A Navigation System for Wheelchair Users

Individual Final Report of Nip Chi Fung

Supervisor
Dr. T.W. Chim

Student
Nip Chi Fung, Terry (2013571644)

Another Group Member
Chau Shing Yi, Sabrina (2013712810)

Date of Submission
16th April 2017
Abstract

Path finding is not an easy task for wheelchair users. In Hong Kong, many wheelchair users rely on a physical map or Google Maps for navigation. However, those maps may not suitable for this group of users as they have special needs to take care of; Wheelchairs users sometimes suffer from taking the wrong paths because their applications cannot reveal the ‘barriers’ such as staircases in the routes for them.

The objective of the project is going to develop a navigation application WheelGO which addresses the navigation issues and provides optimal paths to wheelchair users in a smartphone-based approach.

This paper reviews the existing applications on the market and evaluates their strengths and weaknesses in the purpose of enhancing the performance and usability of our application. Like other map applications, WheelGO displays turn-by-turn routing instructions that suitable for wheelchair users, indicates accessibility information of each facilities and allows users to report bugs via the feedback form. Besides, the application also has add-on features such as voice guidance function to ensure wheelchair users have a good routing experience.
Acknowledgement

We would like to express our gratitude to the friendly support provided by our final year project supervisor Dr. T. W. Chim, English class instructor Mr. Ken Ho and the Department of Computer Science, Faculty of Engineering, The University of Hong Kong during the preparation of this final year project.
Table of Contents

LIST OF FIGURES .................................................................................................................. 5
LIST OF TABLES ...................................................................................................................... 5
ABBREVIATIONS ..................................................................................................................... 5

SECTION I – INTRODUCTION ................................................................................................ 6
1.1 INTRODUCTION .................................................................................................................. 6
1.2 EXISTING SOLUTIONS ........................................................................................................ 7
1.3 OBJECTIVES ...................................................................................................................... 9
1.3 TECHNICAL OBJECTIVES ............................................................................................... 10
1.4 SCOPE ............................................................................................................................... 10

SECTION II – APPROACH AND METHODOLOGY .............................................................. 12
2.1 SOFTWARE DEVELOPMENT CYCLE ............................................................................... 12
   2.1.1 Design Phase .............................................................................................................. 12
   2.1.2 Implementation Phase ................................................................................................. 12
   2.1.3 Testing Phase ............................................................................................................. 12
2.2 DEVELOPMENT COMPONENTS ...................................................................................... 13
   2.2.1 Equipment Setup ....................................................................................................... 13
   2.2.2 Development Tools ................................................................................................... 13
2.3 TASK LIST, DIVISION OF WORK AND STATUS ........................................................... 14

SECTION III – PRODUCT ARCHITECTURE ....................................................................... 15

SECTION IV – PRODUCT FUNCTION DESCRIPTIONS ...................................................... 17
4.1 MAP FUNCTION ................................................................................................................ 17
4.2 ROUTING FUNCTION ....................................................................................................... 18
4.3 ACCESSIBILITY INFORMATION ....................................................................................... 19
4.4 VOICE GUIDANCE ........................................................................................................... 19
4.5 GPS LOCATION ................................................................................................................ 19
4.6 FEEDBACK MECHANISM ............................................................................................... 20

SECTION V–WORK DONE BY NIP CHI FUNG .................................................................. 21
5.1 OVERVIEW ....................................................................................................................... 21
5.2 MAP DATA CONSTRUCTION ............................................................................................ 21
5.3 MAP VIEW CONSTRUCTION ............................................................................................ 24
5.4 ROUTING ALGORITHM CONSTRUCTION ......................................................................... 27
5.5 USER EXPERIENCE ENHANCEMENT ............................................................................ 34

SECTION VI – CONCLUSION ............................................................................................... 38

SECTION VII – REFERENCES .............................................................................................. 39

SECTION VIII – APPENDIX I ............................................................................................... 40
List of Figures

