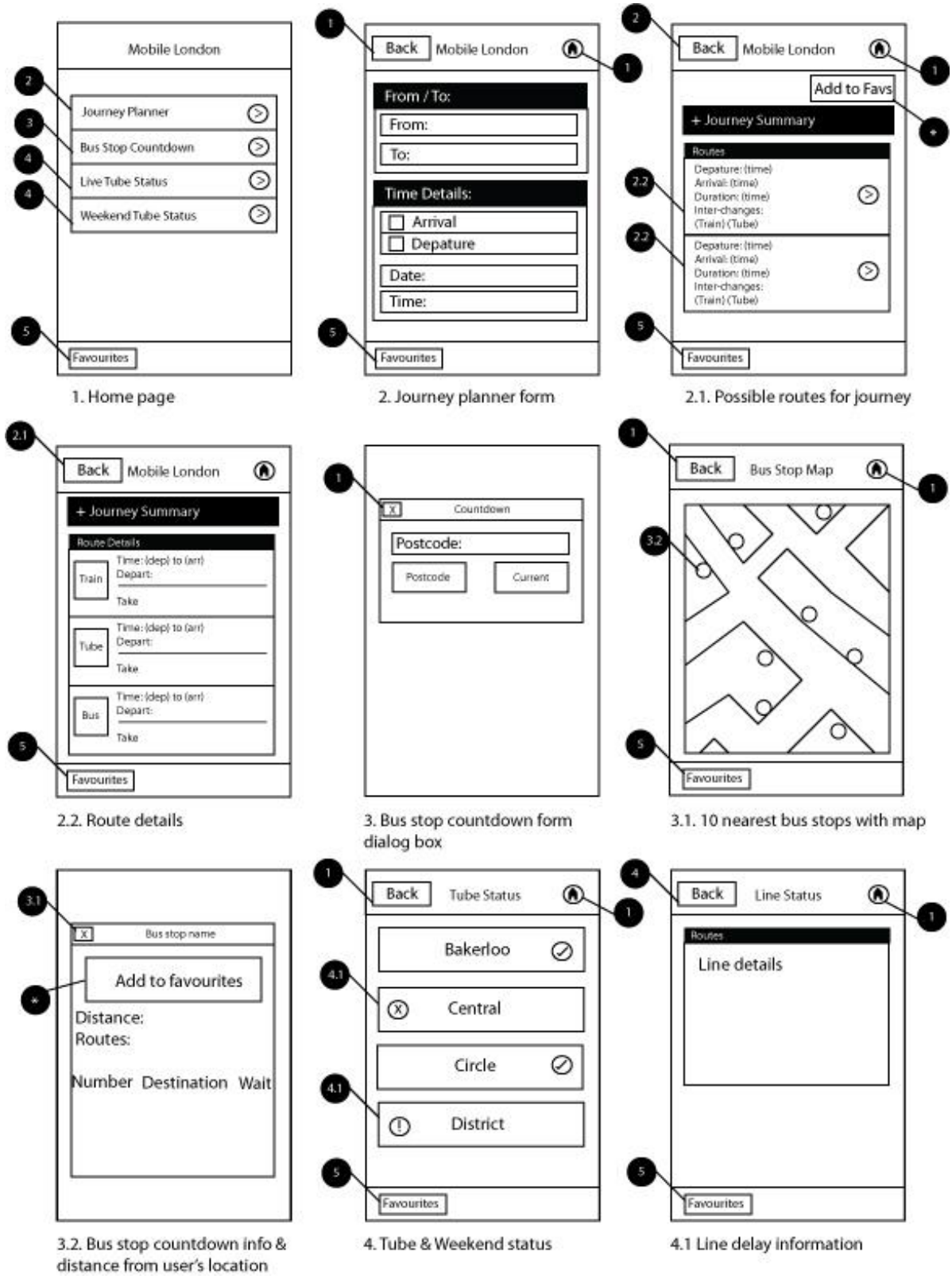


Figure 17 - Wireframes Part 1

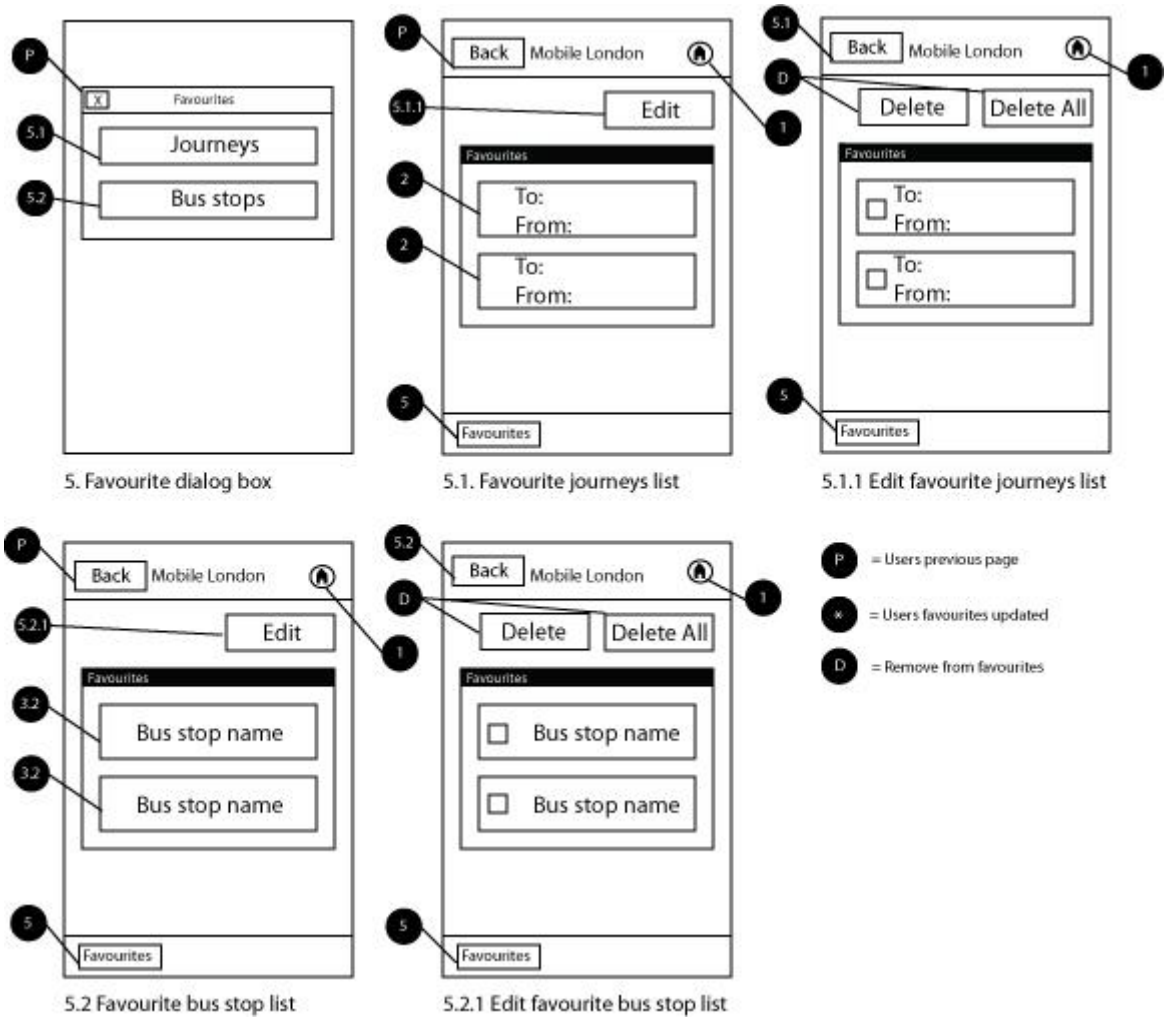


Figure 18 - Wireframes Part 2

3.6 Summary

With the use of the prototyping methodology, this section has outlined the concept of the mobile web application; whether it is how the artefact is used and the layout of the artefact, what it provides to users or what the artefact is required to do.

Some sections of this design stage have been completed more than once, such as the wireframe stage has been tested as a prototype by users to ensure that the intended users can use the system well. The proposed solution was finalised in this stage before the start of the implementation. This section has stated that the journey planner functionality is the most important as well as the technologies that were used. NodeJS was used for the server technology as well as the jQuery Mobile framework for the front-end.

4 System Implementation

This chapter gives an insight into how the project application was implemented, with any problems that occurred during development, testing and evaluation.

4.1 Introduction

The application was created using NodeJS server technology, using open source modules available with NPM (these modules are listed in 4.2.1 Node Modules). The mobile framework jQuery Mobile was used to deliver the data to the user. All code used, that was created by other sources are open source, meaning no copyright laws have been breached.

4.2 Open source code

For the build of this artefact some open source code was used for both server and client side. Sections 4.2.1 Node Modules lists the Node modules used, all of which are open source and 4.2.2 Client-side code lists all of the client-side code.

4.2.1 Node Modules

A list of open source NodeJS modules used to build the artefacts:

- express (Holowaychuk, Express)
- express-resource (Holowaychuk, Express-Resource)
- dateformat (Geisendörfer)
- GeoDistance (Witt)
- xml-stream (XML-Stream)

4.2.2 Client-side code

For the creating of this artefact some open source client-side code was used:

- jQuery (Resig, 2011)
- jQuery Mobile (Parker, 2012)
- jQuery Mobile Datebox (Sage)
- Google Maps API v3 (Google Maps JavaScript API v3)
- Google Maps API Web Services (Google Maps API Web Service, 2011)
- jQuery UI Map (Larsson)
- jQuery Mobile ThemeRoller (ThemeRoller for jQuery Mobile)

4.2.3 Data APIs used

Most of the information displayed to users is provided by APIs from TFL:

- Journey Planner API Beta (Guidelines and Support)
- Tube this weekend (Guidelines and Support)
- Tube departure boards, line status and station status (Guidelines and Support)
- Bus stop countdown (Live Bus Departures)

4.3 Interactions with other services

The majority of the functionality built for this artefact relies on other services. The aim of this section is to list all of these interactions along with workflows of how these services are used.

1. Journey Planner
2. Bus stop locations
3. Bus stop countdown information
4. Bus stop postcode search
5. Bus stop map
6. Weekend tube status
7. Live tube status

4.3.1 Journey Planner

Using the journey planner functionality, the NodeJS server interacts with TFL's journey planner API to send and retrieve route information for the user's selection.

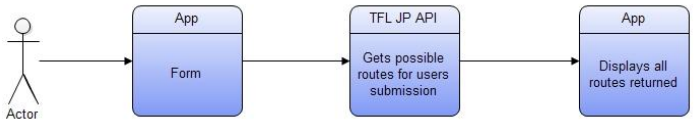


Figure 19 - Journey Planner System

4.3.2 Bus stop

The bus stop functionality uses potentially four different services. Google Web Services gathers co-ordinates based on the postcode entered by the user. This then passes the co-ordinates to TFL bus stops to retrieve 50 local bus stops. The server then orders this result by distance of the user to retrieve 10 local bus stops and using the Google Maps API displays these 10 to the user in a visual bird's-eye view. The last service used with this, retrieves the countdown information from TFL for the bus stop a user selects; this is run using AJAX to ensure information is the most up-to-date.

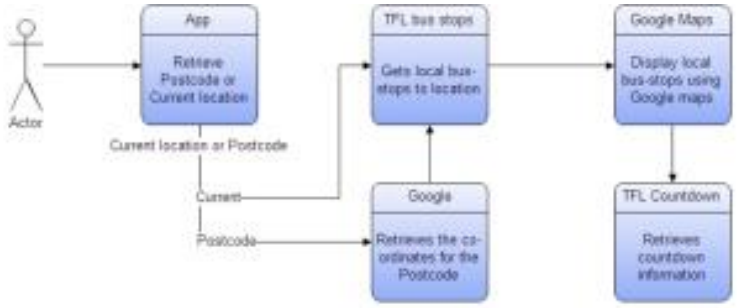


Figure 20 - Bus stop countdown system

4.3.3 Live and weekend tube status

The live and weekend tube status functionality retrieves information directly from an API provided by TFL about all TFL lines.

