Mobile dynamic campus GPS turn-to-turn navigation system

CHI CHUNG DEREK LEE

BSc in Computer Science

27 April 2016

SCHOOL OF COMPUTING AND MATHEMATICS

Keele University

Keele

Staffordshire

ST5 5BG

Abstract

This report presents the development of a web-based navigation system to assist students and visitors at the Keele University campus, a guidance system with emphasis on pedestrian navigation aimed at suggesting the shortest path from one location to another within the campus. A mobile campus guidance system can improve the campus experience and help first-time students and visitors get familiar with the environment in a short time and lead them to places of interests quickly.

This research demonstrates a practical solution using open sources to create mobile routing services for university campuses or small towns, using crowd sourced data and open-source mapping application program interfaces (API). Open-source "Leaflet" (http://leafletjs.com) JavaScript library and mapping API for mobile-friendly interactive maps have been used to create the interface. The API uses map data from "OpenStreetMap" (http://openstreetmap.com) built by a community of mappers who contribute and maintain geographic data about roads, trails, cafés, railway stations and much more, all over the world. A prototype web-based navigation system was developed for mobile users. The system demonstrates that a solution to detailed navigational services for pedestrians, cyclists and drivers can be economical and feasible for university campuses and small towns.

Acknowledgements

I want to thank my supervisor Mr. Dave Collins for his guidance and his helpful advices.

I would like to thank my family and all my friends from Keele University. Thank you for the help and support before exams and also for the fun times we had during our entire studies.

I would also like to take this opportunity to thank everyone that helped and supported me throughout my studies and especially this report.

Table of Contents

Table of Figures

List of Abbreviations

Symbol	Definition
API	Application Programming Interface
CSS	Cascading Style Sheets
GIS	Geographic Information Service
GeoJSON	Geographic JavaScript Object Notation
GPS	Global Positioning System
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
JS	JavaScript
LBS	Location Based Service
NGO	Non-Governmental Organization
ODbL	Open Database License
OSM	OpenStreetMap
OSRM	Open Source Routing Machine
POI	Point of Interests
RF	Radio Frequency
RFID	Radio-frequency identification
RSSI	Received Signal Strength Indication
TSP	Traveling Salesman Problem
UI	User Interface
URL	Uniform Resource Locator
WiFi	Wireless Fidelity

1. Introduction

The main goal of this project is to present a practical solution for mobile routing services for a small town, such as the Keele University campus, using crowdsourced map data and open-source APIs to develop a Location Based System (LBS) for mobile devices. The System is useful for students by helping users finding the shortest path from one location to another, thus reducing trip delays and user frustration, particularly when navigation directions accounts for campus events, such as a graduation ceremony or a football game.

Routing is one of the most essential services provided by Location Based Services (LBS). Mobile routing services encompass way-finding applications for pedestrians, vehicle drivers and cyclists, delivered using mobile terminals. Much research has been done on mobile routing algorithms because of the complexity of application environments and the variety of user requirements (Huang et al., 2007; Huang and Wu, 2008). Like POI (Point of Interests) query services, mobile routing services have become more and more popular in the real world, especially those applications for vehicle drivers. Some free map platforms, such as Google Maps and Microsoft Bing Maps, include direction modules which provide turn-by-turn routing services. However, most current mobile routing services are provided mainly with detailed content only in relatively big cities or for major streets in rural areas. LBS for small towns are generally ignored as many people and most companies believe that there is little demand in such areas, and there is no need to invest in detailed data collection for such places. University towns and some tourist resorts are typical examples of small towns where there are many visitors potentially requiring access to LBS. They do not have much time to get familiar with the places. Better local LBS, especially with routing services, can help visitors to get familiar with a new environment in a short time and lead them to places of interest.

Most location-based services focus on major cities while people believe that small towns generally does not in demand of such applications. On the contrary, the demand for LBS applications in some small towns such as university towns and tourist resorts can be as strong as big cities. Better location-based services, especially routing services, can help visitors get familiar with the strange environment in a short period of time and lead them to places of their interest. However, system developers are faced with two problems. The first one is cost; both in terms of development time and data costs. Mash-up technology and open source data could provide an answer. The other obstacle is the availability of suitable data with the required accuracy and detail. This is more serious as most free map services, such as Google Maps and Microsoft Bing Maps, do not provide sufficient detailed or accurate data for routing services. One feasible and economical way is to create the map ourselves and have it updated by the public.

1.1. Motivation

With 240 hectares (600 acres) of landscaped grounds (Keele.ac.uk, 2016), Keele University is relatively hard to navigate, especially for new students, and visitors coming to the university ground everyday. A campus guidance system is always a helpful tool for those unfamiliar with the campus to orientate themselves and guide them around the campus. Where is the classroom? Where are the conference sessions at? Where is the pub? These are some common questions asked by students and visitors, especially when they are new to the university campus. These type of queries are among a set of challenging navigation tasks for students and visitors

With the widespread use of mobile mapping Google Maps and Bing Maps, it is now easy to obtain mapping data. This has allowed system developers to build Location Based Services (LBS) for environments such as small towns and university campuses. The main motivation for this research was to see how feasible it is to build a pedestrian navigation system completely with open technologies and freely available data, and to maximize re-use potential for other events on the campus.

1.2. Goals

The aim of this work is to propose an effective, efficient and low-cost solution for providing a Location Based Services system, with routing service, for the Keele University campus.

Local mobile routing services with POI queries demand primarily accuracy and sufficiency of data of the area. The main requirements of the project are (a) efficient system development and easy maintenance, (b) effective, useful data and services, and (c) economical. The solution should also be relatively easy and fast to deploy.

This project illustrates a solution based on the open-source Leaflet Javascript mapping library and OpenStreetMap mapping data, with the use of Leaflet Routing Machine API as routing engines, aiming to achieve the following goals:

To provide a map of the campus for users to view digitally via a web browser.

To provide routes of navigating within the campus, with possible routing options.

To devise a dynamic turn-by-turn navigation system using the GPS built-in to the mobile device.

To incorporate a reactive website which the user can use on either a mobile device or a PC.

To provide a platform for later development of a campus information system based on selected location, such as latest update on events and news

1.3. Report Structure

This report describes the entire process of developing a web-based application for Location Based System with pedestrian navigation purposes. The rest of the paper is organized in five sections following this introduction section. In Section 2, background Geographic Information System and navigation techniques investigated are briefly explained. In Section 3, related work on the system components is discussed, such as mapping data, the reasons why “OpenStreetMap” is used as a data source, “Leaflet” for rendering map data, as well as the routing engines APIs. Section 4 presents a discussion of the development, tests and evaluation. Section 5 discusses problems with the project and puts forth some suggestions for future development. Section 6 closes the report with a review of the conclusions made from the work and future opportunities.

2. Background Studies

2.1. Geographic Information System

A Geographic Information Systems (GIS) is designed to capture, save, retrieve, map and check geographical data, that is, information related to positions on Earth's surface. A GIS system stores any kind of information relating to a particular geographical location. Spatial features are saved in a coordinate system referencing a certain place on the surface of the earth.

The main use of geographic information systems is resource management, development planning and scientific research.

Geographical positions are described by latitude and longitude which are spherical measures of the angles from a point on the surface, from the center of the earth. As Shown in Figure 2-1, (1) represent latitude lines and longitude line represented by (2). The red dot in this figure is described by the coordinates as 50 degrees east (3) and 40 degrees north (4)

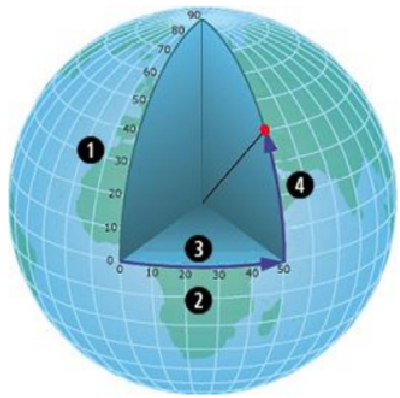


Figure 2-1 - GIS - Latitude and Longitude (Webhelp.esri.com, 2014)

2.2. GPS-based Positioning

The Global Positioning System (GPS) is a freely accessible system based on satellites, which is the leading technology to determine locations on mobile devices. Almost all smartphone and mobile devices on the market are capable of receiving GPS signals. The GPS receiver requires a line of sight to at least four satellites in order to determine a position. A highly accurate atomic clock in each satellite sends out the time along with position data of the satellite and error correction data. Upon receiving the signal, a GPS receiver computes the distance to the satellite based on the difference between the satellite time and its own internal clock. The distance to the satellite and its absolute position defines a sphere, centered at the satellite (see Figure 2-4). The position of the receiver has to be at one point on this surface. With additional data from a second and third satellite, three spheres can be drawn, and the position of the receiver position can be narrowed down to two points of intersection. The intersection point closest to the earth's surface is the correct position of a GPS receiver such as a smartphone. The fourth satellite is for error correction in terms of accuracy of time since the receiver clock is not as accurate as the atomic clocks of the satellites. The position determining accuracy by GPS depends on the receiver, where most consumer GPS receivers have an accuracy of 5 to 10 meters.

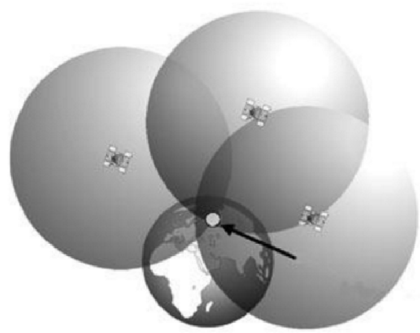


Figure 2-2 - GPS - Intersecting Spheres (Webhelp.esri.com, 2014)

2.3. WiFi-based Positioning

Due to the fact that GPS receiver needs a line of sight to at least four satellites, the GPS method for positioning only works outdoors, it is not usable indoors. The location tracking method using mobile phone network is also not suitable with a low positioning accuracy of 50 - 300 meters. However, different approaches based on Wi-Fi technology can be used to determine a usable indoor location estimate.

2.3.1. Nearest Sensor

The simplest way for determining a location estimate based on wireless networks is to use the nearest access point. This system is integrated into most of the access point management systems. It determines the access point to which a client is connected. Under the assumption that this is the closest access point and based on the information, it computes how far the signal of this access point radiates. The client has to be within the sensor range in the area.

2.3.2. Received Signal Strength Indication (RSSI)

Similar to the computation of the GPS-based position, a location estimate can be calculated by the received signal strength of the nearby WiFi networks. With the outgoing power level of the access point and signal strength received by the client, the absolute loss of the signal strength is calculated and the distance to WiFi access points can be deduced from the RSSI. With the distance to three access points and their absolute position, the position of the client can be computed by using a trilateration algorithm.

2.3.3. Radio Frequency Fingerprinting

A relatively high effort is needed for the initial setup of this method. A physical walk around with special spectrum analysis units is required to create radio frequency (RF) fingerprints for different points of the area where the location should be tracked. A fingerprint identifies locations by measuring the radio frequency setting, which is created by the wireless network access points. Management systems from different vendors include functionality to manage these fingerprints. Based on the measured fingerprints, these systems have the ability to compute fingerprints for every other point of the target area with sophisticated interpolation algorithms. To determine the position of a mobile device, the device sends the current RF fingerprint of its environment to a server. The server compares this real-time fingerprint with the ones in the database and computes a position based on the similar fingerprints. The benefit of Radio Frequency Fingerprinting is that it also takes into account environmental effects such as reflections on walls or other objects.