Figure 1 A screenshot of the route suggested by the Hong Kong Society for Rehabilitation website ..... 7
Figure 2 Two paths from Knowles Building, HKU to Meng Wah Complex, HKU suggested by Google Maps, both paths include staircases .................................................. 8
Figure 3 Prototype demonstrating the features of WheelGO .......................................................... 9
Figure 4.1 Logo of Android Studio .................................................................................................. 13
Figure 4.2 Logo of PostgreSQL ........................................................................................................ 13
Figure 4.3 Logo of QGIS .................................................................................................................... 13
Figure 4.4 Logo of Mapbox .............................................................................................................. 13
Figure 5 System architecture ............................................................................................................. 15
Figure 6 Application UI showing the mapview ................................................................................. 17
Figure 7 Application UI showing the routes ..................................................................................... 18
Figure 8 Application UI showing the accessibility information ...................................................... 19
Figure 9 Application UI showing the feedback form ................................................................... 20
Figure 10 Application UI showing the GPS location .................................................................. 20
Figure 11.1 OSM map data visualized by QGIS ............................................................................ 22
Figure 11.2 WheelGo map data visualized by QGIS .................................................................... 22
Figure 12 showing the accessibility information provided by MTR website .................................... 24
Figure 13 Mapbox Editor displaying the bus stop dataset ............................................................ 25
Figure 14 Mapbox Editor editing the MTR railway line ................................................................ 25
Figure 15 Mapbox Editor highlighting the staircases in red ............................................................ 26
Figure 16 Mapbox Editor showing the available MTR exits ............................................................ 26
Figure 17 Mapbox Editor showing the entire mapview used in the application ............................. 27
Figure 18 Street illustration for NN v1.0 ......................................................................................... 28
Figure 19 Street illustration for NN v1.1 .......................................................................................... 29
Figure 20 Street illustration for NN v2.0 .......................................................................................... 30
Figure 21 an example of bus route .................................................................................................. 32
Figure 22 an example of MTR route ................................................................................................ 32
Figure 23.1 an example showing 2 nodes on a edge ...................................................................... 34
Figure 23.2 an example showing 2 nodes on two adjacent edges ................................................. 34
Figure 24 a demonstration of zoom in/out function ...................................................................... 35
Figure 25 a demonstration of the progress bar ............................................................................. 36

List of Tables

Table 1 Comparison of different map products on the market ......................................................... 8
Table 2 Smartphone operating system market share report in 2016Q2 ........................................ 10

Abbreviations

API Application Programming Interface
HKSRC Access guide by the Hong Kong Society for Rehabilitation
OSM OpenStreetMap
Section I – Introduction

1.1 Introduction

Route planning is a crucial task for navigation. For wheelchair users, the case is even more challenging as there are several unique constraints imposed to the wheelchair users. For instance, in the course of planning, the route planner must not consider any route that contains staircases as most of the wheelchairs cannot climb the staircases. The routes also need to take into account how far the users are able to travel with their wheelchairs and how long is the travel time when exceeded (threshold), wheelchair users may want to travel with public transport instead because a lengthy travel may exhaust some of the users.

Apart from route planning, when traveling to an unfamiliar place, many wheelchair users are also interested in knowing the accessibility information of the facilities. For example, for an underground plaza, people may want to know are there any accessible toilets and are the entrances have only steps or elevators for the incomer to get in.

Through extensive research, it is appeared that there does not exist any effective tool on the market for wheelchair navigation. Many wheelchair users are currently using the existing map products on the market such as the paper map and Google Maps for navigation. However, map products as such do not provide any accessibility information and cannot detect and eliminate staircases in their routes. Moreover, most of the tools require users to regularly stop at every junction to check the maps for direction. Resulting in safety issue and might even make the users fall into a danger situation. Besides, there are some organizations do offer routing and accessibility information look-up services to wheelchair users. For instance, the access guide of the Hong Kong Society for Rehabilitation provides routes planning service specifically to wheelchair users. Wheelmap is a free map that indicates wheelchair accessible places (detailed discussion of the existing products can be found in section 1.2). However, these applications have their limitations and, most of the time, cannot satisfy users’ needs. Many wheelchair users still hesitate to visit an unfamiliar place.

In a nutshell, there is a gap in the market for products that can integrate the accessibility information look-up function, and at the same time allow routing planning targeting wheelchair users.