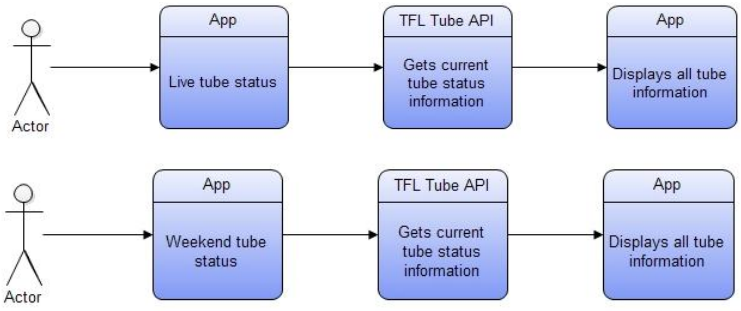
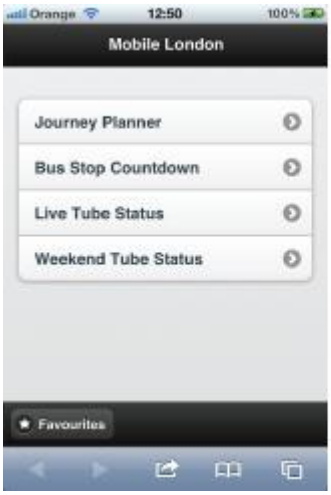


Figure 21 - Tube live and weekend status

4.4 Final build

This section shows the completed artefact built from the wireframes in Figure 17 and Figure 18 running on an iPhone 4's device running the OS iOS5 (see Figure 22, Figure 23 and Figure 24). Some further functionality was added to the final build compared to the wireframes; this was to help enhance the user experience.

Firstly, as seen in Figure 22 - Screenshots Part 1, the Journey planner form had a date box popup and a time box popup added. This further functionality has been used to help provide an easier user experience when changing the time and date from the current. The last enhancement added, as seen in Figure 24 - Screenshots Part 3, displaying of the user's favourite bus stop information has been changed to have its own page and to provide a bird's-eye view of the bus stop location (using a static image).

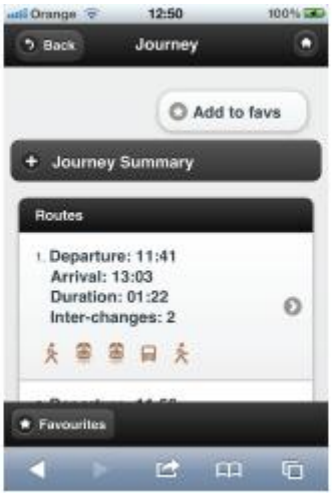


Home Screen

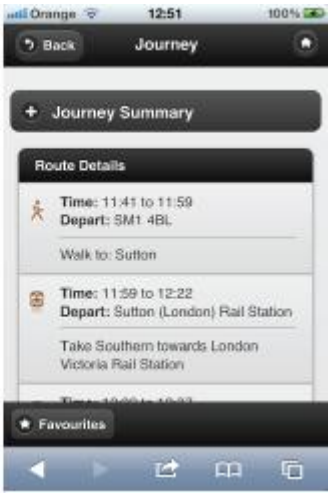
Planner Geo-location

Planner date box

Planner time box



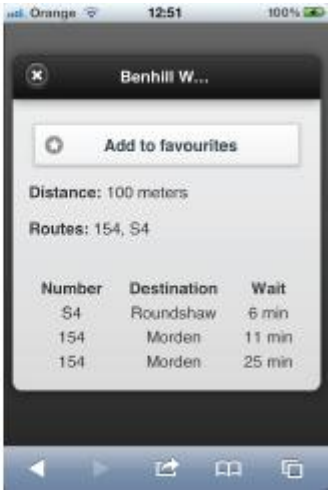
Route options



Route details

Bus stop form

Bus stop map



Bus stop countdown

Figure 22 - Screenshots Part 1

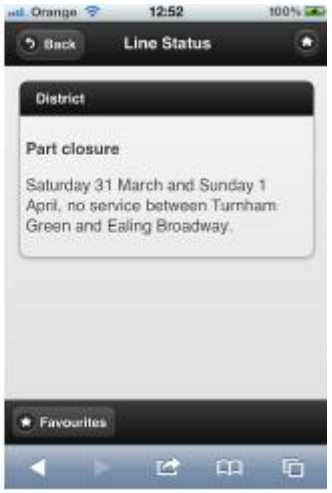
Bus stop filter system

Bus stops filtered

Tube status

Tube status options

Weekend tube status



Line status information

Favourites page

Favourite journeys

Edit favourites

Figure 23 - Screenshots Part 2

Favourite journey form | Favourite bus stop | Favourite stop map

Figure 24 - Screenshots Part 3

4.5 Interesting Problems

Whilst attempting to meet all of the objectives, some problems occurred during the build of the artefact. This section looks at some of these problems and how they have been addressed during development.

1. Bus stop countdown API
2. Postcode search
3. Journey Planner API
4. NodeJS

4.5.1 Bus stop countdown

The main problem, which occurred during the build, was that it was initially believed that the bus stop countdown API would be released by TFL. However, this did not happen which meant a different approach was needed.

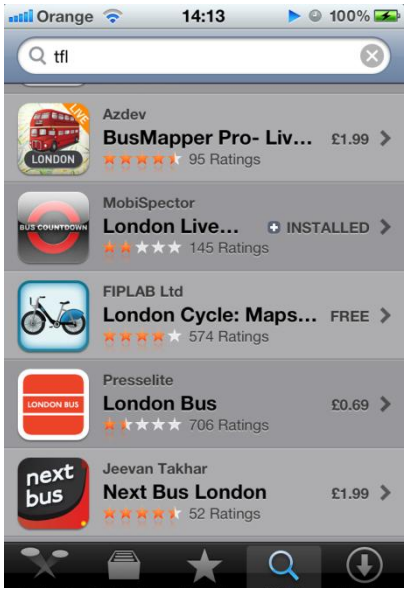


Figure 25 - Bus stops Apps Apple Store

The first stage was to ensure that providing this information was viable. By looking at some of the current iOS apps available on the Apple app store, proved this was possible (as seen in Figure 25).

This proved that it is possible to produce third party applications, which gather bus stop countdown information. Further research was needed, the results of which showed others have looked at this issue and Arian Short on Github (Short) had found the JSON

link used by TFL, which is an unofficial TFL API (the link directly to this JSON file is http://countdown.tfl.gov.uk/stopBoard/75078). Due to the nature of retrieving this information, further research was used to ensure that there would be no breach of copyright laws from TFL. As seem in the appendix (7.3 Transport for London disclaimer) no copyright laws are breached.

This proved that the concept behind the functionality was completely feasible for this project. However, the next issue that still occurred was the finding of local bus stops in response to the user's submission (of either postcode or current location). This was less of a problem as TFL do provide an official Bus stop locations API. However, this in itself causes a problem as the data that is returned by TFL is in a CSV format, which is not read natively with NodeJS.

Two possible ideas came from this; to reproduce the CSV file as a JSON file or to input the data from the CSV file to a MySQL database. With both feasible, the number of bus stops currently being 19,562 this requires NodeJS to go through the JSON file or MySQL database and retrieving 10 of the closest bus stops to the user's current location (or postcode). With some further research a blog post from Robin Osborne (Osborne, 2011), showed detailed information of how to retrieve 50 local bus stops using TFL's website (Figure 26 - Bus stop search Fiddler and Figure 27 - Bus stop countdown JSON Code) proves this concept will work. To do this it is required to change the current location retrieved from HTML5, which is longitude and latitude to north east longitude, north east latitude, south west longitude and south west latitude.

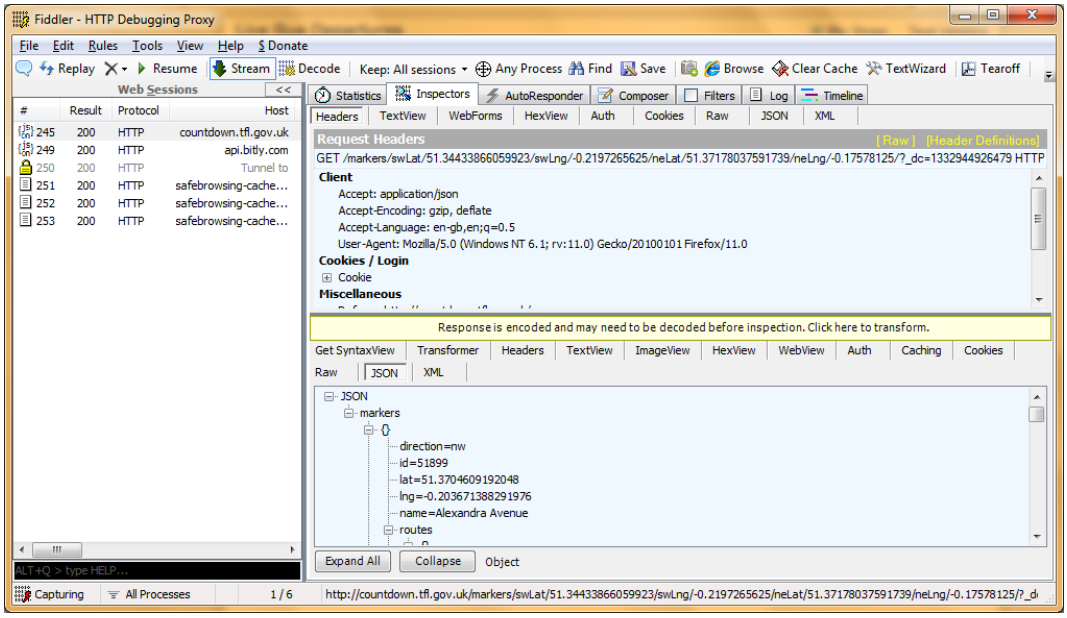


Figure 26 - Bus stop search Fiddler

```
{
    "lastUpdated":"14:59"
    ,   "filterOut":[]
    ,   "arrivals":[
        {   "routeId":"154"
            ,   "routeName":"154"
            ,   "destination":"West Croydon"
            ,   "estimatedWait":"8 min"
            ,   "scheduledTime":"14:07"
            ,   "isRealTime":true
            ,   "isCancelled":false
        }]
        ,   "serviceDisruptions":
            {
                "infoMessages":[]
                ,   "importantMessages":[]
                ,   "criticalMessages":[]
            }
}
```


Figure 27 - Bus stop countdown JSON Code

There is a potential future problem using both of these unofficial APIs. If TFL changes its own system, it will affect the artefact because it will no longer be able to provide bus stop countdown information. The risk of this will have to be taken until an official API is provided by TFL in which maintenance will be required straight away, to reduce future risk and keep the application working.

4.5.2 Postcode search

Once the bus stop countdown issue had been dealt with another issue arose with this functionality. One of the aims with this was to allow both current location and postcode to be used by the user to find the bus stop locations.


```
▼<ResultSet version="1.0">
  <Error>0</Error>
  <ErrorMessage>No error</ErrorMessage>
  <Locale>us_US</Locale>
  <Quality>60</Quality>
  <Found>1</Found>
 ▼<Result>
    <quality>60</quality>
    <latitude>51.478741</latitude>
    <longitude>-0.214100</longitude>
    <offsetlat>51.478409</offsetlat>
    <offsetlon>-0.213920</offsetlon>
    <radius>100</radius>
    <name/>
    <line1/>
    <line2>London</line2>
    <line3>SW6 6TU</line3>
    <line4>United Kingdom</line4>
    <house/>
    <street/>
    <xstreet/>
    <unittype/>
    <unit/>
    <postal>SW6 6TU</postal>
    <neighborhood/>
    <city>London</city>
    <county>Greater London</county>
    <state>England</state>
    <country>United Kingdom</country>
    <countrycode>GB</countrycode>
    <statecode>ENG</statecode>
```


Figure 29 - Yahoo places results

Figure 28 - Yahoo test

An issue with this was to allow the postcode search. The reason for this is the Post Office currently owns the database of co-ordinates to postcodes and sells this at a high cost. In order to keep costs down this was not seen as a possible option and therefore further research was required on this matter.

During research it was possible to see that some work has been carried out on collecting this data from individuals, to create a database of co-ordinates for the postcodes (for example http://www.freethepostcode.org). This is still not a viable option as it is not possible to ensure that all of the data is accurate. The next stage was to look at free APIs provided; two in particular were looked at, Yahoo's Place Finder, and the Google Maps API Web Services. As seen in Figure 28 (which uses the results from Figure 29) point A is the correct point for the postcode, whereas Yahoo finds point B. Google Maps API was therefore used, which under the terms and conditions have to be used with Google Maps, which the application does by displaying the co-ordinates with the bus stop locations.

4.5.3 Journey Planner

Two issues occurred when coding the Journey Planner functionality. These were the user interaction for selecting their route, with the form and reading the XML provided back from the TFL API. After the wireframe stage of testing it was deemed that the form enabling the user to plan their journey was not as intuitive as it could be.

A jQuery Mobile module called Datebox (Geisendörfer) was used to help the user with a calendar style popup and time popup for these input selections, providing a more visual approach to the form.