2.4. Dijkstra's Shortest Path Algorithm

The Dijkstra’s algorithm (GeeksforGeeks, 2012) is a powerful and the most commonly used tool to solve the single-source shortest path problem today. The algorithm efficiently finds the shortest paths from one node in a graph to another, which may represent, for example, road networks. It was discovered by Dutch computer scientist Edsger W. Dijkstra in 1956 and published three years later. The basic iteration of the algorithm can be understood using the following example network of nodes (node 1 to 8) and distances between each node:

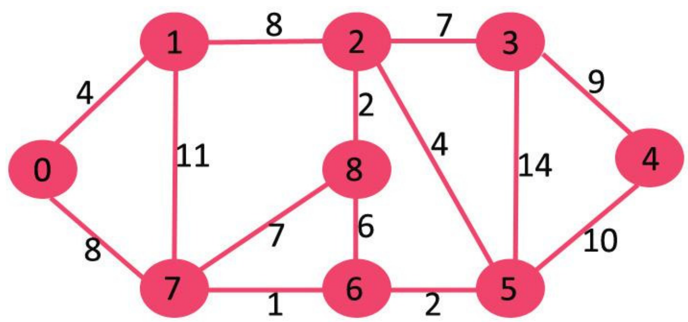


Figure 2-3(a) - The Dijkstra's algorithm example (GeeksforGeeks, 2012)

1. Initialize every node a tentative distance value: set initial node 0 to zero distance value, and infinity for all other nodes.

2. Pick the node with the minimum distance value and set as current, i.e. node 0. Set the initial node as current. Mark all other nodes unvisited.

3. For the current node, consider unvisited neighbours and update distance values of its adjacent nodes. Adjacent nodes of 0 are 1 and 7. The distance values of 1 and 7 are updated as 4 and 8.

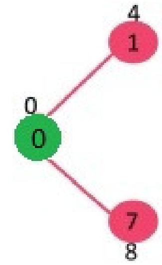


Figure 2-3(b) - The Dijkstra's algorithm example (GeeksforGeeks, 2012)

4. After considering all of the neighbors of the current node, mark the current node as visited which is then removed from the unvisited set. A visited node will never be checked again.

5. Find the vertex with the minimum distance value and not already visited. The vertex 1 is picked. Revise the distance values of adjacent vertices of 1. The distance value of vertex 2 becomes 12. Mark node 4 as visited node.

6.

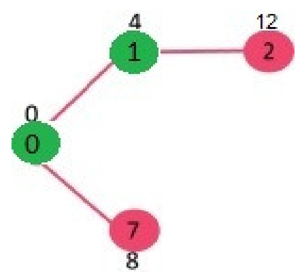


Figure 2-3(c) - The Dijkstra's algorithm example (GeeksforGeeks, 2012)

6. Pick again the node with minimum distance value and not already visited. Node 7 is picked. Update the distance values of adjacent nodes of 7. The distance value of vertex 6 and 8 becomes finite (15 and 9 respectively).

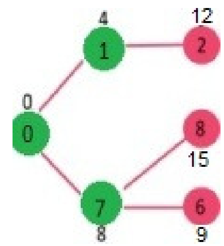


Figure 2-3(d) - The Dijkstra's algorithm example (GeeksforGeeks, 2012)

7. Repeat the above steps until all unvisited nodes on the graph are visited, and a Shortest Path Tree is achieved.

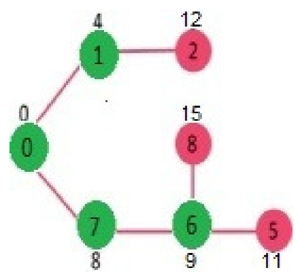


Figure 2-3(e) - The Dijkstra's algorithm example (GeeksforGeeks, 2012)

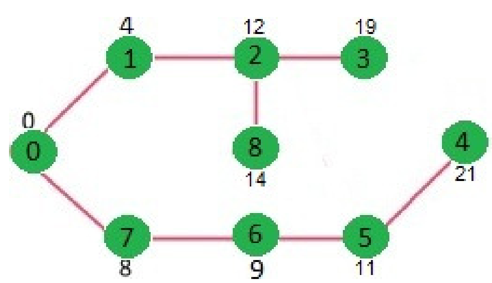


Figure 2-3(f) - The Dijkstra's algorithm example (GeeksforGeeks, 2012)

3. Related Work

Integrated web applications or mashups are increasing being used. In recent years, there has been an increase in the use of free Maps API provided by various web map services, such as Goggle Maps (https://developers.google.com/maps/) and Bing Maps (http://www.microsoft.com/maps/) to create customized map based web service using their interface and data along with some of the user's own data. When developing such navigation applications, the availability of complete digital map data for an area of interest presents a challenge particularly if the extent of the area is not within the limits of a major city. And when the chosen area is a small town and in a small city, the availability of reasonably good digital web-based maps is also inadequate. Maps of Keele University campus area across various web-based map service providers (Google Maps, Microsoft's Bing Maps, and OpenStreetMap) was compared and found that for most map providers, there is a lack of high-quality spatial data for our area of interests.

3.1. Map Data

Keele University is located about three miles from Newcastle-under-Lyme, Staffordshire, England (Longitude 53.0027, Latitude -2.273). The university occupies over 600 acres of rural campus in the neighbourhood of the village of Keele. The University ground includes a science park and a conference centre, making it the largest single-site campus in the UK. (The Guardian, 2015)

Accurate and sufficient data of the campus is the basis for local mobile routing services together with POI queries. There are four major ways to obtain local spatial data and associated attribute data:

- From current data available for LBS
- From professional survey companies or agencies
- From free map providers
- Collect the data by own field survey or by crowd sourcing

In most cases, the first two are either outdated and incomplete or too costly and time consuming to achieve in any details. The third is the most convenient and economical. However, data is often poorly represented in most popular commercial free map platforms, such as Google Maps, and Microsoft Bing Maps. Navteq and TeleAtlas, the two biggest map data companies in the world, are the map providers of those platforms and their data mainly focuses on the requirements of vehicle navigation. Figure 3-1 shows the map of Keele campus on Google Maps and Figure 3-2 the content on Microsoft Bing Maps. Only the main streets and a few POIs of the campus can be obtained from Google Maps while Microsoft Bing Maps practically contains no valuable spatial data of the campus. This poor spatial data coverage is not sufficient for establishing effective LBS routing services for this area. As purchasing commercial data is expensive, data have to be collected ourselves. OpenStreetMap provides an outstanding example of a spatial data source for this area to which we can contribute.

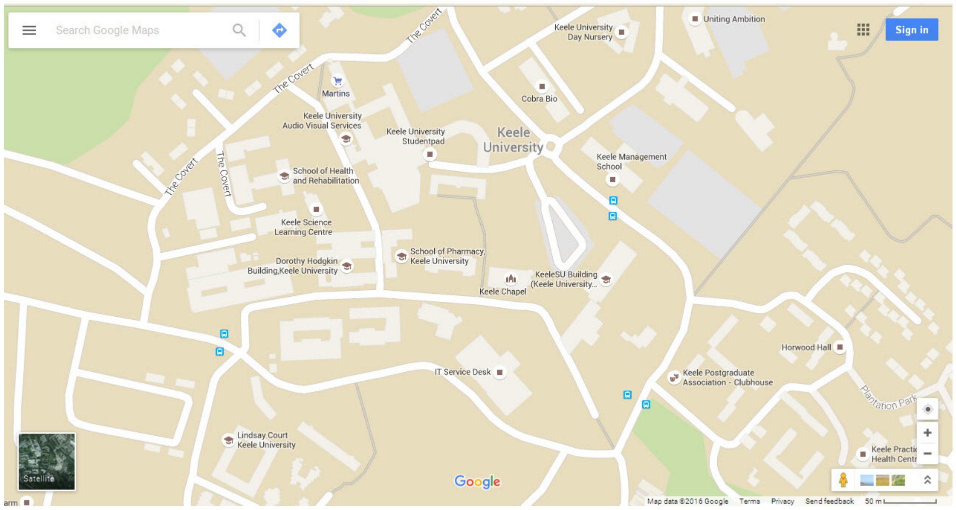


Figure 3-1 - Keele University on Google Map (Google Maps, 2016)

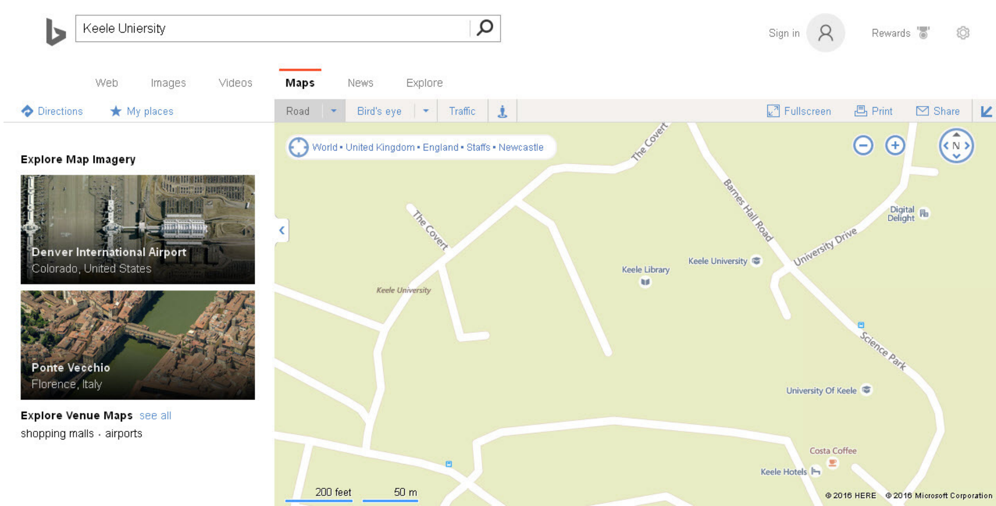


Figure 3-2 - Keele University on Microsoft Bing (Bing.com, 2016)

3.2. OpenStreetMap

OpenStreetMap (OSM) was created by Steve Coast in the UK in 2004. It is a free editable map of the entire world, with many design aspects inspired by Wikipedia. OSM allows users to view, edit and use geographical data in a collaborative way from and for anywhere on Earth. The application is made available under the Open Database License (ODbL), which enables registered users (Wiki.openstreetmap.org, 2016) to edit the map by contributing mapping data collected using manual survey, GPS devices, aerial photography, and other sources. OpenStreetMap is a powerful tool for creating and sharing map information. Over two million registered users across the world can contribute data to OSM every day. It integrates useful tools for importing, editing, exporting and generating geometry from GPS trails. Users are not just users but also data generators. For map data updating purpose, there are also some offline editing tools, such as JOSM and OSM2Go, all of which are free and user-friendly.

More and more organizations and individuals provide APIs using the OSM data, such as for map rendering and routing. The reasons for selecting OpenStreetMap as a platform for representation can be summarised as:

Totally free - All data are generated by the public with little usage restrictions and no cost. The OSM tools and APIs are powerful but are also free.

Multiple outputs - It is possible to render various popular formats of data of the same area from the OSM dataset hence giving flexibility of use.

More vivid map data with various attributes - OSM provides the public with a set of powerful tools to render own style OSM data for personal map based applications.

More current data - OSM data is being updated constantly. OSM also has a mechanism for users to update local data.

Easy database management - OSM data can be stored and managed easily and efficiently using PostgreSQL/PostGIS, a powerful open-source spatial database management system, making it easy to create new applications.