In the following part, the paper will first discuss the other existing solutions and highlight their strengths and weaknesses. Next, the objectives and scopes of the application will be illustrated in detail. After that, the methodology of the project and a brief progress report with the
difficulties encountered will be listed out. Based on the work done and the problems encountered at the current stage, the future work plan of the project will be delivered.

1.2 Existing Solutions

There are several map providers in Hong Kong. The following briefly discusses three applications mainly used by the wheelchair users, namely, Access guide by the Hong Kong Society for Rehabilitation, Wheelmap and Google Maps:

1. Access guide by the Hong Kong Society for Rehabilitation (HKSR) [4]
   The organization website provides an access guide for wheelchair users to a list of public places (see Figure 1 for the screenshot). Some of the popular attractions have YouTube videos to demonstrate how to get across the roads and buildings. However, the destinations available are limited and the starting points are fixed at public transport stations. Users are not able to choose freely any other starting points and endpoints. This induces to a problem that once the user’s route deviate from the suggested path, he can hardly find his way based on his current location. Moreover, the website does not provide mobile version, it is inconvenience for users to look up the route when he is traveling. This website is not necessarily helpful for the local citizens but nonetheless a great guide in the point of view of a tourist. While different from this access guide, WheelGo allows users to dynamically select any two points on the map as the source/destination pair and support Android application, thus foster flexibility.

2. Wheelmap [1]
   Wheelmap is a free map that indicates wheelchair accessible places all over the world. The map model is based on OpenStreetMap(OSM), a crowdsourcing map project that allows users to contribute by editing the data. It imports the street view from Google street view and provides valuable information such as the accessibility of each facilities and their contact numbers. Since Wheelmap does not provide routing service, wheelchair users cannot use the application to obtain suitable routes and need to find the alternatives for routing. In WheelGo, not only the accessibility information, but also the feasible routes to destinations will be displayed. Since all the functions are integrated, wheelchair users do not need to check information from different websites.

Google Maps is the most popular navigation application in Hong Kong. The functionalities include turn-by-turn navigation, street view seeing, distances and traveling time measurement between two points etc. The application supports turn-by-turn routing for people with different modes. However, it does not provide routing solution designed for the wheelchair users. The most similar alternative for wheelchair users is the ‘walking’ mode. While wheelchair cannot take full advantage of it because in some situations, all the routes recommended includes waypoints that are not suitable for them. Figure 2 shows the two paths suggested by Google Maps. No route is suitable for wheelchair users as both include staircases. Unlike Google Maps, WheelGO is only designed exclusively for the wheelchair users. Each route WheelGO recommends is suitable for them.

To conclude, Table 1 shows the comparison of strengths and weaknesses among different products on the market, WheelGO summarizes the strengths of each existing product and ready to provide the best service to the users.

<table>
<thead>
<tr>
<th>Functionality</th>
<th>WheelGO</th>
<th>HKSR</th>
<th>Google Maps</th>
<th>Wheelmap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile Version</td>
<td>✔</td>
<td>✖</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Route for Wheelchair</td>
<td>✔</td>
<td>✔</td>
<td>✖</td>
<td>✖</td>
</tr>
<tr>
<td>Turn-by-turn routing instruction</td>
<td>✔</td>
<td>✖</td>
<td>✔</td>
<td>✖</td>
</tr>
<tr>
<td>GPS Application</td>
<td>✔</td>
<td>✖</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Voice Guidance</td>
<td>✔</td>
<td>✖</td>
<td>✖</td>
<td>✖</td>
</tr>
<tr>
<td>Accessibility Information</td>
<td>✔</td>
<td>✔</td>
<td>✖</td>
<td>✔</td>
</tr>
</tbody>
</table>

Table 1 Comparison of different map products on the market
1.3 Objectives

The project *WheelGO* is a smartphone-based approach that aims to address the navigation issues encountered by the wheelchair users and to build a free, open and tailor-made navigation solution for wheelchair users.

The preliminary goal is to develop an Android application which can process user requests, acquire accessibility information and suggest barrier-free paths to users (See illustration in Figure 3). With a mobile application, users can enjoy the mobility provided and thus enjoy their journeys more. The application will also provide routing information including time and transportation details. To assist wheelchair users to find path when they are in an unfamiliar place, special facilities such as accessible toilets and ramps, or streets that are inaccessible will be highlighted on the map. The application is primarily designed for local use only. Accordingly, the application can only process the map requests for the Hong Kong area.