This issue was only a minor one that was easily dealt with. However, the main issue for the build of this facility of the application was the way the XML is returned by TFL. This issue was not identified straight away due to initial testing only using the postcode feature.

```
<itdRouteList>
▼<itdRoute active="1" delete="0" changes="2" distance="123
 routeIndex="1" hasFrequency="1" routeTripIndex="1" cTime=
  ▼<itdDateTime>
     <itdDate day="28" month="3" year="2012" weekday="4"/>
     <itdTime hour="19" minute="6"/>
   </itdDateTime>
  ▼<itdDateTime>
     <itdDate day="28" month="3" year="2012" weekday="4"/>
     <itdTime hour="20" minute="33"/>
   </itdDateTime>
```

Figure 30 - Journey Planner Postcode XML

Different XML is returned depending on the type of search. As seen in Figure 30 two <itdDateTime> tags are returned. The first is the departure of the route and the second is the arrival of the route. However, as seen in Figure 31 these are no longer returned.

```
<itdRouteList>
▼<itdRoute active="1" delete="0" changes="0" distance="-1" alternative
 cTime="20120328193925363" searchMode="1" vehicleTime="6" method="itp"
  ▼<itdMapItemList>
    ▼<itdMapItem text="" type="RM">
       <itdImage src="FILELOAD?Filename=TLJXML04P1__4F735ADD2.pdf"/>
     </itdMapItem>
   </itdMapItemList>
   <itdDaysOfService/>
  ▶<itdPartialRouteList>...</itdPartialRouteList>
 </itdRoute>
```

Figure 31 - Journey Planner Station XML

This caused an issue as it meant the server believed that there was no route available for the user's search. To combat this, an array was created (as seen in Figure 32 and Figure 33) to store the departure time at the start of the code and the last arrival time for all of the sub-routes was collected.

```
// Test to ensure depature and arrival times have been passed
var depHr = item.itdPartialRouteList.itdPartialRoute[0].itdPoint[0].itdDateTime[0].itdTime[0].$.hour;
var depMin = timeCheck(item.itdPartialRouteList.itdPartialRoute[0].itdPoint[0].itdDateTime[0].itdTime[0].$.minute);
routeDepature[i] = depHr + ":" + depMin;
```

Figure 32 - NodeJS Departure code

```
// Create Arrival information for route
var arrHr = data.itdPoint[1].itdDateTime[0].itdTime[0].$.hour
var arrMin = timeCheck(data.itdPoint[1].itdDateTime[0].itdTime[0].$.minute);
routeArrival[i] = arrHr + ":" + arrMin;
```

Figure 33 - NodeJS Arrival code

4.5.4 NodeJS

The main challenge of this project was to learn how to create and run the server using NodeJS with the JavaScript language. The general learning of coding and syntax of the language was not a problem due to previous experience with the language. However, the build, layout and execution of the code were a completely new concept compared to previously used server languages such as PHP.

```
var http = require('http');
pCLatLng = '';

function postCodeCheck() {
    var pCode = {
        host: 'geo.jamiethompson.co.uk',
        path: "/" + 'SW1A2AA' + ".json"
    };

    http.request(pCode).on('response', function(response) {
        var data = '';
        response.on("data", function (chunk) {
            data += chunk;
        });
        response.on('end', function () {
            pCJSON = JSON.parse(data);
            pCLatLng = pCJSON;
        });
    }).end();
    console.log(pCLatLng);
}

postCodeCheck();
```

Figure 34 - NodeJS example code PHP style

The initial problem was grasping the fact of the asynchronous model, whilst attempting to write the code with the same concept as may be used with PHP.

```
var http = require('http');
function postCodeCheck(callback) {
    var pCode = {
        host: 'geo.jamiethompson.co.uk',
        path: '/SW1A2AA.json'
    };

    http.request(pCode).on('response', function(response) {
        var data = '';
        response.on('data', function (chunk) {
            data += chunk;
        });
        response.on('end', function () {
            callback(JSON.parse(data));
        });
    }).end();
}
postCodeCheck(function (pCLatLng) {
    console.log(pCLatLng);
});
```

Figure 35 - NodeJS example code non-blocking

Figure 34 shows an example of NodeJS code, which was written during the build that returns the Geo-location for a postcode. The problem with the way that this code has been written is that it expects code to be executed in order (otherwise known as 'blocking'). However, due to the NodeJS server technology, it is not possible to use this concept and instead a non-blocking method needs to be used (as seen in Figure 35).

This example is only on a small scale. However, with some code or larger code, this can cause great differences such as the information returned by the request; the code would need to wait on the callback to be completed before the next part of the code to be run.

As the only previous experience prior to this project was with PHP, it was difficult to grasp this new concept building the artefact. This caused the artefact build to take longer than initially thought.

4.6 Summary

In summary, this section has outlined all open-source code whether it is server or client side code, as well as all services (or APIs) such as TFL's APIs, to provide the information and functionality of this

application, which is needed due to the nature of the project. From the wireframes it can also be seen that the implementation of this application meets the intended design.

4.7 Testing

Different steps have been taken during the project to test a variety of aspects. Firstly, a prototype was created to test before the implementation stage; the wireframes were given to potential users of the application to ensure that the user interface was at a satisfactory standard.

4.7.1 Prototype Testing

This stage used wireframes (as seen in 7.1 Initial Wireframes). This was tested in order to ensure that the layout and the information displayed were, as the user would expect. During the test users were asked to find local bus stops depending on their current location, as well as planning a journey.

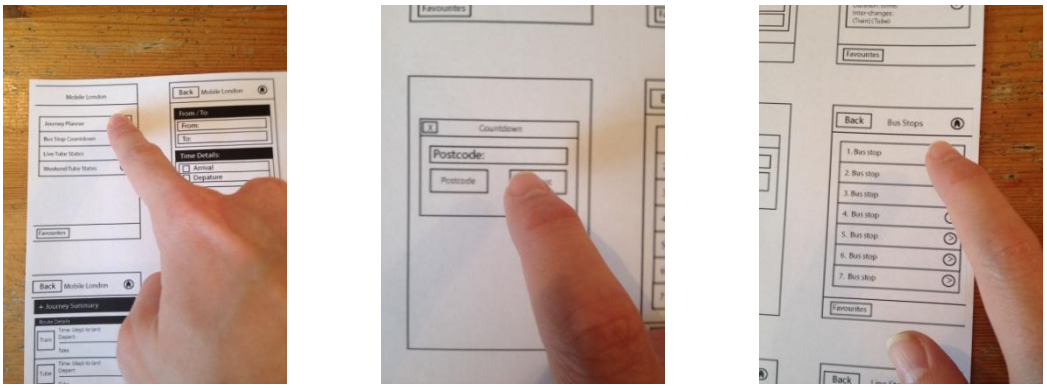


Figure 36 – Wireframe tests

Figure 36 illustrates some of the user tests performed during this stage. Once testing was completed participants were asked whether the navigation through the system met their requirements and as they expected. The majority of participants verified this.

One issue raised was the listing of bus stops. This wasn't a navigational problem as much as an information problem. The participant raised an issue with not always knowing which bus stop that he goes to by name and some bus stops being named the same, due to being located in the same road. This was deemed the biggest issue from the prototype testing, as one of the requirements was to provide bus stop countdown information; which is not possible if the user does not know which bus stop they are initially going to. In order to combat this it was already known that there was a module created allowing Google Maps V3 API on jQuery Mobile. It was decided that this would be the best viable option to fix the problem, providing users with a bird's-eye view and a more visual approach. This change can be seen in Figure 17 wireframes and Figure 22 in the final stage build.

4.7.2 Functionality testing

In order to ascertain that all the functionality works, testing throughout the build of the artefact was important. In order to do this from Portsmouth it was important that testing against available current systems was used.

Some other programs were therefore required; an add-on 'Geolocater' was used with Firefox to test the use of current location. This add-on simply changes the browser's co-ordinates which means that using this facility should not affect the testing's integrity (Figure 37 shows this add-on in use).

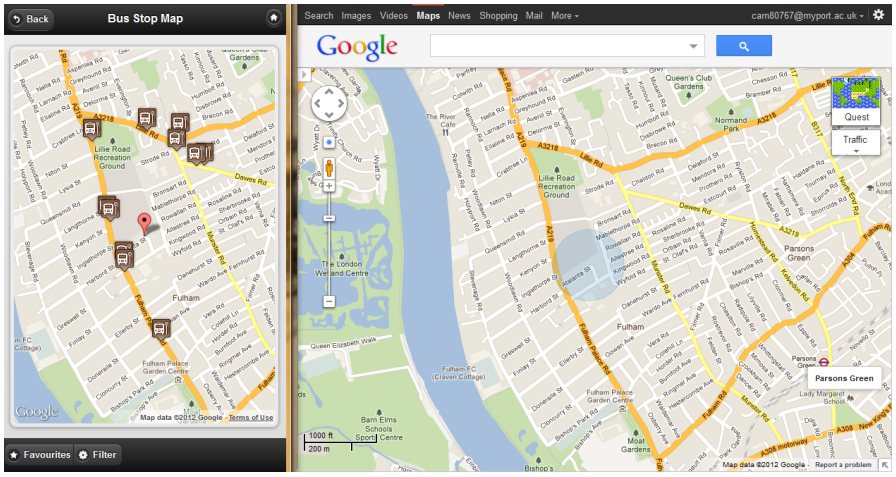


Figure 37 - Current location add-on

4.7.2.1 Journey planner

A series of tests was needed to ensure the accuracy of the journey planner:

1. Postcode
2. Current location
3. Station

Firstly, the postcode test needed to check that the user has entered a valid UK postcode, and then return the journey information.

Test Type	Data Used	Expected outcome	Outcome
Normal	SW6 6TU	Accept	Accepted
Normal	SW1A 2AA	Accept	Accepted
Erroneous	1SA 2AA	Error message	Error message
Erroneous	SW6 6TU1	Error message	Error message

Table 5 - Journey planner postcode test

The error message for the '1SA 2AA' test can be seen in Figure 38. It was important to check for errors such as valid UK postcodes on the client-side. This helps speed the client's interaction with the form and stops the client continuously submitting data that will cause errors costing the user's data allowance.

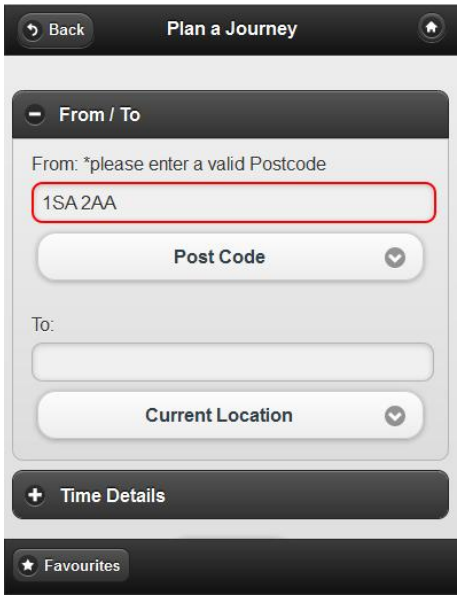


Figure 38 - Journey planner postcode error

The next stage of testing this functionality was to ensure that the correct data was returned, whether it was the user's current location, by postcode or by station name. In order to test, the same location was used by entering the current location as Sutton station, which was in return used for postcode and station going to Morden.

The data of this test should generally be the same. However, it was unknown whether the information returned would be 100% identical due to the positioning provided by TFL.

Figure 39 shows from left to right a search from current location (Sutton station co-ordinates), postcode (SM1 1DE) and Sutton station. The data retrieved from TFL is not 100% the same. This however, does not mean that the information is incorrect. Due to the area that a postcode can cover and the degree of accuracy of the co-ordinates, it should be expected that the information would differ slightly. Some of the information is still the same, such as the departure at 15:15 is the same route as the departure at 15:17 from the station, as walking distance is not needed for the station as the user should already be considered as being at the station.

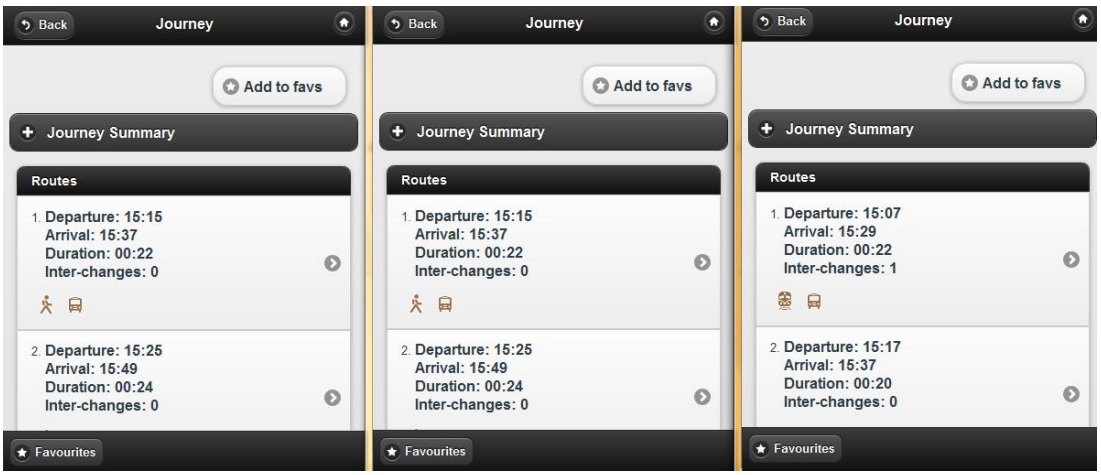


Figure 39 - Journey planner test

The next test for this was to ensure that the information being retrieved was valid. To do this a journey from SM1 1DE to Morden was searched using the artefact compared to TFL's mobile version. The data retrieved should display the same routes, departure/arrival times etc. which can be seen in Figure 40.