OpenStreetMap for Keele University is already a rich spatial model with many points, lines and polygon features. The Keele campus has been mapped extensively to almost its entirety and thus usable for creating this prototype to demonstrate how we can bring together the capabilities of a personalized Pedestrian Navigation System using a Maps API with the OSM providing the mapping to create a campus guidance system. Figure 3-3 shows the map of Keele University Campus in OpenStreetMap.

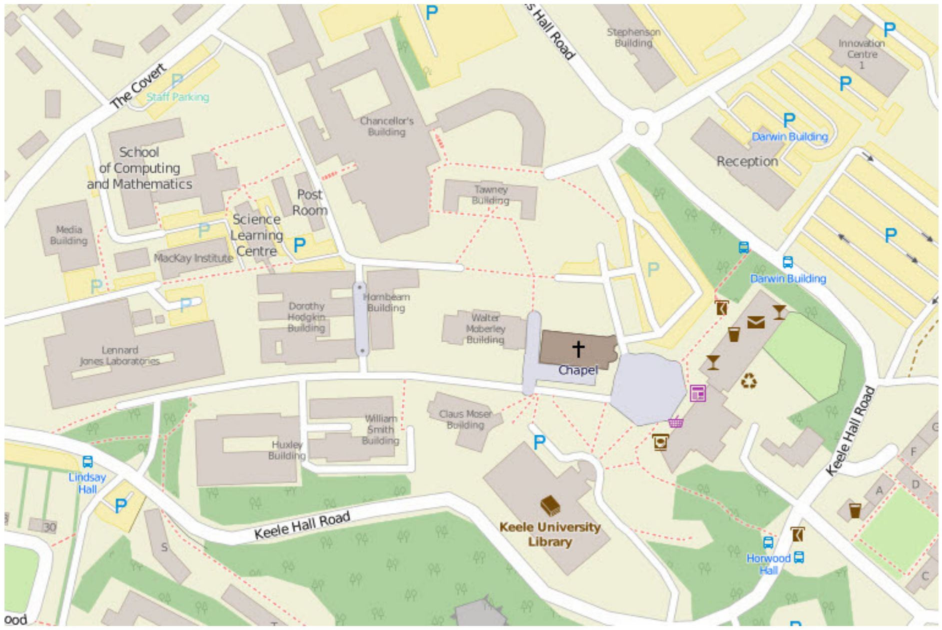


Figure 3-3 - Example of high resolution mapping of Keele University in OSM (OpenStreetMap, 2015)

3.3. MAP Rendering and Routing API

The primary output of OpenStreetMap is the data generated by the project, rather than the map itself. The geospatial data are made available for use in both offline and web applications. The process of rendering a map generally means taking the raw geospatial data and making a visual map from it, or more specifically, the production of a raster image, or a set of raster tiles. Map data can be used as an input to generate "3D rendering". One of the most exciting aspects to having open access to geospatial data is the provision to render maps in a unique and an interesting style, with highlighted features of special interest. OpenStreetMap's developer community has created a wide variety of software for rendering OpenStreetMap data. The data can also be converted to other data formats for use with existing rendering software.

3.4. Leaflet Javascript libraries

Leaflet is an Open Source JavaScript library that makes deploying maps on a web page easy. Developed by Vladimir Agafonkin, and first released in 2011 (Lovelace, 2014). It is an extensively used open source JavaScript mapping library for constructing web mapping applications, supporting most desktop and mobile platforms such as HTML5 and CSS3. The library is a collection of prewritten JavaScript that does some of the heavy liftings of scripting and creating interactive, mobile-friendly web maps. It serves as a framework for showing and interacting with map data. Interaction with the library is through its documented API. It allows developers to easily display tiled web maps hosted on a public server, with optional tiled overlays. Feature data can be loaded from GeoJSON files and styled creatively. Interactive layers such as markers with popups can be made. It has a small file size but is packed with useful features and can be extended even further with plugins.

Leaflet is the newest mapping API, which has gained popularity quickly because it works well with both mobile and desktop applications, and is small and fast. The library is not as powerful as some other APIs, but its flexibility has earned its popularity. The powerful object model makes it easy to customize existing features to various needs. Leaflet allows access to tile servers using templates, which means It can be used with almost any online map including proprietary tile servers. Figure 3-4 shows a simple code base for deploying a map of Keele University Campus (Figure 3-5) by embedding the Leaflet JavaScript library into the web page.

```
<!DOCTYPE html>
<html>

    <head>
        <link rel="stylesheet" type="text/css" href="leaflet/leaflet.css" />
        <script src="leaflet/leaflet.js"></script>
        <style>
            html, body, #map {
                height: 100%;
            }
            body {
                padding: 0;
                margin: 0;
            }
        </style>
        <title>Simple Leaflet map</title>
    </head>
    <body>
        <div id="map"></div>
        <script type="text/javascript">
            var map = L.map('map', {
                center: [53.0033918, -2.2765526],
                zoom: 16
            });
            L.tileLayer('http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
                attribution: '&copy; OpenStreetMap contributors'
            }).addTo(map);
        </script>
    </body>

</html>
```

Figure 3-4 - A Basic code for render a map of Keele University.

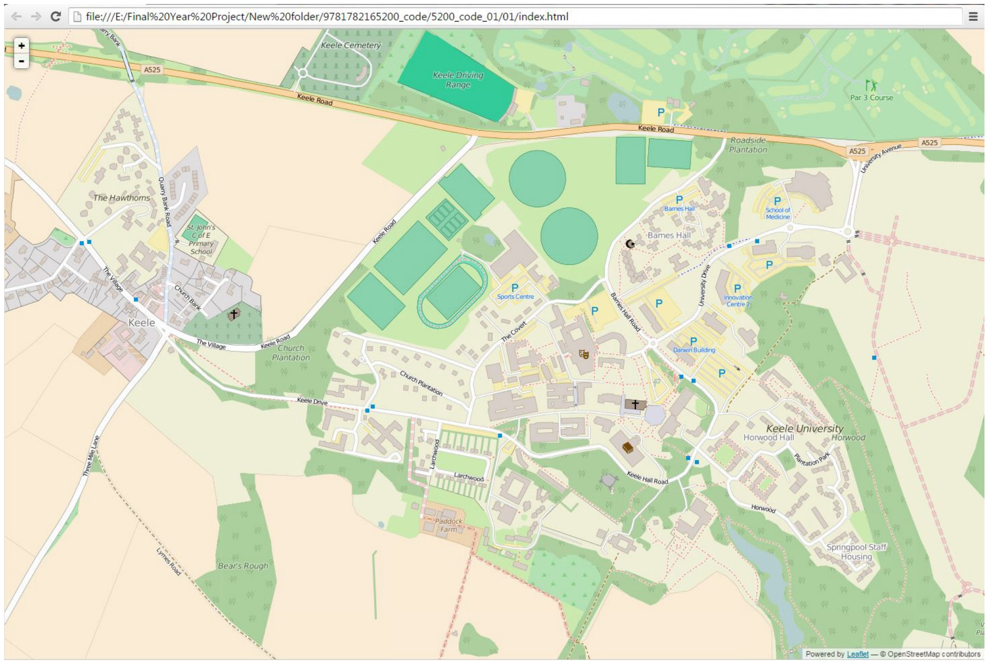


Figure 3-5 - The map of Keele University campus shown using Leaflet.

3.5. Routing Engines

There are two typical software development strategies to construct routing procedures namely third party APIs and implementing original routing algorithms. Third party APIs is convenient and fast for developers to deploy applications. However, if there are some special requirements, such as preferences to pass through buildings as far as possible because it is raining heavily, it becomes necessary to build more parameterised routing modules.

There are several third-party routing APIs, which could be used with OSM data, such as Open Source Routing Machine (OSRM) (http://project-osrm.org/), Mapzen Valhalla (https://mapzen.com/projects/turn-by-turn/) and GraphHopper (https://graphhopper.com/) [1]. All of these are open source routing engines that use routing algorithms such as the traditional Dijkstra, A*, Shooting Star and Travelling Salesman Problem (TSP). They provide car, foot and bicycle routing modes for users, with turn-by-turn direction descriptions. Leaflet Routing Machine (http://www.liedman.net/leaflet-routing-machine/) is the default Leaflet plugin for adding full functional routing capabilities to a Leaflet map. Using the default is convenient, flexible and extensible. Almost every aspect of the user interface and interactions can be customized. Leaflet Routing Machine uses OSRM as routing engine by default, but other engines can be implemented via plugins.

4. Development, Tests and Evaluation

4.1. System Architecture

Figure 4-1 shows the system architecture of the proposed campus navigation system. The underlying spatial data is on an OpenStreetMap database server, and it provides map services through the OSM API. The Leaflet Javascript Library and API deploy mobile-friendly interactive maps across all major desktop and mobile platforms, which can be extended with routing plugins and APIs, providing robust routing algorithm for both web and mobile applications based on OSM data.

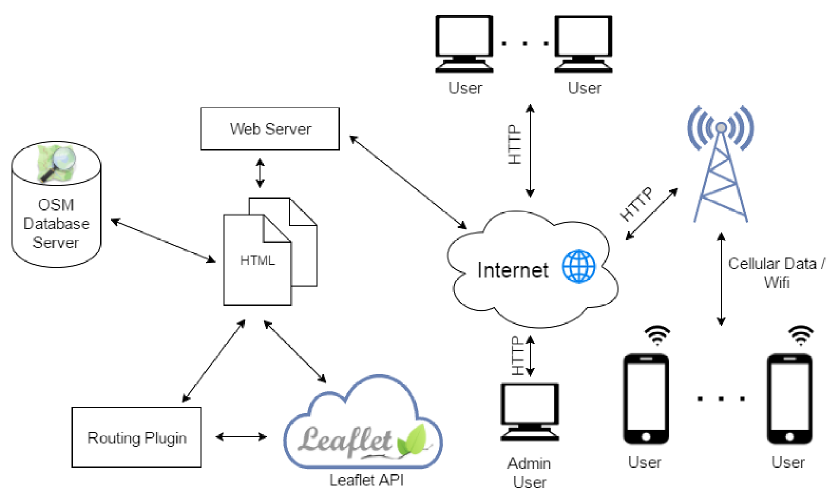


Figure 4-1 - System architecture for the campus navigation system.

4.2. Map Data Creation

Although three different navigation modes (car, walking and bicycle) can be provided by the routing engines, only pedestrian navigation would be considered around the campus. Most students and visitors will be travelling on foot between various buildings during their visits.