Increase travel satisfaction of the wheelchair users is also one of the objectives of this project. As the application is intentionally build for the wheelchairs, functions and information which are uniquely helpful to wheelchair users are deployed in our application. Voice guidance, and feedback mechanism are therefore implemented. Voice guidance is similar to the GPS driving audio routing, the system will voice out the route according to the users’ real-time GPS location. For feedback system, users can report staircases or places that are not readily accessible for wheelchair users, after receiving the responses, the system then will undergo validation process and update the map so that the accuracy of the routing function can be improved. By adding more functions and features into the application, it is anticipated that the safety and travel satisfaction of the wheelchair users will greatly increase with the aid of one single application *WheelGO*. 

![Figure 3 Prototype demonstrating the features of WheelGO](image)
1.3 Technical Objectives

The project also defines some technical objectives with respect to accuracy. The final deliverable should be capable of achieving:

- **80% Staircase coverage in Hong Kong**
  
  A dataset (database table) is built for storing the locations of staircases in Hong Kong. The limitation of revealing all staircases (excluding indoor corridors) accounts for the application covering only 80% of the staircases in Hong Kong. As agreed by supervisor, brute-force way of achieving map data (i.e. site visit) is inefficient and therefore this application collect staircase data mainly from existing resources on the Internet. Site visit for few districts will be conducted to estimate and validate the staircase coverage.

- **100% Routing accuracy without staircases**
  
  Routes computed by the application should be able to avoid staircases in the dataset.

1.4 Scope

The navigation system implements on the Android platform only.

There are two reasons for not choosing to build a mobile version and iOS version. First, get start with the most familiar one. Trying to build an application for iOS while simultaneously building for Android device increase the complexity and the time spent on the project. Since the team is familiar with the android development platform, thus building application with Android first is expected to consume less development time in getting familiar with the development platform. Second, Android currently has the largest global platform share, as seen in Table 2, the market share of Android is account for 80-90% of the market share and keep increasing, building an Android application can benefit the most population.

<table>
<thead>
<tr>
<th>Period</th>
<th>Android</th>
<th>iOS</th>
<th>Windows Phone</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015Q3</td>
<td>84.3%</td>
<td>13.4%</td>
<td>1.8%</td>
<td>0.5%</td>
</tr>
<tr>
<td>2015Q4</td>
<td>79.6%</td>
<td>18.6%</td>
<td>1.2%</td>
<td>0.5%</td>
</tr>
<tr>
<td>2016Q1</td>
<td>83.4%</td>
<td>15.4%</td>
<td>0.8%</td>
<td>0.4%</td>
</tr>
<tr>
<td>2016Q2</td>
<td>87.6%</td>
<td>11.7%</td>
<td>0.4%</td>
<td>0.3%</td>
</tr>
</tbody>
</table>

Source: IDC, Aug 2016

*Table 2 Smartphone operating system market share report in 2016Q2*
The routing service will be built based on the map of Hong Kong area only. The purpose of this application is to serve the location citizens as well as the tourists come to Hong Kong.

**The navigation system does not cover indoor map routing.** Indoor map routing requires three-dimension indoor data which is not feasible to collect the data. Most of the facilities are in fact private place that it may have a risk that public is forbidden to view the internal structure of the buildings on the map and result in privacy issue. Besides, the GPS technology nowadays is not mature enough to get the accurate location of the users inside a building.

**The application will have 2 layers: cross districts layer and detailed map layer**
- Detailed map layer: Indicates the accessibility of the places (e.g. restaurants/shopping malls) and shows the point-to-point shortest route
- Cross districts layer: Shows the point-to-point shortest route across different districts and suggests transport means

1.5 List of Features
- Recommends routes for wheelchair users (limited to Hong Kong area)
- Displays turn-by-turn routing instructions
- Indicates the accessibility information of each point of interest
- Provides voice guidance and notification function
- Allows users to report bugs via feedback form
Section II – Approach and Methodology

2.1. Software Development Cycle

The project will follow a software development cycle framework. The cycle is divided into three phases, namely, design phase, implementation phase, and testing phase.