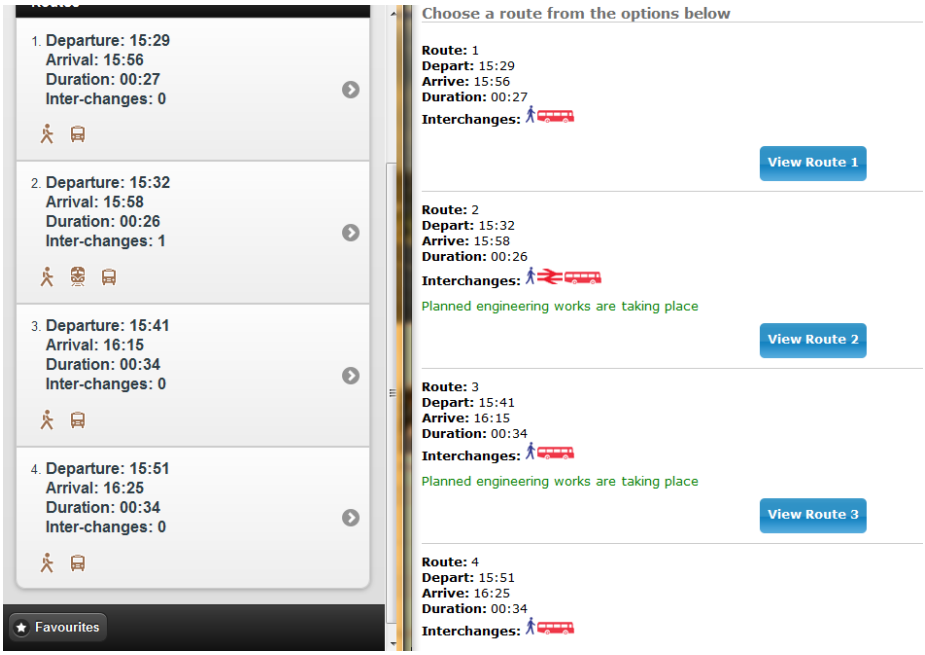


Figure 40 - Journey details test

4.7.2.2 Bus stop countdown tests

Due to the information coming from a non-official API, however still provided by TFL the information should be correct. To test this functionality the bus stop countdown information was used against TFL's provided desktop system.

Firstly, it is important to ensure that the correct bus stops are displayed to the user; Figure 41 shows this test. The test uses the artefact's bus stop countdown functionality compared to TFL's mobile version.

In this test it is possible to see that the same bus stops are returned to the user. However, more bus stops are returned to the user compared to TFL's static image; with the possibility of interaction with the map by clicking on points, moving and zooming in or out on the map. This test proves that the correct bus stops and locations are displayed to the user.

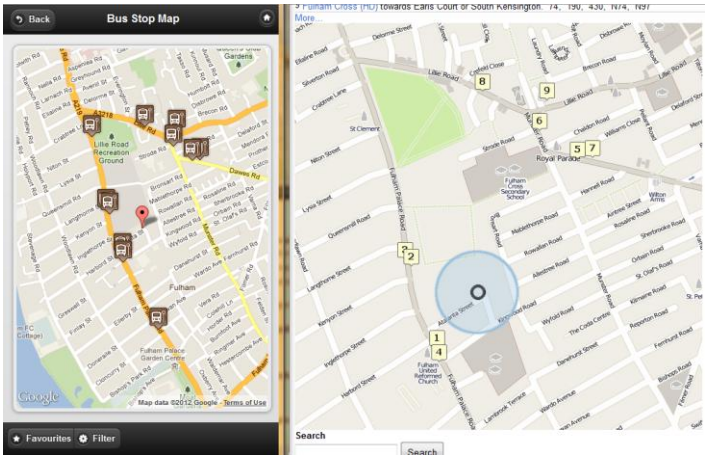


Figure 41 - Bus stop location test

The next step of testing this functionality was to ensure that the correct bus stop countdown information is collected and provided to the user. Figure 42 looks at this information provided from TFL's mobile site, which corresponds with the information, provided on the countdown for this bus stop countdown.

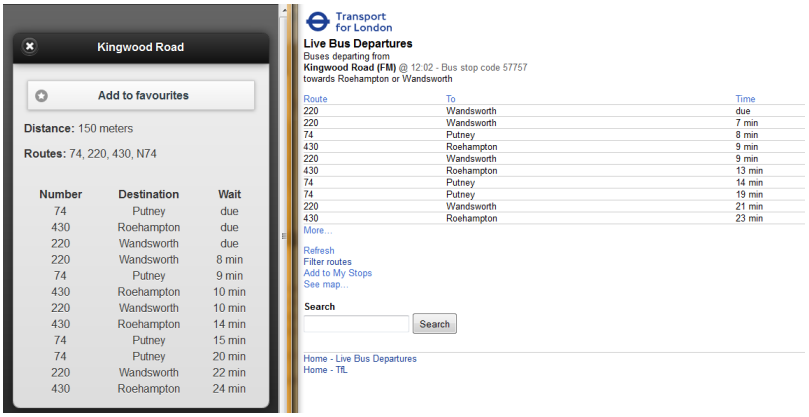


Figure 42 - Bus stop countdown information

4.7.2.3 Weekend tube status

Testing the weekend tube status was needed to ensure that the correct data is provided for lines (by showing delays etc.) and line information. Testing was carried out, to compare the TFL's website with the artefact, ensuring that the data corresponds with each other as well as showing more information on any delays.

As seen in both Figure 43 and Figure 44 the information mutually corresponds with each other. However, the way that information is displayed differs between both applications. TFL's application uses colour coding of lines, whereas the project application being built uses a colour scheme for displaying delays on the line (Green – Good, Amber – Delays / Part closures and Red – Closure).

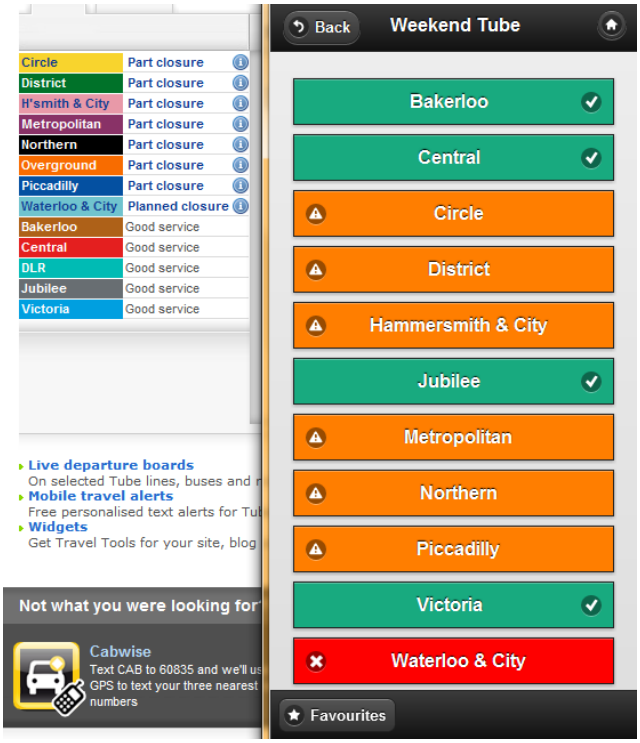


Figure 43 - Weekend tube status lines test

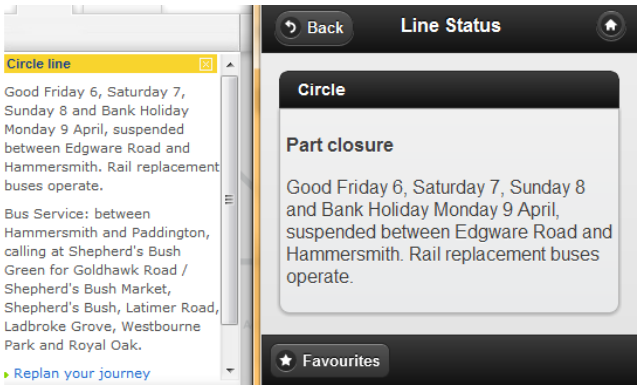


Figure 44 - Weekend tube info test

4.7.2.4 Live tube status

In order to test this functionality, the same test was used as in 4.7.2.3 (Weekend tube status). The only thing that differs is the information being displayed.

It can be seen in both Figure 45 and Figure 46 that the information corresponds correctly again. As in the case of the weekend tube status functionality the way that the information is displayed is slightly different. However, keeping both the way that live and weekend tube status is displayed with the same interactions for the user, keeps the functionality of this application the same across the site allowing users to be able to pre-empt what will be displayed.

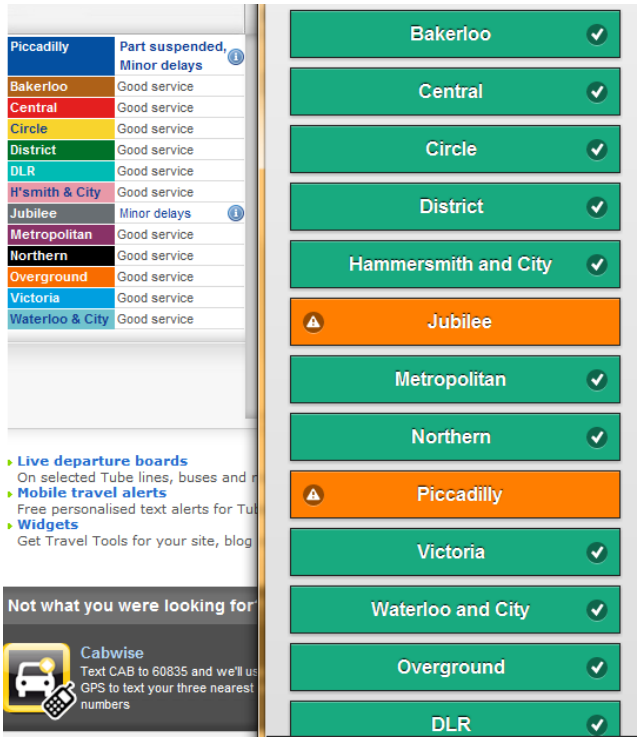


Figure 45 - Live tube status lines test

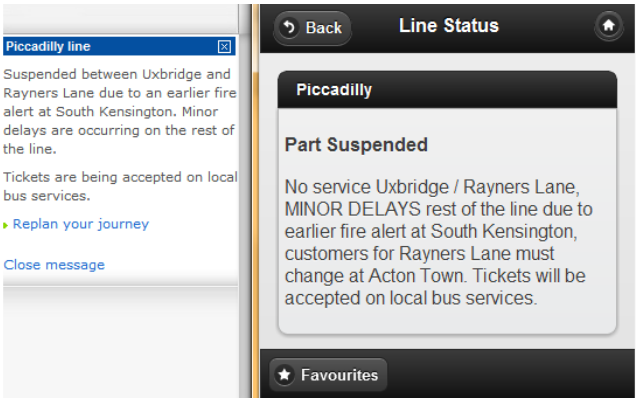


Figure 46 - Live tube info test

4.7.2.5 Favourites

This stage of the testing incorporates two different functionalities into one (both the bus stop countdown information and the journey planner). Initially the test needed to make sure that the user could add favourites for both and delete current favourites.

4.7.2.5.1 Adding favourites

This step tested that users can add favourites for both journeys and bus stops. Looking at Figure 47 it is possible to see that favourites can be added correctly providing an option to edit their current favourites when on their favourites screen.

It should be noted that if the user tries to access the favourites page with a device that does not support HTML5 local storage, they will be provided with an error page as all favourites are stored on the user's device rather than in a database (such as MySQL).