KEELE CAMPUS BUILDING INDEX

Buildings & Facilities
Art Gallery (2 - F5)
Bank (8 - H4)
Barnes Hall (H7)
Bookshop (8 - H4)
Chancellor's Building (2 - F5)
Chapel (7 - G4)
Church Plantation housing (C5 etc)
Claus Moser Research Centre (10 - G4)
Clinical Education Centre (Hospital campus)
Clock House (14 - G1)
Colin Reeves Building (24 - E5)
Covert housing (around D6)
David Weatherall Building (28 - L9)
Darwin Building (1 - J5)
Dorothy Hodgkin Building (18 - F4)
Exhibition Suite (2 - F2)
Guy Hilton Research Centre (Hospital campus)
Harrowby (Hall) (in F2)
Hartshill Campus (Hospital campus)
Hawthorns Hall (see inset)
Hawthorns Restaurant (38 - inset)
Health Centre (33 - L2)
Hexagon (Lindsay Hall) (E2)
Holly Cross (34 - B3)
Home Farm Buildings (39 & 40 in L7)
Hornbeam Building (19 - F4)
Horwood Hall (L3)
Horwood housing (L2)
Huxley Building (16 - E3)
Innovation Centre IC1 (30 - J6)
Innovation Centre IC2 (29 - J7)
Innovation Centres IC3 & 4 (27,28a - K6)
Jack Ashley Building (22 - E5)
Keele Hall (12 - H1)
Keele Management Centre (37 - inset)
Larchwood housing (D3)
Lennard-Jones Labs. (17 - E4)
Library and IT Services (11 - G3)
Lindsay Hall (D2)
Lindsay Studio 2 (41 - D2)
Mackay Building (25 - E5)
MacKay Institute (22 - E5)
Media Building (23 - D5)
Med IC 3 & 4 (27,28a - L7)
Medical School (Keele Campus) (28 - L9)
Medical School (Hospital Campus)
Novacentre (39 in J7)
Oaks (35 - A3)
Observatory (in L6)
Plantation Park housing (in L2)
Reception (2 - F5)
Science Learning Centre (21 - F5)
Shops (8 - H4)
Sneyd (Hall) (E3)
Sports Centre (3 - E7)
Springpool housing (L1)
Stephenson Building (31 - H6)
Students' Union (8 - H4)
Sustainability Hub (40 - L7)
Taylor House (E1)
Tawney Building (5 - G5)
Templar House (see inset)
Walter Moberly Building (6 - G4)
Westminster Theatre (2 - F5)
William Emes Building (4 - F7)
William Smith Building (15 - F4)

RESTAURANTS AND FOOD OUTLETS:

The Refectory (2 - F5)
Hawthorns Restaurant (inset)
Chacellor's Cafe (2 - F5)
Lindsay Cafe Bar and Shop (E2)
David Weatherall Cafe (28 - L9)
CB Grab and Go (2 - F5)
Street Food at The Atrium (2 - F5)
ST5 takeaway (8 - H4)
Students' Union shop and catering (8 - H4)

Figure 4-2 - The Official list of building and facilities in Keele University. (Keele University, 2016)

The first stage of data collection and mapping involved preparing the line features for roads and footpaths within the Keele Campus. The more detailed Keele Campus containing all the buildings and other points of interest(POI) was done using the OSM web-based editing tool. Various campus buildings and relevant POI's are initially identified. The list is based on the location of buildings from the official Keele campus map (Figure 4-2) and the relevant POIs like bus stops, post office, etc. The data collection phase of the project has created a spatially rich OSM map of Keele. Although the Keele University campus has been mapped quite extensively to almost its entirety in OpenStreetMap, with information gathered from different sources like Google Map's satellite view and personal knowledge of the campus, many data like footpaths rendered in OpenStreetMap are outdated or incorrect. Figure 4-3 shows a part of Keele campus near the Student Union, which is mostly outdated and is missing many components like footpaths.

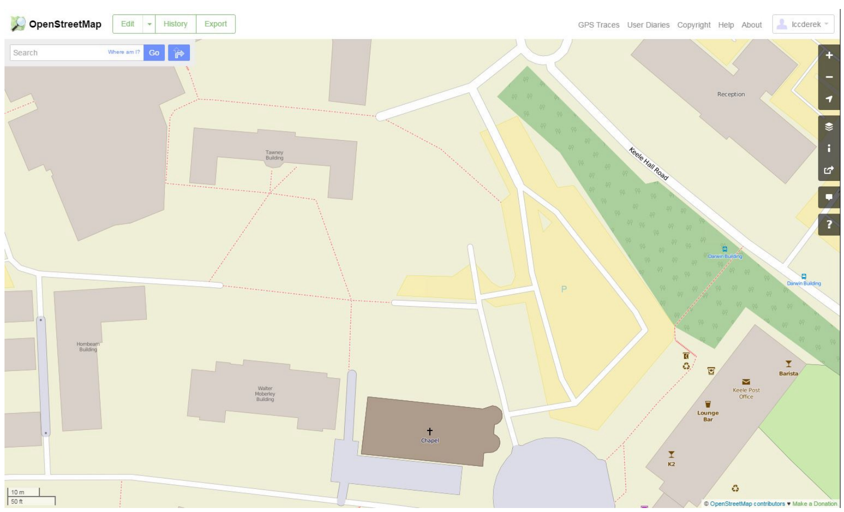


Figure 4-3 - Outdated POIs and missing footpaths in OSM before the edit.

A web-based editor based on aerial imagery is provided by OSM, making map data editing and updating very easy. New components can be created and added to the map image with three essential tools; points, lines, and area. Figure 4-4 shows the OSM map editor. After saving of the edited map, OSM took a few minutes to render the new map and made data available for public usage. Figure 4-4 shows the edited map presented to the public view.

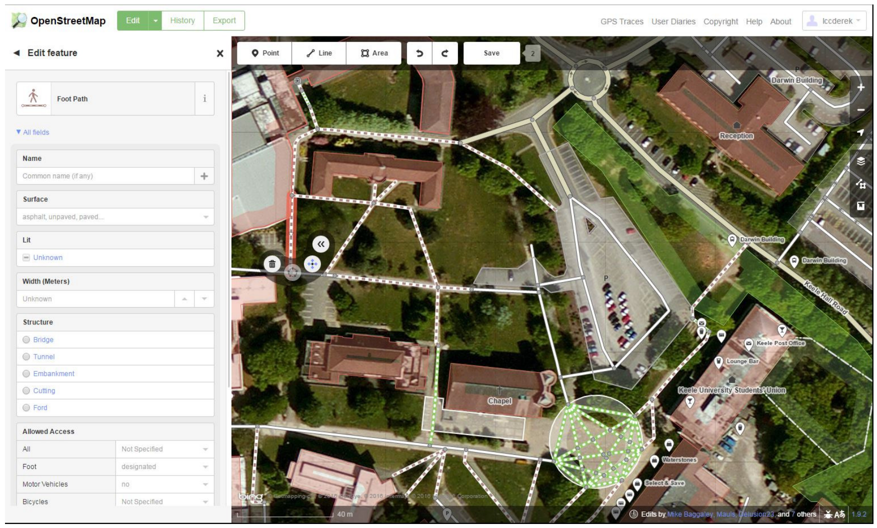


Figure 4-4 - The OSM map editor, editing the area around the Keele Student Union.

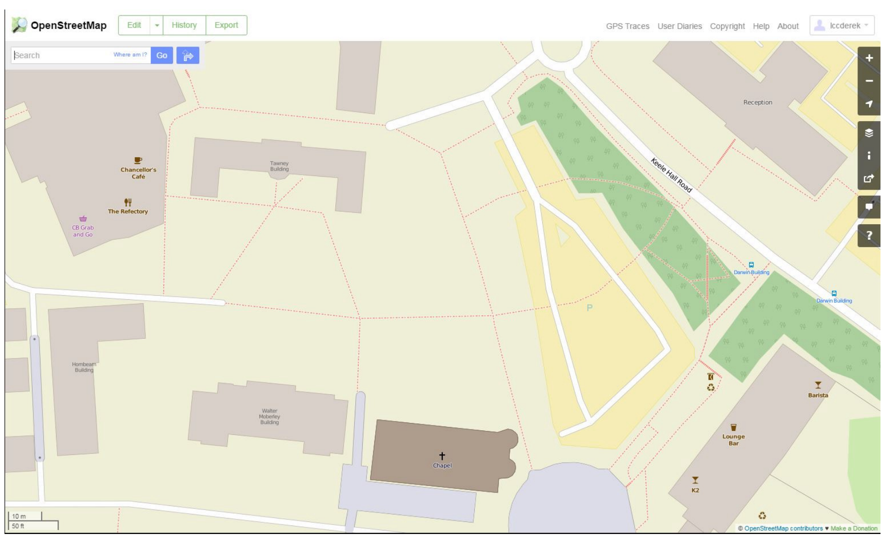


Figure 4-5 - The area around the Keele Student Union after edit, rendered and published in OSM.

4.3. Rendering Mapping data on web page

To start building a web-based navigation system with a map rendered in a web page, a map instance must be created, the simplest way is to embed the javascript as shown in Figure 3-4. However, when more map rendering and control properties needed to be defined or plugins are used, a better way is to include a separate javascript file(../map/map.js in this project) to create and set the map instance(Figure 4-6). Within the map instance, the properties of the map can be configured. For example the initial center of the map, maximum, and minimum zoom level and layers, shown in Figure 4-7.

```
<!-- Leaflet core -->
<link rel="stylesheet" type="text/css" href="http://cdn.leafletjs.com/leaflet/v0.7.7/leaflet.css" />

<script type='text/javascript' src="http://cdn.leafletjs.com/leaflet/v0.7.7/leaflet.js"></script>

<!-- Main JS for the Leaflet map -->
<script type='text/javascript' src="map/map.js"></script>
```

Figure 4-6 - Including an additional JavaScript file besides the Leaflet library.

```
//set all the layers needed
var standard = L.tileLayer('http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {id: 'OSM Standard', attribution: '&copy; <a href="http://osm.org/copyright" title="OpenStreetMap" target="_blank">OpenStreetMap</a> contributors|Routing By <a href="https://mapzen.com/" title="Mapzen" target="_blank">Mapzen</a>'}),
    grayscale = L.tileLayer('http://{s}.tiles.wmflabs.org/bw-mapnik/{z}/{x}/{y}.png', {id: 'OpenStreetMap.BlackAndWhite', attribution: '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>|Routing By <a href="https://mapzen.com/" title="Mapzen" target="_blank">Mapzen</a>'}),
    dark   = L.tileLayer('http://{s}.tile.thunderforest.com/transport-dark/{z}/{x}/{y}.png', {id: 'Thunderforest.TransportDark', attribution: '&copy; <a href="http://www.thunderforest.com/">Thunderforest</a>, &copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>|Routing By <a href="https://mapzen.com/" title="Mapzen" target="_blank">Mapzen</a>'});

//set the boundary of the map
var bounds = L.latLngBounds(L.latLng(52.994024, -2.312167), L.latLng(53.009881, -2.258690));

//properties of the map
var map = L.map( 'map', {
    center: [53.0027539, -2.2751052], //inital centre of the map
    maxBounds: bounds,  //maximunm boundary of the map
    maxZoom: 18,  //maximum zoom level
    minZoom: 15,  //minimum zoom level
    layers: [standard]  //the inital layer
});

//sets the map view to fit only within bounary
map.fitBounds(bounds);
```

Figure 4-7 - Creating and setting the map in JavaScript (/map/map.js), with additional setting a boundary for the map.

Also shown in Figure 4-6, is the variable “bounds” being created with two sets of latitude and longitude, which are the location of the southwest and northeast point of where the map’s boundary, which will be set with map.fitBounds(bounds), such that the map will only be rendered and loaded within the boundaries.