2.1.1 Design Phase

The design phase started in October. At this stage, the project team was focus heavily on studying the feasibility of implementation different functions and devising the application architecture.

The system design phase includes the following works:

• Brainstorm any value-added functions that wheelchair users need
• Decide the software used
• Design the routing algorithm
• Design the database
• Design the user interfaces
• Design the testing area

2.1.2 Implementation Phase

The implementation phase started at early December, where the features and functions designed is implementing into the WheelGO.

2.1.3 Testing Phase

The testing phase was carried out at late March. The main focus of the whole testing process was to reveal the performance of the application under normal and abnormal inputs. Details of testing can be referred to Appendix I.
2.2 Development Components

2.2.1 Equipment Setup

- Smartphone OS Platform: Android (with GPS receiver installed)
- Smartphone Minimum SDK: Android 4.4 (KitKat)
- Development Platform: Android Studio
- Map data service: OpenStreetMap (OSM)

2.2.2 Development Tools

The tools used are as follow:

- **Android Studio**
  
  Android studio is the standard development IDE for developing android applications.

- **PostgreSQL (with PostGIS)**

  PostgreSQL is a NoSQL Object-oriented Database. With its extender PostGIS, PostgreSQL is able to support storing geographic objects and allow queries on the objects. In PostgreSQL Map Data are stored as Points/ Polylines/ Polygons.

- **QGIS**

  QGIS is a geographic information system that allow create, edit, visualise, analyse geospatial information.

- **Mapbox**

  Mapbox is a mapping platform that allow users to visualizing map data and do analysis in real time. It provides API (Application Programming Interface) for developers to design their own map in Mapbox Studio.
### 2.3 Task List, Division of Work and Status

<table>
<thead>
<tr>
<th>Time</th>
<th>Task</th>
<th>Division of Work</th>
</tr>
</thead>
<tbody>
<tr>
<td>September</td>
<td>Researched on the existing products on the market</td>
<td>Sabrina Chau, Terry Nip</td>
</tr>
<tr>
<td></td>
<td>Investigated the feasibility of implementation different functions</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Defined the project scope</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Designed the project webpage</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Prepared the project plan</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Acquire the necessary development tools</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Brainstormed any additional value-added features</td>
<td></td>
</tr>
<tr>
<td>October</td>
<td>Build app with basic routing function</td>
<td>Terry Nip</td>
</tr>
<tr>
<td></td>
<td>Design and input data into the map database</td>
<td>Sabrina Chau, Terry Nip</td>
</tr>
<tr>
<td></td>
<td>Draft the tentative UI</td>
<td></td>
</tr>
<tr>
<td>November</td>
<td>Import map data into android studio to build the MapView [Milestone 1]</td>
<td>Sabrina Chau, Terry Nip</td>
</tr>
<tr>
<td>December</td>
<td>Implement the turn-by-turn routing function</td>
<td>Sabrina Chau, Terry Nip</td>
</tr>
<tr>
<td>January</td>
<td>Continue implementing the turn-by-turn routing function [Milestone 2]</td>
<td>Sabrina Chau, Terry Nip</td>
</tr>
<tr>
<td></td>
<td>Write the interim report</td>
<td>Sabrina Chau, Terry Nip</td>
</tr>
<tr>
<td>February</td>
<td>Implement acoustic navigation function [Milestone 3]</td>
<td>Sabrina Chau</td>
</tr>
<tr>
<td></td>
<td>Construct and design the Mapview</td>
<td>Terry Nip</td>
</tr>
<tr>
<td>March</td>
<td>Implement the feedback mechanism [Milestone 4]</td>
<td>Sabrina Chau</td>
</tr>
<tr>
<td></td>
<td>Improve the UI design of the app</td>
<td>Sabrina Chau</td>
</tr>
<tr>
<td></td>
<td>Improve the routing algorithm in term of user experience and include other modes of routing</td>
<td>Terry Nip</td>
</tr>
<tr>
<td></td>
<td>Test and debug</td>
<td>Sabrina Chau, Terry Nip</td>
</tr>
<tr>
<td>April</td>
<td>Finalize the implementation [Milestone 5]</td>
<td>Sabrina Chau, Terry Nip</td>
</tr>
<tr>
<td></td>
<td>Write final report</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Prepare the exhibition poster</td>
<td></td>
</tr>
</tbody>
</table>
Section III – Product Architecture

Main Component

◆ WheelGo Application
The only application available to the users, the application is developed under the Android platform. It is responsible for most interactions with the users. The application provides different functions for the users. Please refer to later session for details of the product functions.