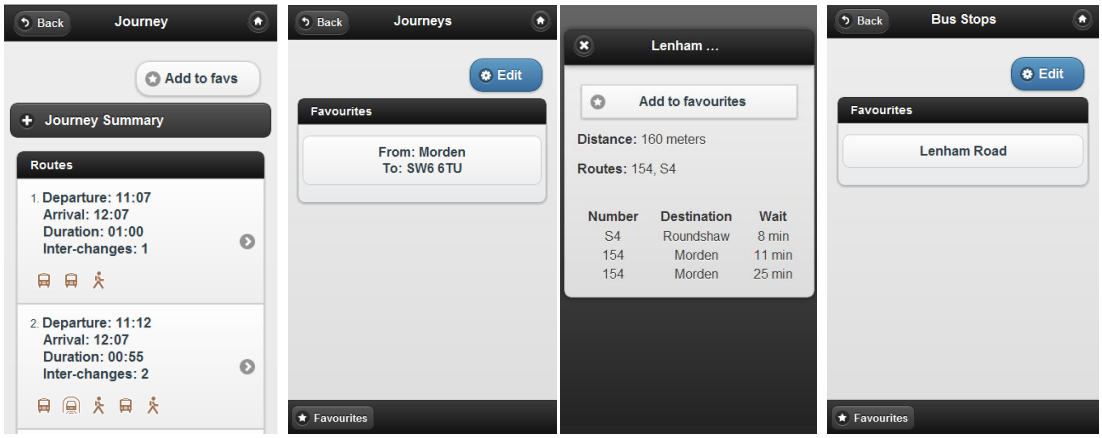


Figure 47 - Journey and Bus favourites added test

4.7.2.5.2 Removing favourites

Having favourites is important for this application with the interaction by the user. However, it is important to ensure that users have the opportunity to remove current favourites to ensure that the list is up-to-date with what the user actually wants and need.

During initial tests it was found that deleting from favourite journeys also cleared bus stop favourites. This is not how the system should work; a simple fix was needed for this. The reason for this bug was the use of localStorage.clear. This was then changed to nullify the stored variable with localStorage.setItem.

Looking at Figure 48 it can be seen that since bug fixing this issue, editing both the bus stop and journey favourites now work as intended, removing only ones that the user selects.

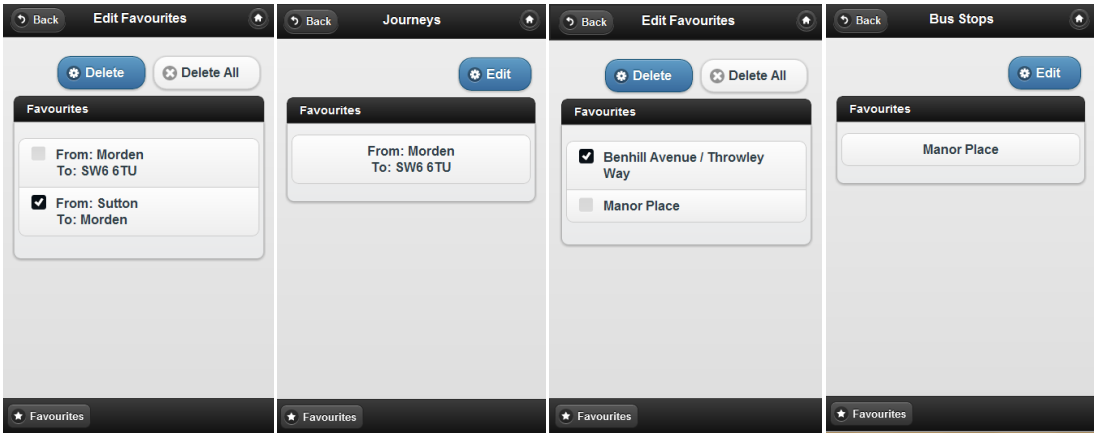


Figure 48 - Journey and Bus favourites delete test

4.7.2.5.3 Using the journey favourites

It is all well and good allowing users to store favourites. However, this functionality is not needed unless the user can use this as a search facility. This section aims to test that the use of journey favourites can be used to return route information.

As seen in Figure 49 the journey favourite search works as expected, allowing users to set time and date details as with the origin and destination pre-set. This then returns the information the same as the normal journey planner option.

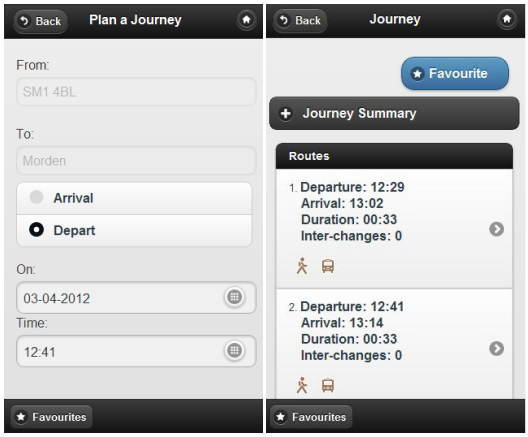


Figure 49 - Favourite journey search

Using the bus stop favourites as seen in Figure 50, displays the information slightly differently compared to the normal bus stop countdown option. This information has been changed as using the map facility is not needed, as the user already knows which bus stop they require.

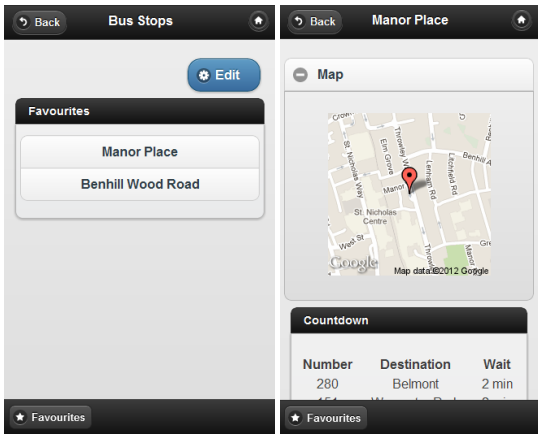


Figure 50 - Favourite bus stop countdown

4.8 Evaluation

The aim of this section was to look at the artefact as it has been produced compared to the requirements, aims and objectives of the project.

4.8.1 Requirements Review

In order to evaluate this artefact it was important to check the finished product compared to the requirements set before build.

	Requirement met
Journey planner	✓
Bus stop countdown	✓
Live tube status	✓
Weekend tube status	✓
Favourites	✓
Cross-platform	½
Simple design	✓
Data size	X
Server speed	X

Table 6 - Requirement review

As seen in Table 6 most of the requirements initially laid out in 3.2 Requirements have been met. All the individual functionality requirements have been met in the build; with the journey planner, bus stop countdown, favourites, live and weekend tube status all provided within the same application. The simple design requirement, which is a usability requirement that was set out so users would have an easier interaction with the application, has been reached. To meet this requirement, testing with prototypes was carried out in 4.7.1 Prototype Testing to ascertain whether users would have any issues when using the site. Only one issue was raised in this stage about the bus stops and how they were listed to the user.

A high-ranking requirement for this application to work cross-platform has not been fully met. The mobile site is accessible on all Smartphone platforms that were outlined during the literature review for market share. The site has not been fully tested on any mobile device other than an iPhone (running iOS), meaning that it is not fully optimised for the client's use; this was due to a lack of funding for different devices for the testing stage.

During the build of this project, new technologies have been learnt which therefore means some code may not be fully optimised. This requirement has not been fully completed and should be looked at with further development and maintenance.

A personal machine was used for the development of this application. This however, was not a viable option due to internet and processing speeds. On completion, the application was moved to Amazon's EC2 micro 12-month free tier service. Due to budget constraints this server has been used at its current stage. However, once going live and providing full service this will need to be moved to a paid server with more stable speeds. In order to deal with peak times it is proposed that further

development should be completed so that the application can be run on multiple instances using Amazon's auto scaling facilities.

4.8.2 Project conference feedback

The application was demonstrated at a student research conference at the University of Portsmouth on the 14th March 2012. In order to present this application a laptop running Firefox using the same 'Geolocater' add-on from the testing stage was used.

During the demonstrations many people were intrigued to use their current location to plan a journey, which most had not seen before for a TFL journey application. The other main functionality that users were interested in was the bus stop information, as it meant they could know when a bus was due without needing to wait.

Further communication was made by email with Stuart Reece from IBM. This was to try and get a more in-depth response from potential users. One point that was relayed back was the look "For the application overall, I think the interface is simple, nicely laid out and easy to use. I also like the way it will save your previous selections, such as Current Location versus Station in the Journey Planner." (S. Reece, personal communication, March 16, 2012). This was a major factor for the design stage to ensure that the application had a simple interface.

Another point of Stuart's was that "the Journey Planner, I like the way that Time Details are hidden by default, to keep the interface simple." (S. Reece, personal communication, March 16, 2012). Going back to the literature review, it is important to ensure that the page layout meets the needs of the users, which has been successfully achieved as most of the time users will be using the system to undertake a Journey from now.

4.8.3 Artefact Review

On completion of the artefact, the overall target of the application was achieved. Due to some problems, which occurred along the way, such as the learning of NodeJS, unavailable APIs and funding caused the site not to be fully optimised for all OS'. However, they do still run and work. The main aims and objectives created at the start of the project have been met.

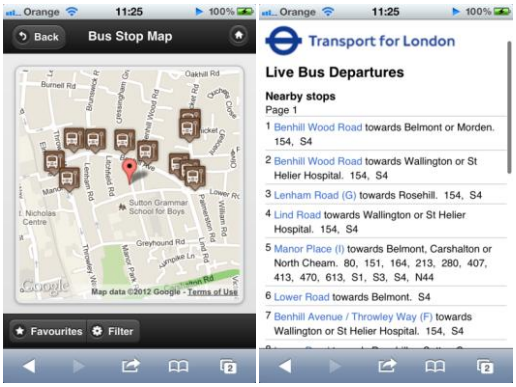


Figure 51 - Mobile London Bus and TFL Bus

The artefact created provides a simple interface to users whilst remaining aesthetically pleasing with user interaction when necessary. Looking at Figure 51 it can be seen that the artefact provides information in a more visual way to the user, which will helps the user when using the mobile site.

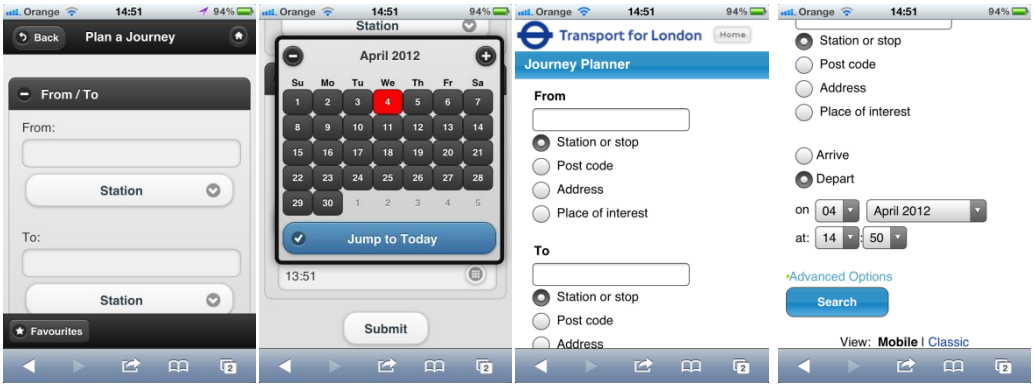


Figure 52 - Mobile London and TFL Journey Planner

As seen in Figure 52 it is not possible to plan a journey based on a user's current location when using the TFL provided option, compared to the Mobile London version. This will help users plan journeys whilst on the move as they may not necessarily know their current location or what is near them. A simpler user interface has been used for date, providing a calendar look. One point of this was that some users may know what day they need to leave (such as the following Thursday) although, they may not know the date of that particular day.

4.9 Testing and Evaluation Summary

In this section, it can be seen that the final product has met the aims of the project, how the information is provided, the functionality and what information users have access to within the same application. The design of the artefact was of a high importance for this project to ensure that it was simple for users to use while being aesthetically pleasing, which in the testing sections it is possible to see. Within the evaluation stage it can be seen that the requirements have been met as far as possible due to time constraints.

5 Conclusion

The overall aim of this project was to provide TFL's customers with a mobile web application accessible on multiple devices, with a simple design layout and up-to-date live information. These aims were then split up into objectives with individual functionality options for users.

The research section (2 Literature Review) was used to look at how the data should be displayed to the user, looking at W3C web standards and the possible coding languages that could be used. The problem identified from the research is that not 100% of all Smartphones will necessarily be able to access the site if HTML5 and JavaScript are used, as older phones will not have this capability. However, the number of possible users outweighs this percentage.

The result of the research for this project outlined the possible frameworks for the front-end of the artefact and multiple server-side technologies. However, two were mainly focused on, PHP due to prior knowledge and NodeJS, a JavaScript language technology as a new upcoming server concept. The outcome of this research was to build a mobile web application available for multiple devices using NodeJS and jQuery Mobile. The reason for this was that the use of JavaScript both client and server side seemed a natural fit for any variables that may need to be parsed between the two.