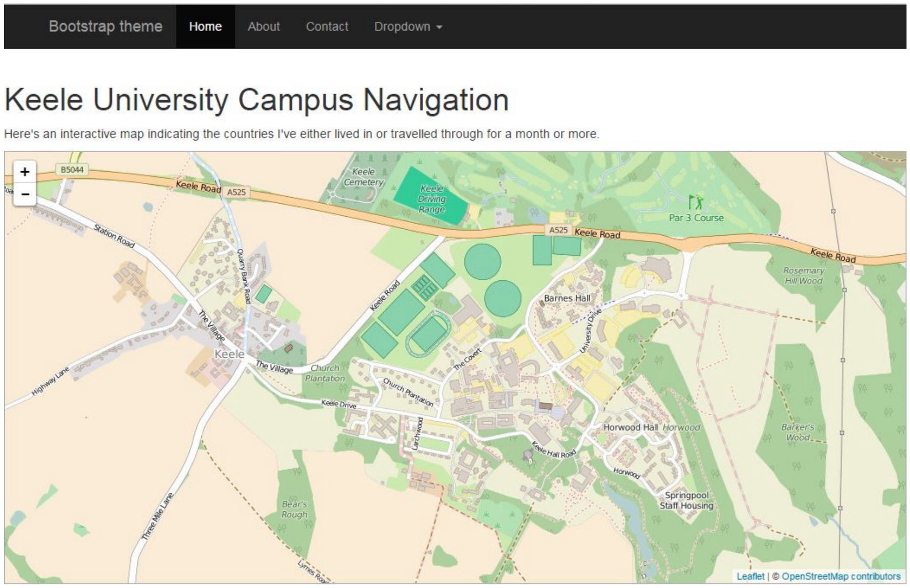


Figure 4-8 - The first edition of the webpage with a map added and some styling using Bootstrap's template for the navigation bar on top.

After adding a map to the page using Leaflet, plugins can be included and added to the map instance. In this project's case, a routing plugin is needed, which will be Leaflet Routing Machine to add basic routing to a Leaflet map, as shown in Figure 4-8.

```
<!-- Leaflet routing machine -->
<link rel="stylesheet" type="text/css" href="map/routing_machine/leaflet-routing-machine.css" />

<script type='text/javascript' src="map/routing_machine/leaflet-routing-machine.js"></script>
```

Figure 4-9 - A code snippet for including Leaflet Routing Machine into the page.

To add routing function into the leaflet map instance, L.Routing.control is used, which is the main class of the leaflet routing machine plugin (Figure 4-9), presenting a full routing user interface.

A geocoder should also be added since the routing software can only route between location by their latitudes and longitudes and not every user would know the exact latitude and longitude of a location, a geocoder is needed for the software to look up the coordinate of address. A geocoder can be used in reverse geocoding, which is to look-up an address with a waypoint. The author of Leaflet Routing Machine also created a simple geocoder for Leaflet that would work stably with the Leaflet Routing Machine, and can be extended easily to multiple data providers while uses Nominatim by default.

Figure 4-10 shows the code for setting the waypoints, geocoder and other options of the router using L.Routing.control, these options includes “position”, reverseWaypoints” and “collapsible”, which are to set the position of the routing interface, “reverseWaypoints” is set to true so that a button is shown for the user to reverse the order of the waypoints and “collapsible” is set to true such that a collapse button is alway indicated for the user to hide the routing interface.

```
//Routing
var control = L.Routing.control({
  position: 'topleft',

  waypoints: [
    L.latLng(setFrLat(),setFrLng()),
    L.latLng(setToLat(),setToLng())
  ],
  //geocoder
  geocoder: L.Control.Geocoder.nominatim(),

  //other settings
  reverseWaypoints: true,
  collapsible: true,
}).addTo(map);
```

Figure 4-10 - Adding the routing function into the leaflet map with L.Routing.control, and setting the geocoder and other options for the router.

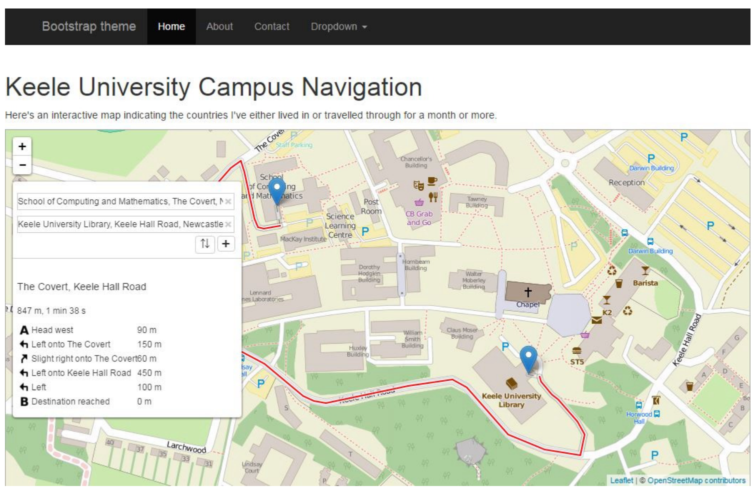


Figure 4-11 - Basic routing using Leaflet Routing Machine, with geocoder added for searching location address.

4.4. Pedestrian mode Navigation

The leaflet routing API computes the shortest path between two buildings on the campus; this depends on the richness of the OSM map of Keele campus and whether the routing API can find routes on external footpaths, lanes, streets, and internal corridors. The exact nature of the path is determined by the setting of the routing mode option in the route finding algorithm in the API.

Pedestrian navigation differs in many ways from the conventional navigation system used in a car. Key differences between these two systems are worked out and adapted for pedestrian use. A route on the campus of the Keele University can be long and confusing for someone not familiar with the campus. The route has to be simplified and broken down into smaller segments for easy overview. The route segments should not be longer than a few hundred meters, given the low traveling speed of a pedestrian. With smaller segments it would be easier for a user to track the progress.

With the core routing system set up using the Leaflet Routing Machine plugin, the system already has the functionality of a simple web-based routing application and capable of providing a route from A to B, with turn-by-turn instruction. However, Figure 4-12 shows the resulting route for traveling from Tawney Building to University Library on the campus. The resulting path, in reality, is a path for vehicles. Since the Leaflet Routing Machine queries Open Source Routing Machine (OSRM)'s public servers by default, which only have support for car routing.

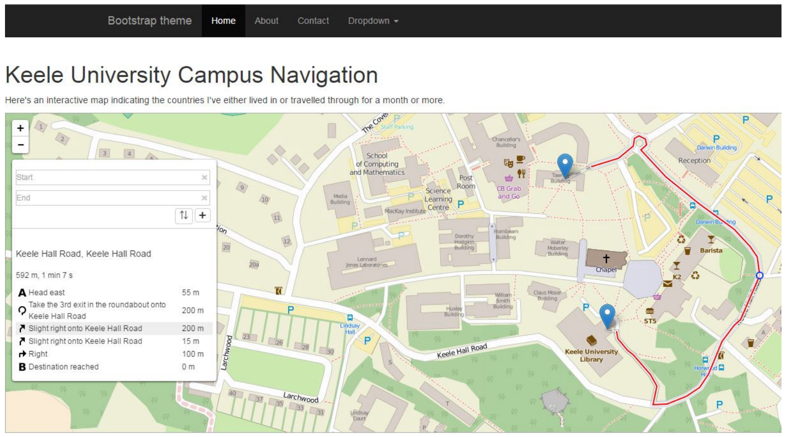


Figure 4-12 - Using the default setup of Leaflet routing machine to routing from the Tawney Building to the library. Output routing will be a path routed for vehicles

Although OSRM does have support for pedestrian navigation, a separate server would need to be setup which would be very time-consuming and not very efficient to develop within this project while other more convenient routing engines can be integrated.

Leaflet Routing machine allows change of routing engine from OSRM to another routing software like GraphHopper, Mapbox, and Mapzen Valhalla. It was decided that Mapzen Valhalla will be used as the routing engine for the prototype of the routing system since it is free to use, provides dynamic and customizable routes with clear directions.

To incorporate Mapzen Valhalla API with Leaflet Routing Machine, it provides a simple ways of including the JavaScript and CSS file into the web page and changing the router for the Routing Machine within L.Routing.control, as shown in Figure 4-13.

```
lineOptions: {
  styles: [
    {color: 'white',opacity: 0.8, weight: 10},
    {color: '#2676C6', opacity: 1, weight: 4}
  ]
},

//router
//mapzen
router: L.Routing.mapzen('valhalla-SbgByba', 'pedestrian'),
formatter: new L.Routing.Mapzen.Formatter(),
```

Figure 4-13 - Changing the router for Leaflet Routing Machine to Mapzen Valhalla in the L.Routing.control class in “map.js”. While also setting the style of the path.

In addition to changing the router for Leaflet Routing Machine to Mapzen Valhalla using the “router” option, Figure 4-13 also shown setting the style of the line using the “lineOption” option, with the colour, opacity, and weight of the resulting path of the route.

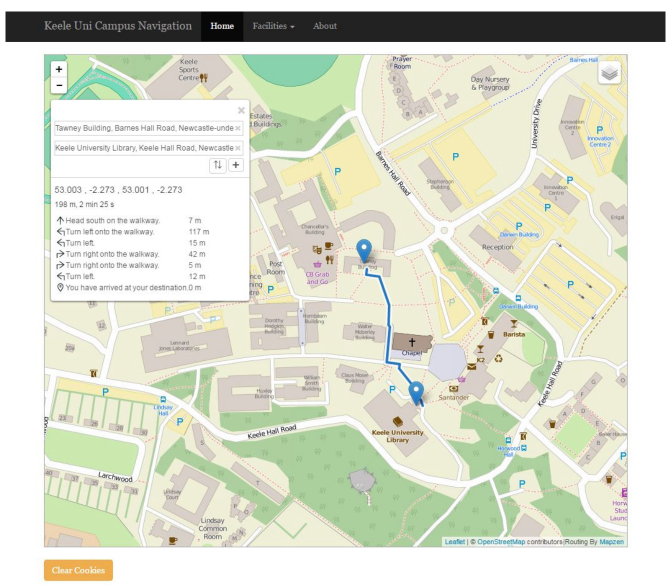


Figure 4-14 - A path routed after change the router to using Mapzen Valhalla’s pedestrian routing mode.

4.5. User Interface

A simple working prototype website was made up for testing the web-based navigation system (Figure 4-15). A list of POI included in a separate page for users to choose as either their starting point or destination of the navigation.(Figure 4-16).

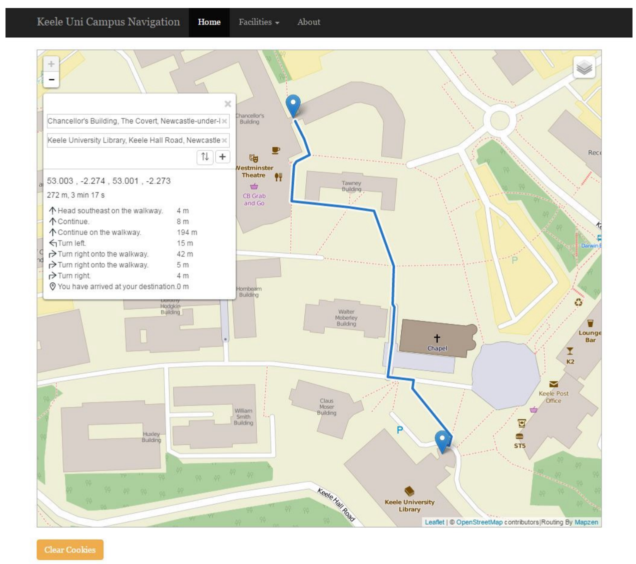


Figure 4-15 - An example of the routing from the Chancellor's building to the library using the navigation system.