◆ Web Server
The web server acts the communication channel between the WheelGo application and the database server. The server is written in Node.js. The main responsibility of the server is to handle requests from the WheelGo application, compute the related results and return to the application. For examples, when the user wants to calculate a shortest routes of 2 points. The WheelGo application will send a HTTP request with the 2 points to the web server. The server after obtaining result from the database will then send the GSON response back to the application.
**Database Server**

In our project, a spatial database Postgres is used. In our database, we have stored different data for calculations of routes and other information, e.g., the bus stop location, the accessibility information. Both the database and the web server are operated in the virtual machine provided by the department of Computer Science. Therefore, the service is always available.

**External Mapview**

The only component maintained by the Mapbox. Our application has a mapview provided by the Mapbox API.
Section IV – Product Function Descriptions

In this section, the major functions in our application will be discussed in detail.

4.1 Map function

The map is obtained by calling the access token from Mapbox API. The map is used to show the route with the starting location as the present location of the user and the destination inputted by the user. Details of the construction of the map can refer to later session.

![Mapview](image-url)
4.2 Routing function

The most fundamental function in the application, 3 modes of routing available for the users: Wheeling Mode, Bus Mode and MTR mode. User input the source/destination by tapping the map or typing in the search box, a route will then be calculated.

![Application UI showing the routes](image)

*Figure 7 Application UI showing the routes*
4.3 Accessibility information

The application also displays accessibility information of different locations. Users can select the categories they want to show; the application will then spot out all the accessibility information of the selected categories. The red spots stand for not accessible places, while the green spots are the accessible places.

![Accessibility Information](image)

*Figure 8 Application UI showing the accessibility information*

4.4 Voice guidance

When user taps the immediate nodes in the routes, the application will voice the instruction of the node. The details of how the instructions are calculated can refer to Chau Shing Yi Sabrina’s report.

4.5 GPS location

Like other map application, our application allows users to obtain their GPS locations instantly by clicking a single button. This facilitates the users to select their own locations as the source of the routing algorithm.
4.6 Feedback mechanism

There should always be ways for our application to improve. Our application has implemented a feedback form to allow users to report bugs or comments related to our application. All the feedback information will be stored into our server database.
Section V–Work done by Nip Chi Fung

5.1 Overview
I worked closely with my partner Chau Shing Yi Sabrina throughout the project. Each of us was assigned to complete different tasks. I personally have more interested in algorithm design and analysis. Therefore, in this project, I was mainly responsible for implementing the routing algorithm for wheelchair users, the steps included map data construction, map view construction, routing algorithm design and finally visualizing the routes in the Android application.

Besides, I also took part in a portion of User experience enhancement implementation: construction the zoom in/out function and the progress bar.

5.2 Map Data Construction
In this project, 3 different sources of data were used: OpenStreetMap, Data.gov.hk and MTR. The following session will report how the datasets were collected and the difficulties encountered when managing the data.

OpenStreetMap
OpenStreetMap is a collaborative project to create a free editable map of the world, it provides free worldwide map data. Since our project is only focusing Hong Kong, we downloaded and analyzed the Hong Kong region data from the OSM.

Since the OSM data is tailor-made for map application, its data provides lots of attributes which suit our project’s need. For instance, every node from the OSM is labelled with latitude, longitude and the category, so that we can easily plot it on the map and identify the node effectively.