A result that has come from completing this project has been the use of the technologies and recommendations for their use as 'real-world' applications. NodeJS as a server technology is new. However, the community surrounding the technology is vastly growing with modules constantly being built and maintained. Using the server technology prevents developers from dealing with concurrency. NodeJS makes it simpler for developers to create applications where multiple computations are executed simultaneously. As the language is JavaScript, which can be used both client and server side, it means that less code should need to be written with the possibility of using the same code, as well as parsing variables between the two.

During the research the difference between native and web applications was looked at. The result of this is that organisations can easily create web applications using a mobile framework (such as jQuery Mobile), which can be mixed with another mobile framework PhoneGap to be used as a native application. This provides a far greater opportunity for organisations to create an application as it means the same code can be used across multiple devices, saving time and therefore money.

Since finishing the build of the application, some further functionality has been thought of to enhance the system and provide more information to the users. A list and details of some of these enhancements can be found below.

- London charge points
- PhoneGap app
- Tube line alert for users
- Sync favourites for users
- Get me there

With the expansion of electric cars TFL have released an API providing charging point location data. With this information provided by TFL, the proposed future enhancements would provide users with something similar to the bus stop countdown, allowing the user to search for charging points near either their current location or postcode. The further work to be done will also provide directions from the location, which can be done with Google's Map API.

An alternative to the current application is to provide users with a native app, within an app store (such as Apple's App Store) using Adobe's PhoneGap framework. The reason for this alternative for users is that some prefer using an application on their phone rather than using a website. Providing this as an application can mean that a server should not be needed, as it only uses the APIs provided by TFL and Google.

Some potential users of this application will be commuters who use the same tube line every day. This provides the option of allowing users that have logged-in to set favourite tube lines and set an email notification between set hours, to email the user with any delays which are occurring on the line they use. This functionality will however, work better with an application on the user's Smartphone device using push functionality to notify the user within the application about service delays on tube lines that they set.

Since the application is provided cross-platform, some users may change their devices regularly, or completely change their mobile phone. This would therefore mean that they would need to add favourites on both devices or on their new device. Adding a syncing option for the user's favourites will mean that they will be available on multiple devices. This will therefore mean that when swapping between devices they will still be available to the user.

(TomTom app for iPhone/iPad)

A possible enhancement is to make an expansion to the current journey planner, displaying the information using a 'sat-nav' look and feel such as TomTom's navigation system for cars (as seen in Figure 53). This enhancement will be in addition to the current system as it will be for users departing at that

Figure 53 – TomTom

time, rather than in the future.

The potential problem with having an application constantly track a user by GPS will be the amount of data parsing between the user's mobile and the server, and the amount of battery life this will use. To combat this therefore, a secondary option will be to allow the user to click a button to find out the next set of instructions when they have completed their current set of instructions (such as get on the train towards London Victoria). The option of clicking a button will also be more viable for the mobile web application compared to tracking the user.

The prototyping methodology provided information mostly about the layout and navigation process of the application. It was possible to see from the prototype the design layout and how the user would interact with the application; and that the layout originally designed met the user's expectations, which were necessary for this project.

Since completion of this artefact, a deeper knowledge of the NodeJS server technology has been acquired. With this new found knowledge, some of the artefact would have been developed differently to ensure the speed of the system was at a highest priority, as well as minimal code being written and used. Instead, using some code multiple times for different systems may have been viable, such as the Live and Weekend tube status. JavaScript callbacks were not used as much as they could have been due to the knowledge acquired at the point of development compared to now. Starting again, with having some of the functions using the callback's functionality would possibly speed up the system's response time. Another option would be to have used coffee script which is a language which compiles JavaScript. This would help force minimal code to be written and to be compiled containing no errors.

More time was needed with the Journey Planner functionality than was spent, due to time constraints. As the build took longer than anticipated, it meant that the form did not have as much functionality for the user as initially intended; allowing the user to select advanced options such as which transport to use (such as bus only, tube, train etc.). Another intention was to allow the user to amend the form to resubmit, or retrieve more information (such as earlier or later) at a click of a button.

In order to manage this project it was vital to ensure that deadlines were set in order to complete the artefact and documentation by the deadline 27th April 2012. A Gantt chart was created to plot a structure to follow during the life of the project before any documentation and research (found in the Appendices 7.5 Original Gantt chart). It was known that these deadlines might not be strictly

adhered to throughout the project due to time constraints, other projects and the learning curves necessary. Therefore this was used as a guideline.

A Gantt chart of the project can be found in Appendix 7.6. This has been constructed comparing the correct time scales with the guidelines originally created. It can be seen that stages generally took longer than originally estimated. However, the learning curve was approximately the same, with learning of NodeJS being learnt throughout the project from design of the wireframes.

The learning curve of Node was the longest part of this project. In order to do this, a PC was used running Ubuntu 10.04 and small applications were created using tutorials. These tutorials provided an initial insight into how Node applications should be created to effectively use the potential provided by Node.

This project as a whole has been a success. The major functionalities have been created with some further expansion needed. The learning curves were as expected and did not prevent the project from meeting the end deadline of the build and documentation. Finishing this project has provided an insight into possible real-world applications of NodeJS, and its potential future as a server technology.

6 References

25 Billion app countdown. (n.d.). Retrieved March 5, 2012, from Apple: http://www.apple.com/itunes/25-billion-app-countdown/

Allamaraju, S. (2011, November 30). *Announcing ql.io*. Retrieved February 15, 2012, from eBay Tech Blog: http://www.ebaytechblog.com/2011/11/30/announcing-ql-io/

At the Forge - Node.JS. (2011, July 31). Retrieved February 15, 2012, from Linux Journal: http://www.linuxjournal.com/article/11014

Bowman, D. (2009). *What is a Prototyping Methodology?* Retrieved March 23, 2012, from David Bowman's Information Management Checklist: http://www.information-management-architect.com/prototyping-methodology.html

Brown, D. M. (2006). *Communicating Design: Developing Web Site Documentation for Design and Planning.* Berkeley: New Riders.

Checkland, P. (1999). *Systems Thinking, Systems Practice: Includes a 30-Year Retrospective.* Chichester: Wiley.

Garside, J. (2011, November 13). *London becomes 4G high speed internet hotspot*. Retrieved February 7, 2012, from The Guardian: http://www.guardian.co.uk/business/2011/nov/13/4g-high-speed-mobile-internet-trial-in-london

Gartner Says Sales of Mobile Devices Grew 5.6 Percent in Third Quarter of 2011; Smartphone Sales Increased 42 Percent. (2011, November 15). Retrieved February 24, 2012, from Gartner: http://www.gartner.com/it/page.jsp?id=1848514

Geisendörfer, F. (n.d.). *Node DateFormat*. Retrieved February 12, 2012, from Github: https://github.com/felixge/node-dateformat

Google Maps API Web Service. (2011, October 21). Retrieved February 26, 2012, from Google Developers: https://developers.google.com/maps/documentation/geocoding/

Google Maps JavaScript API v3. (n.d.). Retrieved February 13, 2012, from Google Developers: https://developers.google.com/maps/documentation/javascript/tutorial

Guidelines and Support. (n.d.). Retrieved March 28, 2012, from Transport for London: http://www.tfl.gov.uk/businessandpartners/syndication/16493.aspx

Hazaël-Massieux, D. (2012, February 20). *Mobile*. Retrieved March 8, 2012, from Standards for Web Applications on Mobile: February 2012 current state and roadmap: http://www.w3.org/2012/02/mobile-web-app-state/

Holowaychuk, T. J. (n.d.). *Express*. Retrieved October 28, 2011, from Github: https://github.com/visionmedia/express

Holowaychuk, T. J. (n.d.). *Express-Resource*. Retrieved January 18, 2012, from Github: https://github.com/visionmedia/express-resource

jQuery Mobile. (n.d.). Retrieved March 10, 2012, from jQuery Mobile: http://jquerymobile.com/

Larsson, J. S. (n.d.). *Google maps v3 plugin for jQuery and jQuery Mobile*. Retrieved February 15, 2012, from Google Code: http://code.google.com/p/jquery-ui-map/

Lester, S. (2008, April). *Soft Systems Methodology.* Retrieved November 27, 2011, from Stan Lester Developments: http://www.sld.demon.co.uk/ssm.pdf

Live Bus Departures. (n.d.). Retrieved March 15, 2011, from Transport for London: http://accessible.countdown.tfl.gov.uk/

Luden, I. (2012, March 12). *When It Comes To Shopping, Mobile Web Trumps Apps – Led By Amazon, Says Nielsen*. Retrieved March 13, 2012, from Tech Crunch: http://techcrunch.com/2012/03/12/when-it-comes-to-shopping-mobile-web-trumps-apps-led-by-amazon-says-nielsen/

Lyons, G. (2012, February 2). *2012 Mobile Market Share [Infographic]*. Retrieved February 28, 2012, from iCrossing: http://connect.icrossing.co.uk/2012-mobile-market-share-infographic_7962

Maciej. (n.d.). *Benchmarking Node.js - basic performance tests against Apache + PHP*. Retrieved March 8, 2012, from CHANGE(B)LOG: http://zgadzaj.com/benchmarking-nodejs-basic-performance-tests-against-apache-php

Nielsen, J. (1993). *Usability Engineering.* San Francisco: Morgan Kaufmann.

node.js. (n.d.). Retrieved February 11, 2012, from node.js: http://nodejs.org/

Original graded browser matrix. (n.d.). Retrieved March 10, 2012, from jQuery Mobile: http://jquerymobile.com/original-graded-browser-matrix/

Osborne, R. (2011, September 11). *London Buses and The Javascript Geolocation API*. Retrieved January 14, 2012, from Robin Osborne: http://robinosborne.co.uk/2011/09/11/london-buses-and-the-javascript-geolocation-api/#comment-1377

Parker, T. (2012, January 26). *jQuery Mobile 1.0.1 Released*. Retrieved February 4, 2012, from jQuery Mobile: http://jquerymobile.com/blog/2012/01/26/jquery-mobile-1-0-1-released/

Pearce, J. (2011, July 11). *Hello World*. Retrieved March 12, 2012, from Sencha: http://www.sencha.com/learn/hello-world/

Quarterly bus statistics q4. 2011. (2011, March 15). Retrieved March 27, 2012, from Department of Transport: http://www.dft.gov.uk/statistics/releases/quarterly-bus-statistics-quarter-4-2011/

Rabin, J., & McCathie, C. N. (2008, July 29). *Mobile Web Best Practices 1.0*. Retrieved February 7, 2012, from W3C: http://www.w3.org/TR/2008/REC-mobile-bp-20080729/

Resig, J. (2011, September 12). *jQuery 1.6.4 Release notes*. Retrieved December 12, 2011, from jQuery: http://blog.jquery.com/2011/09/12/jquery-1-6-4-released/

Sage, J. T. (n.d.). *jQuery Mobile Datebox*. Retrieved February 24, 2012, from Github: https://github.com/jtsage/jquery-mobile-datebox

Sencha Touch 2. (2012). Retrieved March 12, 2012, from Sencha: http://www.sencha.com/products/touch/

Short, A. (n.d.). *Countdown*. Retrieved January 8, 2012, from Github: https://github.com/adrianshort/countdown/blob/master/bin/countdown

Smart, M. (n.d.). *The Waterfall Development Methodology*. Retrieved March 30, 2012, from Learn Access with the Smart Method: http://www.learnaccessvba.com/application_development/waterfall_method.htm

Synchros Interactive. (2010, April 24). Retrieved March 8, 2012, from Node.js has a bright future: http://www.synchrosinteractive.com/blog/9-nodejs/22-nodejs-has-a-bright-future

Telecommunications. (n.d.). Retrieved February 7, 2012, from Hutchison Whampoa: http://www.hutchison-whampoa.com/europe/eng/telecom/telecom.htm

The Concept of Proto-Typing in the System Development Process. (n.d.). Retrieved March 23, 2012, from MBA Knowledge Base: http://www.mbaknol.com/management-information-systems/the-concept-of-proto-typing-in-the-system-development-process/

The Web and Mobile Devices. (n.d.). Retrieved February 07, 2012, from W3C: http://www.w3.org/Mobile/

ThemeRoller for jQuery Mobile. (n.d.). Retrieved January 9, 2012, from jQuery Mobile: http://jquerymobile.com/themeroller/

TomTom app for iPhone/iPad. (n.d.). Retrieved April 7, 2012, from TomTom: http://www.tomtom.com/en_gb/products/mobile-navigation/tomtom-app-for-iphone/index.jsp

Usage of client-side programming languages for websites. (2012, March 8). Retrieved March 8, 2012, from W3Techs: http://w3techs.com/technologies/overview/client_side_language/all

Usage of JavaScript libraries for websites. (2012, March 12). Retrieved March 12, 2012, from W3Techs: http://w3techs.com/technologies/overview/javascript_library/all