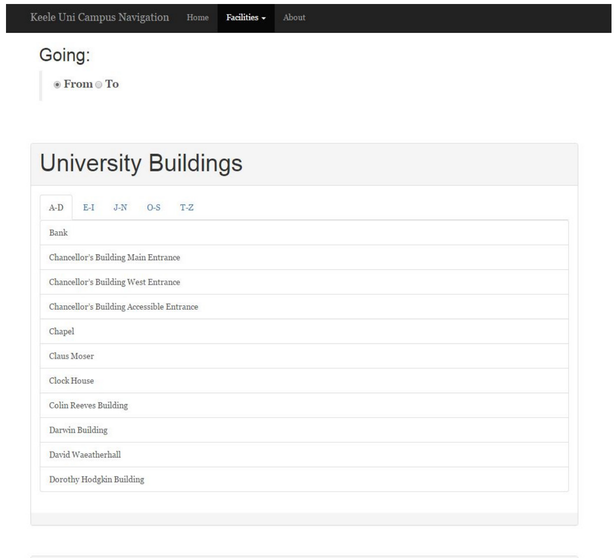


Figure 4-16 - The list of buildings and facilities in the Keele campus for the user to choose from.

One of the primary objectives of this project is to create a web-based navigation system allowing the user to use the system on either a mobile device or a PC; the website must be designed to be responsive and adaptive. There are multiple ways to create a responsive website; one is to build it from scratch, to define all the styling and responsiveness of the web page. Another is to use existing open source framework for developing responsive web pages in HTML,CSS and JavaScript, such as "Bootstrap".

The prototype created in this project also using Bootstrap as a basic framework for it styling of the pages by using combining two of the template Bootstrap provided. Figure 4-17 shows the prototype opened in a mobile browser.

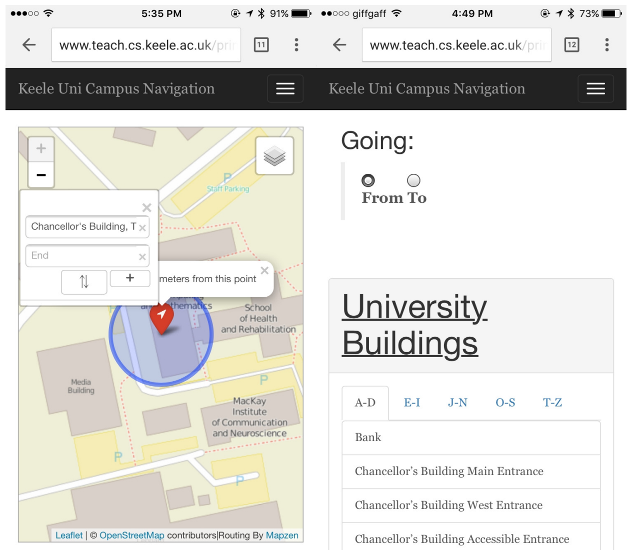


Figure 4-17 - The prototype displaying responsively when opened in a mobile browser.

In this prototype, the user can browse through the campus facilities and select a location in a list where they want to travel to or from. The user can also select any location on the map by clicking a point on the map. (Figure 4-18).

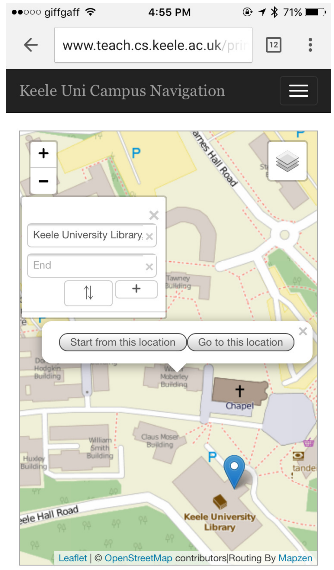


Figure 4-18 - The user clicked on the map and select the location as start or destination.

By adding a popup when the map is clicked using Leaflet's L.popup() function, and creating two buttons for the user to choose from inside the popup, as shown in Figure 4-19. To replace the waypoints in the router, control.spliceWaypoints method should be used, as it would select a waypoint, start or end, and replace it with a new waypoint which is where the user have clicked.

```
//pop up when the user click on the map
map.on('click', function(e) {
  var container = L.DomUtil.create('div'),
    startBtn = createButton('Start from this location', container),
    destBtn = createButton('Go to this location', container);

  L.popup()
    .setContent(container)
    .setLatLng(e.latlng)
    .openOn(map);

  L.DomEvent.on(startBtn, 'click', function() {
    control.spliceWaypoints(0, 1, e.latlng);
    map.closePopup();
  });

  L.DomEvent.on(destBtn, 'click', function() {
    control.spliceWaypoints(control.getWaypoints().length - 1, 1, e.latlng);
    map.closePopup();
  });
});

//create a button
function createButton(label, container) {
  var btn = L.DomUtil.create('button', '', container);
  btn.setAttribute('type', 'button');
  btn.innerHTML = label;
  return btn;
}
```

Figure 4-19 - A code snippets from “map.js” for adding a popup when the user click on the map for the user to select that location as a starting point or destination.

The user’s current location is also important to be displayed on the map since the user would want to know where they currently are in regards to their starting point, the destination, and the path to take. Figure 4-20 shows the mobile view of the prototype website where it marks the user’s estimate currently location.

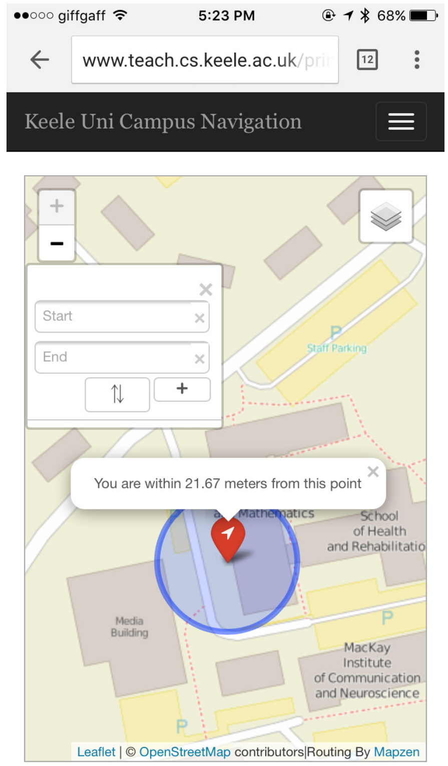


Figure 4-20 - The user's estimated current location marked.

For the map to detect the user's estimate location, Geolocation need to be used by using Leaflet's locate() method with the "setView" option, shown in Figure 4-21, which is to set the map view to display the user's location. To further show the user's location on the map, a marker should be added, using the Leaflet's L.marker() method, with a binded pop up to describe the marker. Since the user's position get by Geolocation is not 100% accurate, the accuracy of the marker should also be indicated, by using the L.circle() method a circle would be placed on the map

With the appropriate position of the marker and the radius of the accuracy, the estimated location of the user would be indicated on the map.

```
//locate the user's location and mark it
function onLocationFound(e) {
    var radius = e.accuracy / 3;

    var redLocMarker = L.AwesomeMarkers.icon({
      icon: 'location-arrow',
      markerColor: 'red',
      prefix: 'fa'
    });

    L.marker(e.latlng, {icon: redLocMarker}).addTo(map)
        .bindPopup("You are within " + radius.toFixed(2) + " meters from this point")
          .openPopup();

    L.circle(e.latlng, radius, {clickable:false}).addTo(map);
}

map.on('locationfound', onLocationFound);

map.locate({
  setView: true
});
```

Figure 4-21 - Code snippets from “map.js” to get the user’s location and marking it.

Multiple tile layers are also integrated into the prototype, with three different layers that the user can choose from such that the user can change how the map is displayed according to their need (Figure 4-22). The user has the option to minimize the routing interface so that if the user interface for the routing function is to block the user from viewing the map, the user can collapse the routing interface and view the map without any obstruction (Figure 4-22).

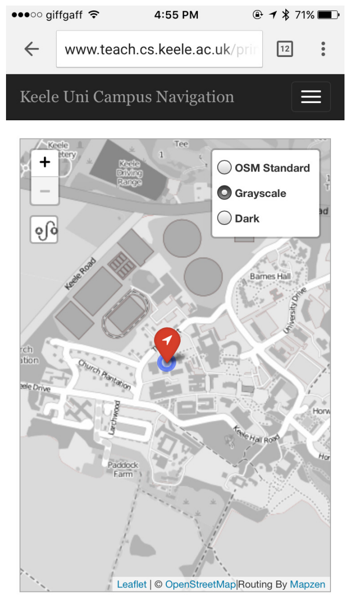


Figure 4-22 - The prototype website viewed in mobile, with the grayscale layer selected and the routing interface closed.

To add multiple tile layers into the map, a Layers Control must be added into the map. With the variables “standard”, ”grayscale” and “dark” set as all the different tile layers (Figure 4-7), and containing all the tile layers that the map will display into a single variable, “baseMaps”, and add it to the Layers Control with L.control.layers(). (Figure 4-23)

```
//base layers, all the layers that can be choose by the user
var baseMaps = {
    "OSM Standard": standard,
    "Grayscale": grayscale,
    "Dark": dark,
};

//add the layers to the map
L.control.layers(baseMaps).addTo(map);
```

Figure 4-23 - Code snippets from “map.js” for adding the tile layer into the map.

5. Problems and Further Work

The employment of open geospatial technologies such as Leaflet API and OSM data is demonstrated in this project, revealing its use in creating a cost-efficient application. The system can easily be deployed as a web application which can be used on mobile devices as well computers. It allows users to plan shortest routes between the specified set of buildings and POI. Users can also drag a marker and place it at any point on the campus map from which they want to generate the shortest path.

The prototype currently uses 2D maps for the routing system, with on-screen turn-by-turn instructions as guidance. It is possible to add geotagged images onto the map to provide geovisual assistance. Ultimately, the campus navigation system should be context aware and make the user interface reusable by changing the schema and by making it customizable by any department in the university. The system dynamically calculates the new shortest path and updates the map display accordingly.

5.1. Problems to be Solved

5.1.1. Indoor Mapping

There is two main consideration that defines important waypoints on a campus route on which it would be useful to divide the path into different segments:

- Entering or leaving a building
- Landmarks such as sculptures, big signs

Consideration of height change can be criteria for future enhancement of the project. On a 2D map, it is not possible to display more than one floor at the same time; the path must be split into multi-level if there are more that one level in a building. The route could be segmented at every elevator or staircase. Also, upon entering a building, a pedestrian's entire perception and orientation may change. To reflect the user’s perception of change the view on his mobile device should also change. For example, the map view should be modified to that of an indoor map corresponding to the level the pedestrian is travelling on, instead of the outdoor map view, which only show the ground level map.

OSM’s map editor, in fact, has support for indoor mapping. OSM default does not display indoor mapping data as shown in Figure 5-2. To view the indoor map, other Overpass renderer API, such as OpenLevelUp, may be required. In Figure 5-3, some area in the

Colin Reeves Building and the Chancellor's Building were displayed with the indoor mappings.

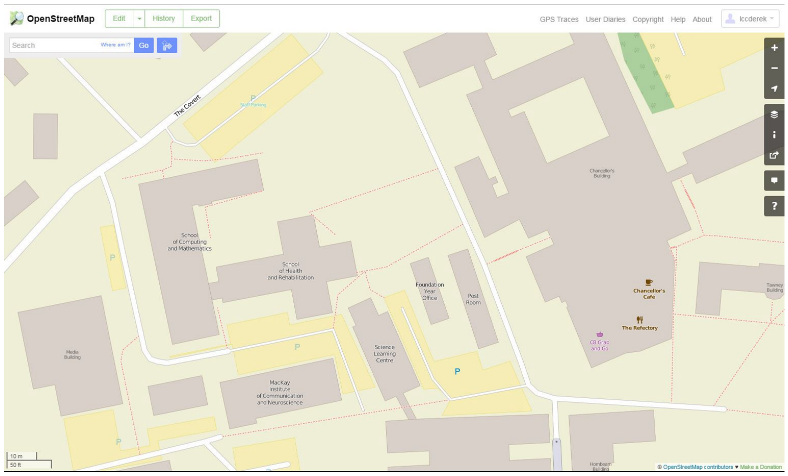


Figure 5-1 - OSM standard display without indoor mapping.

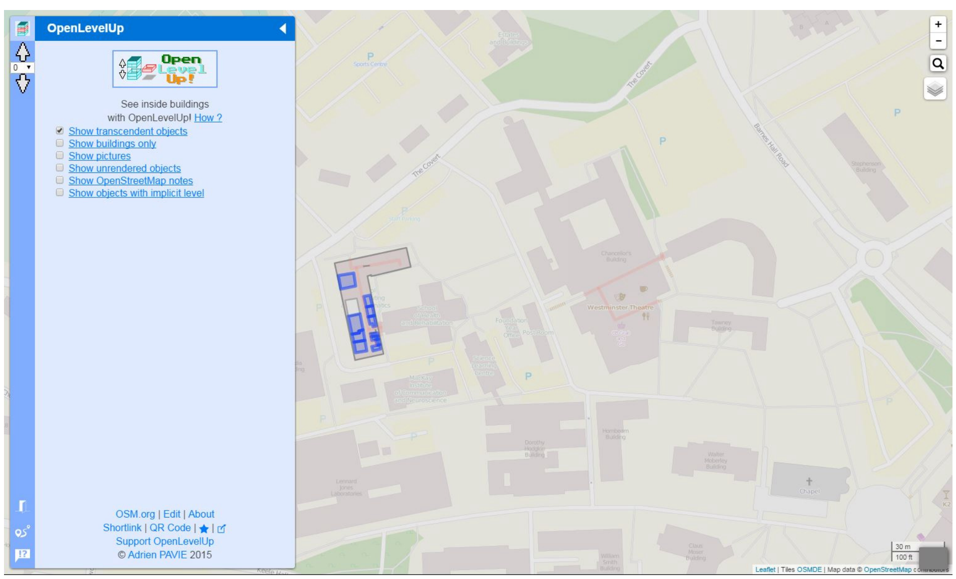


Figure 5-2 - Indoor Mapping displayed using an Overpass API based renderer(OpenLevelUp).

5.1.2. Indoor Routing

Full indoor routing for the campus and delivery of information may require extensive indoor mapping data. However, central indoor corridors could be added as paths in OSM data, so that the routine may compute the shortest path through a building in the campus, which will significantly improve the practically and performance of the navigation system.

5.2. urther Work

5.2.1. Geotagged Images and Photos.

Buildings and POI identified in and around the campus that would be of relevance to the navigation system. For people new to the campus the use of landmarks and geotagged images for navigation is of great assistance. Figure 5-1 shows an example of such a view, where a geotagged photo of an intersection was inserted with a marker marking the place where a change of travel direction was needed. The visual clue will help the user with orientation and assist them in the right direction when taking the suggested route.

Figure 5-3 - Geotagged Images.

5.2.2. Transportation and events information options.

There are always needs to provide the user with events, travel, and transport related information. Figure 5-4 shows an information window inserted with a marker at the transit point, where information such as bus timetables could be provided to the user. Offering users with options to view extended information on how to travel to and from the Keele campus. Other information that could be given at various markers locations includes classes information, special events and notifications, timetables and links to all the important websites that users may be interested in on the route they would be taking. This feature should assist and save users a lot of time by providing them with targeted campus information.

Figure 5-4 - A marker used in a bus stop with the transport information.

5.2.3. Context Aware Routing

An information system that helps users getting usable and updates on campus, such as special campus events, seminars based on users profile information such as personal interests, faculty, research area etc. Students could be reminded with examination timetables, lecture rooms. Visitors could receive invitations to seminars and meetings recommend places to go such as meeting and seminars etc.

6. Conclusion

The project goal was to develop a web application as a university campus guidance and navigation system. The main requirements of the project are (a) highly efficient system development and easy maintenance, (b) effective and useful data and services and (c) economical. Much potential could be seen in how customizable the use of OSM data using APIs could be. The use of open geospatial technologies like Leaflet API and OSM data had been demonstrated, as well as their use in creating cost-efficient applications. The web-based application approach of the campus navigation system allowed the system to be easily deployed and used on mobile devices as well as computers. Users of the system were able to plan for efficient routes within the campus by selecting from POI list, or dragging the "To" and "From" markers and place them on the campus map and generate a walking path.

Although Open Street Map(OSM) is a free source of mapping data of the entire world based on a knowledge crowdsourcing model that provides user-generated street maps, OSM is probably the most extensive and useful project currently under development and has the most comprehensive and free to use data for the Location-Based-System project. The OMS underlying spatial data allows developers the flexibility of access and the ability to use the various APIs while providing excellent user-generated street maps efficiently. OSM has a map editor that enables input of user generated content. It also provides basic GIS map functions, such as zoom in/out, pan and feature information tools. Leaflet has provided a lightweight Javascript library and maps API, which custom applications can be built on with relative ease. Open source APIs are available offering higher level GIS functions such as routing engines APIs, browser API for mobile phones reducing efforts of designing responsive web pages.

In going forward, a campus navigation system can be built, which is context aware and to make this user interface reusable by changing the schema and by making it customizable by any department in the university. Advanced information systems should deliver more than static information. The users would also benefit from extra information surrounding the campus. Examples would include personalized class information, announcements, and event information. Social media aspects, such as a friend's location, would likely be appreciated by most users. It should be possible also to recommend sights or information based on the user's context and personal interests. The user context may be simply the current time or their current location, interest major at their current position, their means of travel. The user's interests can be captured by giving feedback about items in their travel history around the campus, in user's profile, their travel history, etc.. The issue is how to integrate such contextual information into the campus navigation process.

7. References

Akanbi, C., Ogundoyin, I. & Lawal, A. 2014, "Implementing A University Mobile Navigation System", .

Bing.com. (2016). *Bing Maps*. [online] Available at: http://www.bing.com/maps/ [Accessed 27 Apr. 2016].

GeeksforGeeks. (2012). *Dijsktra's algorithm*. [online] Available at: http://www.geeksforgeeks.org/greedy-algorithms-set-6-dijkstras-shortest-path-algorithm/ [Accessed 14 Apr. 2016].

Google Developers. (2016). *Google Maps APIs | Google Developers*. [online] Available at: https://developers.google.com/maps/ [Accessed 26 Apr. 2016].

Google Maps. (2016). *Google Maps*. [online] Available at: https://www.google.com/maps [Accessed 26 Apr. 2016].

Graphhopper.com. (2016). *GraphHopper Directions API with Route Optimization*. [online] Available at: https://graphhopper.com/ [Accessed 12 Apr. 2016].

Hayakawa, K., Hayakawa, T. & Ito, T. 2014, *"Implementation of an Event Information Sharing Support System Utilizing OpenStreetMap"* in Modern Advances in Applied Intelligence Springer, , pp. 328-337.

Hlaing, H.H. & Ko, K.T. "Location-Based Recommender System for Mobile Devices on University Campus", .

Huang, J., Zhan, Y., Cui, W., Yuan, Y. & Qi, P. 2010, *"Development of a campus information navigation system based on GIS"*, Computer Design and Applications (ICCDA), 2010 International Conference onIEEE, , pp. V5.

Jacob, R., Zheng, J., Ciepłuch, B., Mooney, P. & Winstanley, A.C. 2009, *"Campus guidance system for international conferences based on OpenStreetMap"* in Web and Wireless Geographical Information Systems Springer, , pp. 187-198.

Keele.ac.uk. (2016). *About the Area - Keele University*. [online] Available at: https://www.keele.ac.uk/aboutus/aboutthearea/ [Accessed 27 Apr. 2016].

Keele.ac.uk. (2016). *About us - Keele University*. [online] Available at: https://www.keele.ac.uk/aboutus/ [Accessed 23 Apr. 2016].

Keele.ac.uk. (2016). *Keele campus map - Keele University*. [online] Available at: http://www.keele.ac.uk/kudis/keelecampusmap/ [Accessed 24 Apr. 2016].

Lautenschläger, B. 2012, *"Design and Implementation of a Campus Navigation Application with Augmented Reality for Smartphones"*, Unpublished bachelor thesis Department of Computer Science, University of Calgary, Alberta, Canada, .

Leafletjs.com. (2016). *Leaflet — an open-source JavaScript library for interactive maps*. [online] Available at: http://leafletjs.com [Accessed 20 Apr. 2016].

Liedman.net. (2016). *Leaflet Routing Machine*. [online] Available at: http://www.liedman.net/leaflet-routing-machine/ [Accessed 12 Apr. 2016].

Lovelace, R. (2014). *Testing web map APIs - Google vs OpenLayers vs Leaflet*. [online] Robinlovelace.net. Available at: http://robinlovelace.net/software/2014/03/05/webmap-test.html [Accessed 12 Apr. 2016].

Maloney, A. (2014). *Creating An Interactive Map With Leaflet and OpenStreetMap*. [online] Asmaloney.com. Available at: https://asmaloney.com/2014/01/code/creating-an-interactive-map-with-leaflet-and-openstreetmap/ [Accessed 16 Feb. 2016].

Mapzen.com. (2016). *Mapzen Turn-by-Turn · Mapzen.* [online] Available at: https://mapzen.com/projects/turn-by-turn [Accessed 12 Apr. 2016].

Microsoft.com. (2016). *Bing Maps*. [online] Available at: http://www.microsoft.com/maps/ [Accessed 26 Apr. 2016].

Nitami, R., Suzuki, A. & Murata, Y. 2014, *"Development of a pedestrian navigation system without additional infrastructures"*, Indoor Positioning and Indoor Navigation (IPIN), 2014 International Conference onIEEE, , pp. 203.

OpenStreetMap. (2015). *OpenStreetMap*. [online] Available at: http://www.openstreetmap.org/ [Accessed 20 Dec. 2015].

OpenStreetMap. (2016). *OpenStreetMap*. [online] Available at: http://www.openstreetmap.org [Accessed 26 Apr. 2016].

Panier, A. (2016). *OpenLevelUp!*. [online] Openlevelup.net. Available at: http://openlevelup.net/ [Accessed 19 Apr. 2016].

Project-osrm.org. (2016). *Project OSRM*. [online] Available at: http://project-osrm.org/ [Accessed 12 Apr. 2016].

Singal, P. & Chhillar, R. 2014, *"Dijkstra Shortest Path Algorithm using Global Position System"*, International Journal of Computer Applications, vol. 101, no. 6.

The Guardian. (2015). *University guide 2016: Keele University*. [online] Available at: http://www.theguardian.com/education/2009/may/10/universityguide-keele-uni [Accessed 23 Apr. 2016].

Thecompleteuniversityguide.co.uk. (2016). *Keele University*. [online] Available at: http://www.thecompleteuniversityguide.co.uk/keele/international [Accessed 22 Apr. 2016].

Webhelp.esri.com. (2014). *ArcGIS Explorer - Map projections*. [online] Available at: http://webhelp.esri.com/arcgisexplorer/900/en/map_projections.htm [Accessed 25 Apr. 2016].

Wiki.openstreetmap.org. (2016). *Contributors - OpenStreetMap Wiki*. [online] Available at: http://wiki.openstreetmap.org/wiki/Contributors [Accessed 26 Apr. 2016].