Usage of server-side programming languages for websites. (2012, March 8). Retrieved March 8, 2012, from W3Techs: http://w3techs.com/technologies/overview/programming_language/all

Usage statistics and market share of PHP for websites. (2012, March 8). Retrieved March 8, 2012, from W3Techs: http://w3techs.com/technologies/details/pl-php/all/all

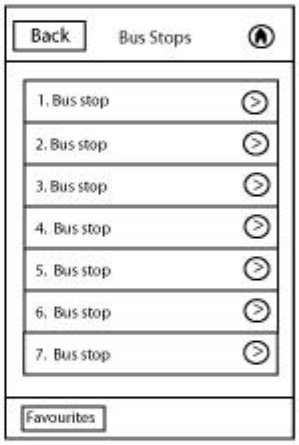
W3C HTML5 Logo. (n.d.). Retrieved March 7, 2012, from W3C: http://www.w3.org/html/logo/

Witt, C. (n.d.). *Get the distance between two (world) coordinates - a nodejs module*. Retrieved January 20, 2012, from Github: https://gist.github.com/1604972

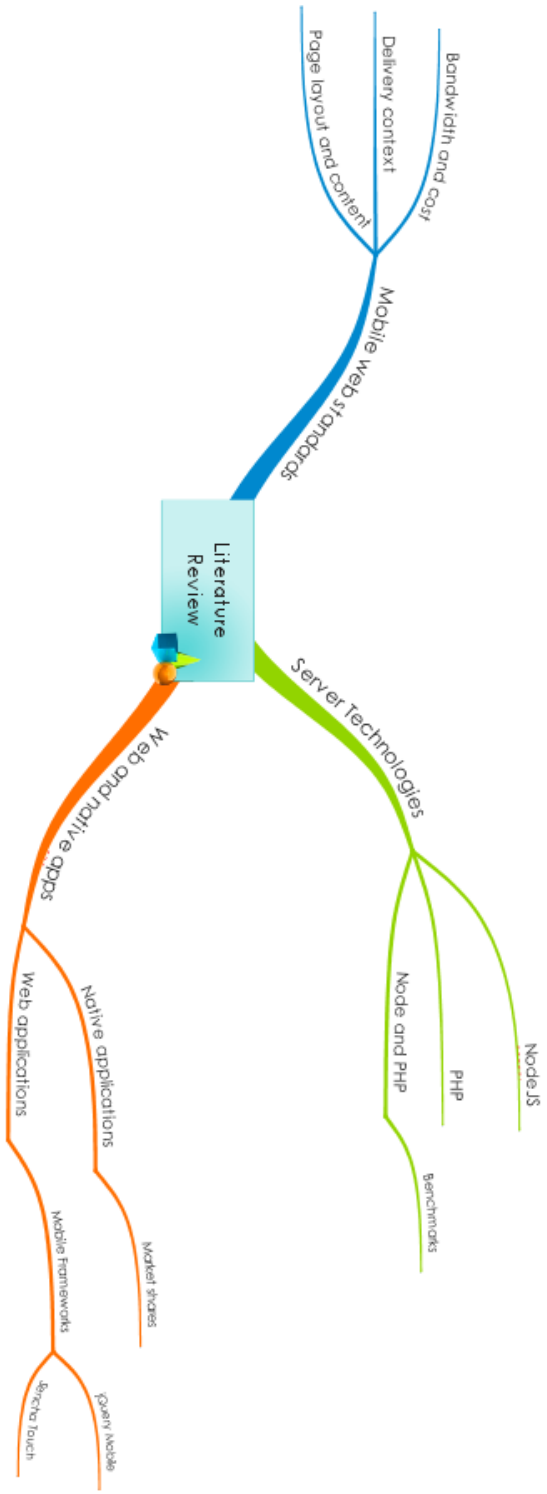
XML-Stream. (n.d.). Retrieved February 5, 2012, from Github: https://github.com/assistunion/xml-stream

7 Appendices

7.1 Initial Wireframes



7.2 Literature Review Mind Map



7.3 Transport for London disclaimer

Terms and conditions & Privacy

Terms and Conditions | Privacy

Use of Transport for London's (TfL's) websites http://www.tfl.gov.uk/ (including sub domains and the following related sites: oyster.tfl.gov.uk, congestioncharging.tfl.gov.uk and web.barclayscyclehire.tfl.gov.uk), is governed by these terms and conditions.

If a user does not accept these terms and conditions they should exit the TfL website immediately. Specific attention is also drawn to the notice concerning copyright (set out below in these terms and conditions) and the privacy statement.

TfL may at any time revise these terms and conditions without notice. It is up to the user to regularly review the terms and conditions in case there are any changes. Continued use of the TfL website after a change has been made is to be treated as acceptance of that change by the user.

Disclaimer
Links
Jurisdiction
Copyright
Privacy policy
Terms of Use specific to Oyster, Congestion Charging and the Barclays Cycle Hire websites

Disclaimer

While TfL tries hard to ensure that these websites are functioning correctly and are as accurate as possible, this will not always be achievable and therefore the following disclaimer applies.

These websites are provided "as is" without any representation or endorsement made and without warranty of any kind whether express or implied, including but not limited to the implied warranties of satisfactory quality, fitness for a particular purpose, compatibility, non-infringement, accuracy and security. TfL does not guarantee or represent that the content and/or facilities available or accessible via these websites will always be accurate, complete or current or that access to the websites will be uninterrupted.

To the extent permitted by law TfL expressly disclaims all liability for any direct, indirect or consequential loss or damage occasioned from the use or inability to use these websites, whether directly or indirectly resulting from inaccuracies, defects, viruses, errors - whether typographical or otherwise, omissions, out of date information or otherwise.

Accessing these websites and the downloading of material from them is done entirely at the user's own risk. The user, and not TfL, will be entirely responsible for any resulting damage to software or computer systems and/or any resulting loss of data even if TfL has been advised of the possibility of such damage.

Links

These websites may contain links to other websites over which TfL has no control. Such links are supplied solely for the convenience of users. TfL is not responsible for the contents or reliability of the linked websites and does not necessarily endorse the views expressed within them or the organisation or persons providing them in any way. TfL does not guarantee that these links will work all of the time and we have no control over the availability of the linked pages.

Providers of other websites may place text-based links to pages on the TfL websites without seeking prior permission. However such links must not open TfL website pages into frames within another website. Pages from the TfL websites must be loaded into the user's entire browser window. The use of the TfL logo or the logos of any associated or subsidiary organisations must not be used for promotional or linking purposes unless prior approval in writing has been given. Please contact TfL Customer Services for further details on obtaining such approval.

Jurisdiction

These terms and conditions will be governed and construed in accordance with the laws of England. The courts of England shall have exclusive jurisdiction over disputes between a user and TfL arising out of the access or use of these websites. Omission by TfL to exercise any right under these terms and conditions will not constitute a waiver of such right unless expressly stated by TfL in writing.

Copyright

Unless expressly stated the copyright and other intellectual property rights (such as, design rights, trademarks, patents etc.) in any material provided on the TfL websites remains the property of TfL (or as the case may be another rightful owner). TfL owned material on these websites including text and images, may not be printed, copied, reproduced, republished, downloaded, posted, displayed, modified, reused, broadcast or transmitted in any way, except for the user's own personal non-commercial use. Permission for any other type of use must be obtained - TfL Customer Services may be contacted for details.

Privacy policy

TfL respects user privacy and operates under a strict privacy statement. The privacy statement explains, amongst other things, that we may collect information about you. By using these websites you consent to the terms of the privacy statement.

Terms of Use specific to Oyster, Congestion Charging and the Barclays Cycle Hire websites

The following are prohibited in relation to the Oyster (oyster.tfl.gov.uk), Congestion Charging (congestioncharging.tfl.gov.uk) and Barclays Cycle Hire (web.barclayscyclehire.tfl.gov.uk, with the exception of web.barclayscyclehire.tfl.gov.uk/maps), websites, except where expressly permitted under a written licence agreement with TfL:

i. Use of data from the Oyster, Congestion Charging and Barclays Cycle Hire websites to populate or update any other software or database
ii. Use of any automated system, software or process to extract content and/or data, including trawling, data mining and screen scraping.

See also

Freedom of Information (/foi/default.aspx)
Privacy (/termsandconditions/897.aspx)

7.4 Project Initiation Document

Basic details

Student name:	Nicholas Simon White
Draft project title:	To create a transport for London jQuery mobile site
Course:	Business Information Systems
Client organisation:	N/A
Client contact name:	N/A
Project supervisor:	Richard Boakes

Outline of the project environment and problem to be solved

For this project the clientele will be individuals that use public transport within London; this could be for commuters going to and from work, tourists and local residents moving around the city. Currently I perceive that the current mobile technologies provided for users do not have a sufficient GUI for the information that is available. Currently this problem has been tackled by creating a number of mobile applications including Tube Map (http://itunes.apple.com/gb/app/tube-map/id320969612?mt=8), London Tube Status (http://itunes.apple.com/gb/app/london-tube-status/id285535503?mt=8 and Traffic Camera TFL (http://itunes.apple.com/gb/app/traffic-camera-tfl/id392523422?mt=8). At present, there is no journey planning application for the Blackberry. Another issue is that Android and iPhone applications do not link multiple functionalities together, which mean users may have to download several different applications to meet all of their travel needs.

Project aim and objectives

My aim is to create a system that links various applications and data sources together. This system will provide those travelling around London with information that meets their needs when planning journeys.

The objectives to meet this aim will be to use their geo-location as a start or end point for travelling; allow users to store favourite travel journeys, look at tube departure boards, and line service information. All linking together with the users planned journeys and cycle hire schemes.

Project deliverables

During the development of this project a website will be built that can be used with Blackberry's, iPhone's and Android smart phones; as well as a database system that runs on the server. Documents shall also be produced alongside this website these will be a project report, wireframes, and design and test scripts with results.

Project constraints

The limitations for this project will be to ensure the size of the site is kept to a minimum to keep the users page load as short as possible. Accuracy of the user's geo-location can

cause limitations as well. The end users mobile data services can be costly, therefore minimising the data transferred between server and client may put a restriction on the design and implementation stages.

Project approach

The background research that I will need to look at is what server technologies are best for my project. I will also look at previously created applications that have been built to tackle this same problem. In order to find out my clientele's requirements I will ask commuters that use the public transport system on a daily basis. I am proposing using a library service such as jQuery Mobile will help to accelerate the development of the system, so skills in this or a similar module must be acquired before build; in order to acquire these I will look at online tutorials to create simple sites in order to build a system that meets the requirements found from my clients.

Facilities and resources

I will require a UNIX server in order to store the website online and a web address for my clients to be able to visit on their smart phones. I will require a web development application in order for me to code the website using jQuery. The use of open-source Ubuntu a UNIX based operating system, Dropbox a free online storage service, PuTTY an open-source terminal emulator, NetBeans a open-source software development program.

For the testing of this service on various mobile devices (e.g. Blackberry, iPhone) emulation software will be used.

Log of risks

Possible risks that could occur during this project is the loss of server, computer and files; in order to combat this files will be backed up on a daily basis using a program called Dropbox. Files shall be stored using the backup method known as grandfather, father, and son.

The use of programming languages that I have had none or little previous experience with; this will then hinder the speed of building my artefact; to combat this the use of online resources such as tutorials will be used to get up to speed.

My final artefact will depend on greatly the use of TFL API's if servers go down or the service is ended by TFL this will then mean that some functionalities if not all of them will contain errors.

Starting point for research

I will look at online resources and existing reports on mobile web technologies and their capabilities there drawbacks. As well as this I will look at existing websites that have been designed purely for smart phones, looking at how the users may use my system.

I will be looking at online resources and existing reports in order to choose which the best choice for my server to use is; this could be anything from NodeJS, PHP or ASP. I will need to look at what is best to fulfil my needs for this project; for build and the clientele's needs.

I will be looking at NodeJS which is JavaScript server technology I will need to research how best to use this language and whether or not it is the best.

Breakdown of tasks

In order to create my artefact:

- I will need to look at all of the functionalities that I wish to bring to the user and link together
- Create a workflow
- Create wireframes for my system
- Have users test my wireframes
- Design my system
- Create the server for system
- Create the database to store users details
- Create the system based on the designs and wireframe stage

Project plan

What are you going to do when? (This may be an attached output from MS Project etc.)

What risks to the success of the project have you identified? What steps can you take to minimise them?

Legal, ethical, professional, social issues

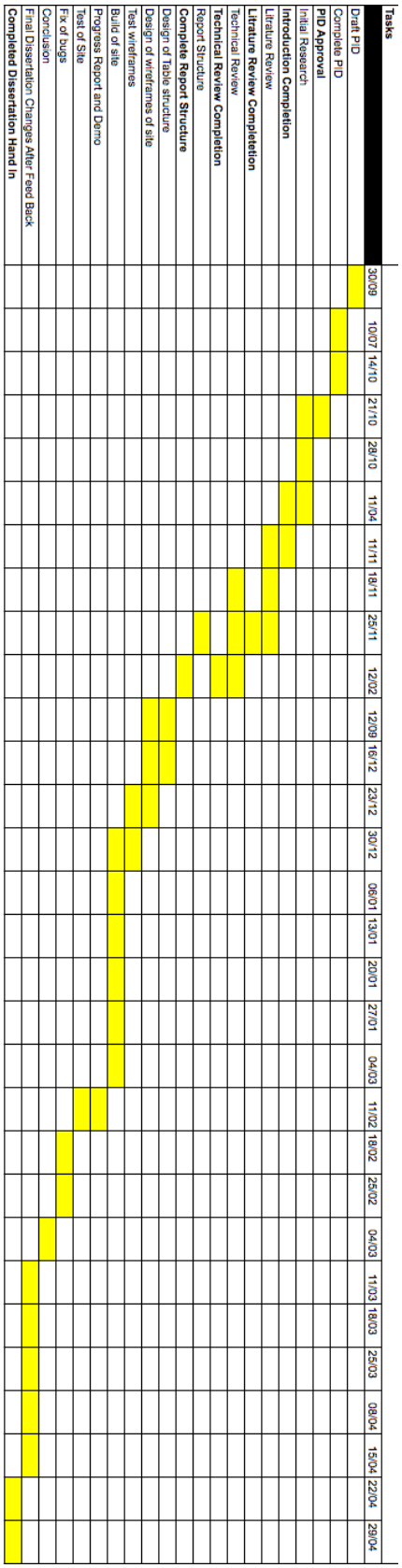
I will need to meet the requirements set by TFL and Google when using their API's; I will also need to ensure that no code is used without knowledge of the user's consent to ensure that no Copyright issues have been breached in building my system.

To ensure this only code from tutorials; or open source systems will be used when not writing my own code.

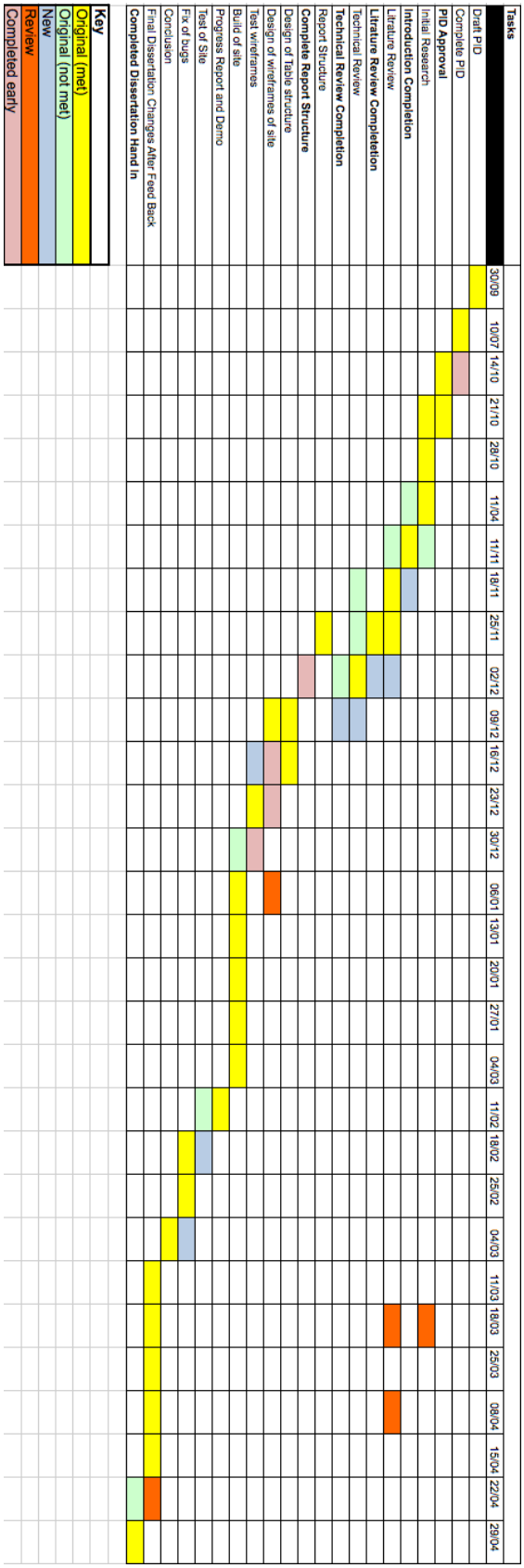
Signatures

	Signature:	Date:
Student		
Client		
Project supervisor		

7.5 Original Gantt chart



7.6 Completed Gantt chart



7.7 Ethical Checklist

PJE40 and PJS40

University of Portsmouth

Ethical Examination

Undergraduate Final Year Projects

School of Computing

Faculty of Technology

This document describes the issues that you need to consider before you start your investigations. This is particularly important where your work may involve other people (human subjects) for the collection of information as part of your project work.

The examination takes the form of a checklist of 12 questions. Each question has come guidance notes.

Consider each question is turn and check the box for Yes or No.

You are then asked to write a short entry explaining the reason for your reply.

For example:

6. Are you in a position of authority or influence over any of the human subjects in your study? Comments: ***Although all the human subjects will be staff members at the University of Portsmouth, none of them are in my own department or area and none are subordinate to me in a management structure. Therefore I can see no way that I could have undue influence over them They could take part completely voluntarily.***	Yes No ☐ X

If a grey box is ticked then your project ideas need to be looked at more closely, and you MUST discuss this matter with your project Supervisor.

The final sections deal with Information Sheet(s) and Informed Consent, and you must attach any extra documents concerning these (where relevant) to this Ethical Examination at time of the submission of your Initial Report.

Ethics Information: 12-point Checklist

1. Will the human subjects be exposed to any risks greater than those encountered in their normal lifestyle? ***For example: could the study induce psychological stress or anxiety; is more than mild discomfort or pain likely to result from the study; will the study involve prolonged or repetitive activities?*** ***Investigators have a responsibility to protect human subjects from physical and mental harm during the investigation. The risk of harm must be deemed to be no greater than in their normal lifestyles.*** Comments:	Yes No ☐ X
2. Will the human subjects be exposed to any non-standard hardware or non-validated instruments? *Human subjects should not be exposed to any risks associated with the use of non-standard equipment: anything other than pen-and-paper, or typical interactions with desktop, laptop PC's, tablet PC's, PDA's or mobile phones are considered non-standard (for example, using a VR room) nor should they be subjected to non-validated instruments e.g. unscrutinised questionnaires.* Comments:	Yes No ☐ X
3. Will the human subjects voluntarily give consent? ***If the results of an evaluation (for example) are likely to be used beyond the term of the project (for example, software is to be deployed or data is to be published), then signed consent is necessary. A separate consent form should be signed by each human subject. Return of a consent email can constitute written consent if this has been made clear to the human subject.*** ***Otherwise verbal consent is sufficient and should be explicitly requested in the introductory script (Information Sheet).*** Comments:	Yes No X ☐
4. Will any financial, or other, inducements (other than reasonable expenses and compensation for time) be offered to human subjects? *The payment of human subjects must not be used to coerce them against their better judgement, or to induce them to risk harm beyond that which they risk without payment in their normal lifestyle.* Comments:	Yes No ☐ X

5. Does the study involve human subjects who are unable to give informed consent (for example: children under 18, people with learning disabilities, unconscious patients). *Parental consent is required for human subjects under the age of 18. Additional consent is required for human subjects with impairments, and people assessed to be lacking in mental capacity. If consent is gained from a person other than the human subject themselves e.g. a parent, then written consent* ***must*** *be obtained.* Comments:	Yes ☐ No ☒
6. Are you in a position of authority or influence over any of your human subjects? ***A person in a position of authority or influence over any human subject must not be allowed to pressurize them to take part in, or remain in, any study.*** Comments:	Yes ☐ No ☒
7. Are the human subjects being provided with sufficient details of the study at an appropriate level of understanding? ***All human subjects should be able to understand the information provided in any documentation and/or verbal information they receive about the experiment or study. They have the right to withdraw at any time during the investigation, and they must be able to contact the investigator after the investigation. They should be given the details of both student and supervisor as part of the debriefing. This information should be in the introductory script (Information Sheet).*** Comments:	Yes ☒ No ☐
8. After the study, will human subjects be provided with feedback about their involvement and be able to ask any questions they may have about this involvement? *If the human subjects request further information, the investigator must provide the human subjects with sufficient details to enable them to understand the nature of the investigation and their part in it.* Comments:	Yes ☒ No ☐
9. Will the human subjects be informed of the true aims and objectives of the study? *Withholding information or misleading human subjects is unacceptable if human subjects are likely to object or show unease when debriefed. It must be clear to human subjects if information is being withheld in order to elicit a true*	Yes ☒ No ☐

response. This should precede any analysis of the data. Comments:	
10. Will the data collected from the human subjects be made available to others (where appropriate and only in relation to this research study), and be stored, in an anonymous form? *All human subject data (hard-copy and soft-copy) should both be stored securely and, if appropriate made available, in an anonymous form. Making human subject data available to a third party may be relevant where a student is taking part in a wider research project eg. for a member of the University staff, in which case anonymity of human subject data must be preserved.* Comments:	Yes No X ☐
11. Will the study involve NHS patients, staff, or premises? ***If yes, then an application must be made to the appropriate external NHS Local Research Ethics Committee (LREC). For projects other than postgraduate research studies, the length of time for gaining external approval may not fit into a project timescale.*** Comments:	Yes No ☐ X
12. Will the study involve the investigator and/or any human subject, in activities that could be considered contentious, morally unacceptable, or illegal? *If yes, then further approval must be sought. For example: a project involving the study of pornography on the web will fall into this category. It is possible that the project may not be allowed to proceed.* Comments:	Yes No ☐ X

Please attach the following:

- Any Information Sheet(s) or introductory script(s) that the investigator has created for the benefit of the human subjects in the study. ***(See http://www.btinternet.com/~trking for examples of Information Sheets that set out details of a research study for human subjects).***

- Any documentation that the investigator has created to gather informed consent from the human subjects. This may be an Informed Consent Form, or a form of wording used to get verbal consent. ***(See http://www.btinternet.com/~trking/icf.htm for an example of an Informed Consent Form for research study with human subjects).***

==========

By signing this form, I AGREE to abide by the decisions made in the above points.

If at any time during my project, my answers would change from a white box to a grey box, then I MUST seek re-approval for my project. I understand that if I do not do so, then it is possible that I may FAIL the project component of my course.

Student name: ……..…… Jupiter number: ……

Student signature: …….. Date ……

==========

Supervisor signature: ……. Date ……..

EthicalExaminationStudGenv2.doc t.king 080506

7.8 Reproducing the development environment

This mobile application currently runs on NodeJS v0.6.13, a tar.gz file for Ubuntu Linux 10.04 can be found on the CD provided with this documentation. This guide will assume that no dependencies have been previously installed on the Linux environment.

Installing NodeJS' dependencies

```
sudo apt-get install g++ curl libssl-dev apache2-utils

sudo apt-get install git-core
```

Installing NodeJS

Initially you will are required to navigate to the same location as you have placed NodeJS (this guide will assume it is on the root).

```
tar –zxvf  node-v0.6.13.tar.gz
cd node-v0.6.13
./configure
make
make install
```

Running Mobile London Site

Node should not be fully installed, typing node into the terminal should start-up Node now simply try:

```
Console.log('hello world');
```

Copy the folders node_modules and Site to the same location as node. Now to initialise the Mobile London website run the code.

```
cd ~/Site/MobileLondon && export NODE_ENV=production &&
node app.js
```

Terminal should respond with 'Express server listening on port 8080 in production mode'. The problem with this is that websites generally use Port 80 when in production, to enforce this rather than granting access with sudo enter the following code to forward the port from 8080 (which Mobile London uses) to 80. This will need to be run each time the server is powered off and on.

```
sudo iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT
--to-ports 8080
```

Now opening your browser and entering http://localhost should open the Mobile London site (the same as which is provided on www.MobileLondon.org) however, the Journey Planner will not work as TFL fix this API to IP addresses which have the correct privileges, to sign up for this visit http://www.tfl.gov.uk/businessandpartners/syndication/16492.aspx. Currently this server will only work while the terminal is kept open, which is not necessarily the best option in order to stop this a node module called Forever should be used. Node Package Manager will need to be installed initially (see http://npmjs.org/doc/README.html for details of installing) once this has been done run the code.

```
sudo curl http://npmjs.org/install.sh | sh
npm install –g forever
NODE_ENV=production forever start ~/Site/MobileLondon/app.js
```