Zheng, J., Chen, X., Ciepłuch, B., Winstanley, A., Mooney, P. & Jacob, R. 2010, *"Mobile routing services for small towns using cloudmade API and OpenStreetMap"*, .

Appendix A - Project Plan

UG Project Plan
CSC-30013/14

Project Overview and Description

Student Name: Lee Chi Chung Derek
Student Username: v9n33
Student Number: 12024326
Module (*delete as appropriate*): CSC-30014
Supervisor Name: Mr. Dave Collins
Project Title: Mobile dynamic campus GPS turn-to-turn navigation system (Original topic: Mobile dynamic timetabling tool for public transport information)

Please provide a brief Project Description:

Produce a web-based system which provides a dynamic campus GPS navigation system and other possible information in the Keele University campus for user to navigate on foot. A web-based application of a campus map for walking navigation, using the user's current location or set locations and the destination, and provide the possible routes to go to the required destination. Other possible information may include room's timetable, special events and news in camps.

What are the aims and objectives of the Project?

- To provide a map of the campus for the user the view digitally.
- To provide the possible routes of navigating within the campus. With possible routing option like fastest, most scenic, etc.
- Provide a dynamic turn-by-turn navigation system using the GPS built-in the mobile device.
- Have a reactive website, so that user can use the website on either mobile device or pc.
- Provide latest information based on selected location, such as events and news.

Please provide a brief overview of the key literature related to the Project:

Web programing:
Nixon, R. (2014) Learning PHP, MySQL & JavaScript: With JQuery, CSS & HTML5. 4th edn. United States: O'Reilly Media, Inc, USA.
Nixon, R. (2014) Learning PHP, MySQL & JavaScript: With JQuery, CSS & HTML5. 4th edn. United States: O'Reilly Media, Inc, USA.
Wroblewski, L. (2011) Mobile First. United States: A Book Apart.
Marcotte, E. and Keith, J. (2011) Responsive web design. New York, NY: A Book Apart.
Kim, B. (2015) 'Chapter 4: Responsive web design, Discoverability, and mobile challenge', Library Technology Reports, 49(6), pp. 29–39.
Mobile GPS navigation:
He, L. and Lai, Z. (2005) 'The study and implementation of mobile GPS navigation system based on Google maps', , pp. 87–90.
Kishen, M. R. and Kumar, D. V. (2031) 'Low-cost mobile GPS tracking solution', , pp. 516–519.
Ishikawa, T., Fujiwara, H., Imai, O. and Okabe, A. (2008) 'Wayfinding with a GPS-based mobile navigation system: A comparison with maps and direct experience', Journal of Environmental Psychology, 28(1), pp. 74–82.

Project Process and Method

Please provide a brief overview of the Methodology to be used in the Project (inc. an overview of best practice within the Methodology):

In order to make the website be as user friendly as possible, multiple personas would be created, estimating different kind of potential user. With the different personas, a more user friendly website could be designed. Personas of different user groups should be included, like students, staffs and visitors.
Since an interactive map would be required to embed inside the website, the possible method of embedding may include using Google API (not suggested), Leaflet (an open-source JavaScript library), and may even be created in scratch (based on other maps).

Which Data Collection Methods will be employed (e.g card sorts, questionnaires, simulations, …)?

Questionnaire on the possible problems of some of the current available navigation system service like the already existing map feature in the Keele University mobile application and Google map navigation function. For example, readability, functionality and user-friendliness.

Time and Resource Planning

Will Standard Departmental Hardware be used? YES/~~NO~~

If NO please outline the Hardware/Materials to be used:

Will Software which is already available in department be used? YES/~~NO~~

If NO please outline the Software to be used including how any necessary licences will be obtained:

Will the project require any Programming? YES/~~NO~~

If YES please list the (potential) Programming Languages to be used (including any IDEs and Libraries you may make use of):

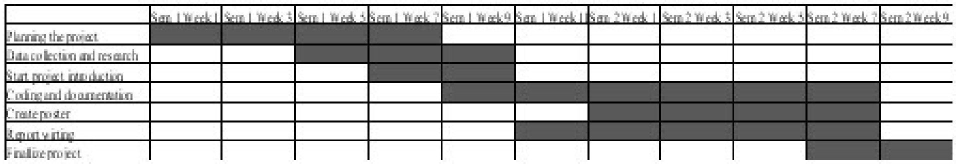
HTML5, JavaScript, CSS, PHP, SQL

Table of Risks (*if non Standard Hardware and/or Software to be used please include backup options/ contingency plans here*):

Possible Risk	Likelihood	Way(s) to avoid	How to minimize the effect of risk if raised
Incorrect data(destination) inputted	Likely	Have a pattern for the form to check against with.	Have other suggested locations for the user to choose from.
Blank data(destination) submitted	Unlikely	Have the input for the destination required in order to submit.	Have error pop-up informing the user they have not entered the destination if ever occur.
Incorrect data(current location) received from the user's GPS	Unlikely	Have an option which allows users to select the correct location if needed.	Check again with the user if the current location is correct.
User's location being compromised to unauthorized people	Unlikely	Have the data sent and received encrypted.	Consult appropriate people for further action.
Unable to access map sensors	Unlikely, but possible	Sense map GL from Keele campus senors	Switch to alternative servers automatically

3/5

Gantt Chart/ Pert Chart (must include milestones and deliverables):



References and Administration

Please include a list of References used in this Plan:

Nixon, R. (2014) Learning PHP, MySQL & JavaScript: With JQuery, CSS & HTML5. 4th edn. United States: O'ReillyMedia, Inc, USA.
Nixon, R. (2014) Learning PHP, MySQL & JavaScript: With JQuery, CSS & HTML5. 4th edn. United States: O'ReillyMedia, Inc, USA.
Wroblewski, L. (2011) Mobile First. United States: A Book Apart.
Marcotte, E. and Keith, J. (2011) Responsive web design. New York, NY: A Book Apart.
Kim, B. (2015) 'Chapter 4: Responsive web design, Discoverability, and mobile challenge', Library Technology Reports, 49(6), pp. 29–39.
He, L. and Lai, Z. (2005) 'The study and implementation of mobile GPS navigation system based on Google maps', , pp. 87–90.
Kishen, M. R. and Kumar, D. V. (2031) 'Low-cost mobile GPS tracking solution', , pp. 516–519.
Ishikawa, T., Fujiwara, H., Imai, O. and Okabe, A. (2008) 'Wayfinding with a GPS-based mobile navigation system: A comparison with maps and direct experience', Journal of Environmental Psychology, 28(1), pp. 74–82.

Submission Date:

PLEASE NOTE THAT SHOULD YOUR PROJECT UNDERGO ANY MAJOR CHANGES FOLLOWING THE SUBMISSION OF THIS PLAN YOU ARE EXPECTED TO SUBMIT AN UPDATED PLAN WHICH ACCURATLEY REFLECTS YOUR PROJECT.

CHANGES IN MODULE FROM CSC-30013 TO CSC-30014 ARE DEEMED TO BE A SIGNIFICANT CHANGE REQUIRING AN UPDATED PLAN.

Appendix B - Project Poster


Course: Computer Science
Module: CSC-30014
Supervisor: Dave Collins
Student: Chi Chung Derek LEE
Student no.:12024326
Year: 2016
Keele University
GPS Campus Navigation System
Web-based mobile routing and way-finding application for Keele University Campus
Introduction
This project uses open-source data source and API's to develop a Location Based System for mobile devices that helps users navigate in the Keele University campus. The application aims at presenting a practical solution for students to find the shortest path from one location to another, reducing delays by means of providing information surrounding the campus, for example, personalized class information, event information and campus police announcements. Additionally, users may find their friends' location on the university grounds. To achieve these much-appreciated features, large chunks of contextual information have to be integrated into the navigation process.
Resources & Methodology
Local mobile routine services with POI queries demand primarily accuracy and sufficiency of data of the area. The main requirements of the project are (a) highly efficient system development and easy maintenance, (b) effective and useful data and services and (c) economical. This project illustrates a solution based on the open-source Leaflet mapping library and OpenStreetMap map data source.
Leaflet, developed by Vladimir Agafonkin, was first released in 2011 [Lovelace, Robin. "Testing web map APIs - Google vs OpenLayers vs Leaflet".] It is an extensively used open source JavaScript mapping library for constructing web mapping applications, supporting most desktop and mobile platforms such as HTML5 and CSS3. It also allows developers to (1) display tiled web maps, with optional tiled overlays, hosted on a public server. Feature data can be loaded from GeoJSON files and styled creatively. Interactive layers such as markers with popups can be made.
OpenStreetMap (OSM) is an editable map of the world, created by contributors and free to use under an open license. Users can view, edit and use geographical data in the map collaboratively. Data collected are then made available under the Open Database License. OSM benefits from the accurate and up to date local knowledge of mappers who use imagery, GPS devices, and low-tech field maps.
System Architecture
User
User
Cellular Data / Wifi
HTTP
HTTP
HTTP
Admin User
Internet
OSM Database Server
User
User
Routing Plugin
Leaflet API
Going Forward
Semester 1: Week 1, Week 3, Week 5, Week 7, Week 9, Week 11
Semester 2: Week 1, Week 3, Week 5, Week 7, Week 9, Week 11
Planning the project
Data collection and research
Start project introduction
Coding and documentation
Create poster
Poster session
Report wirting
Finalize project
Report submission
User Interface
Simulated user interface on smart phones
Aims & Objectives
-To provide a map of the campus for users to view digitally.
-To provide routes of navigating within the campus, with possible routing options.
-To devise a dynamic turn-by-turn navigation system using the GPS built-in to the mobile device.
-To incorporate a reactive website which the user can use on either a mobile device or a PC.
-To provide latest campus information based on selected location, such as events and news.
References & Links
Zheng, J., Chen, X., Ciepluch, B., Winstanley, A., Mooney, P. & Jacob, R. 2010, "Mobile routing services for small towns using cloudmade API and OpenStreetMap",
http://www.openstreetmap.org
http://leafletjs.com/

Appendix - Short Ethic Form

School of Computing and Mathematics: Student Project Ethics Committee Application form (U/G and PGT Students)

Please print off a hard copy of this form and submit it to the School Office.
Please remember to sign it, date it, and to get your supervisor's signature on it.

Student name:	LEE Chi Chung Derek
Student number:	12024326
Course:	CSC-30014
Date:	10/11/2015

Part A: (all students)

Does the topic of the project involve any of the following?	**YES**	**NO**
Recall of personal or sensitive memories		✓
Reporting or discussion of personal or sensitive topics		✓
Tasks which could be harmful or distressing		✓
A significant risk of participants later regretting taking part		✓
Procedures which are likely to provoke inter-personal or inter-group conflict?		✓

If you answer "Yes" to any questions on Part A, then please also complete the University Ethics form (on the KLE) and seek guidance from the School Research Governance officer. The School Research Governance officer (mentioned in the university form) is the projects co-ordinator, Gordon Rugg.

Part B: (if you are doing a software design and/or software build or gathering data from human participants)

Requirements gathering and evaluation: use of unusual techniques	YES	NO
Will the techniques you are using for requirements gathering and software evaluation be unusual in a way which could cause ethical problems?		✓
Will any of the participants be from a vulnerable group (e.g. under 18, or with learning difficulties, or under pressure to help you)?		✓

If you answer "Yes" to any questions on Part B, then please also complete the University Ethics form (on the KLE) and seek guidance from the School Research Governance officer. The School Research Governance officer (mentioned in the university form) is the projects co-ordinator, Gordon Rugg.

Student's signature and date:	
Supervisor's signature and date:	