To facilitate the wheelchair users, we removed all the edges which were indicated as staircases before importing data into our own databases. By doing so, we guarantee that no staircase is presented in our database and therefore when we conduct shortest route algorithms, no routes with staircases can be presented. Below figures demonstrate how data in our own database is different with the OSM original data. Left hand side represents data from OSM, while right hand side is our data in the database. The staircases are missing in our database.
Apart from removing all the staircases in the database, we took one step further to facilitate the wheelchair users. Since all edges from OSM have their categories, e.g., some edges are indicated as pedestrian, residential or motorway, we considered the preference as wheelchair users when need to choose which ways to go. For instance, wheelchair users perhaps prefer pedestrian or living street then walking across a motorway. Therefore, when setting the cost for edges, we were not only taking into account of the distance of the edge, but instead we multiplied the distance with a factor ranging from 1 to 5. The factor represents the category the edge presented. The system penalizes edges indicating highways or motorways by setting high value of the factor, and treasures the edges indicating pedestrian or living street by setting the factor value to 1. As a result, the application provides routes which are more accessible to the wheelchair users.

```xml
<class name="trunk_link" id="105" priority="3.5"/>
<class name="primary" id="106" priority="3.5"/>
<class name="primary_link" id="107" priority="3.5"/>
<class name="secondary" id="108" priority="3.0"/>
<class name="secondary_link" id="124" priority="3.0"/>
<class name="pedestrian" id="114" priority="1.0"/>
<class name="services" id="115" priority="1.0"/>
<class name="bus_guideway" id="116" priority="5.0"/>
<class name="path" id="117" priority="1.0"/>
<class name="cycleway" id="118" priority="1.0"/>
<class name="footway" id="119" priority="1.0"/>
```

Above is the extraction of categories with different priority. By set this configuration, we can lower the chance for wheelchair users to walk through some dangerous path, e.g., highways or motorways.
Data.Gov.hk
Second data source used in our project. The map data provided by the OSM do not contain any information related to public transport. However, public transportation is extremely important to wheelchair users because they usually prefer taking transportation means to destination than walk through it. Therefore, we obtained the public transportation data in Data.Gov.hk.

Our project obtained following data related to bus from Data.Gov.hk:

- **Routes of bus**
- **Stop sequence of bus**
- **Coordination of bus stop location**

The most time-consuming step in importing the bus data is the conversion of bus stop coordination data. The coordinate system used by bus stop is HK 1980 Grid System (x, y coordinate representation), while the OSM data uses geographic coordinate system (represents the geographic information as longitude and latitude). Conversion is needed for each data before we integrated all the bus stop data into the database.

Besides, the bus data from Data.gov.hk originally support Chinese name of their bus stop names. However, we encountered a problem when importing the Chinese name, the entries were corrupted. Therefore, we decided to remove the Chinese name entry from our database schema.

MTR
The final data source used in our project. The construction of MTR data is relatively time consuming when comparing with importing map or bus data. Neither Data.gov.hk nor the MTR provides the exact location (Latitude and Longitude) of the exits. Therefore, we had to create our own MTR exit data. The official MTR website contains information of accessibility facilities in different station. This information is important such as our application can advise the users to use which exit. The following figure demonstrated the data we obtained from the MTR website.
Having known the accessibility facilities in each station is not enough to distribute suitable routes to users, we have to obtain the geographical location of each exit. However, not geographical locations of exits are readily available in the internet. Therefore, we had to manually locate each point onto the OSM map and obtain the latitude and longitude. The database table of MTR exits has around 200 entries, where each entry represents a represent an exit. The construction of MTR exit data is extremely time consuming.

5.3 MapView Construction

One of the very key components in our application is the map view, where the user can see and interact with the application. In this project, Google Map View was not used, we built our own map view using the Mapbox Editor instead.

The first step of building the map view is to import the map datasets to the Mapbox editor. For examples, we have data representing the MTR exits or data representing the staircases. All the csv files storing these data are directly imported to the Mapbox editor.

After the map datasets are imported, the editor then transforms the datasets into different map tiles, where they are ready to be edited for different styles and imported into the map view. Figure 13 demonstrates the bus stop dataset is being loaded into the Mapbox Editor, the blue points are the locations of the bus stops.
Figure 14 shows the tiles for the MTR railways line, and demonstrates how the tiles can be edited. For examples, we can set the visibility zoom level (how close the zoom level is to allow users to see the lines in the map), the colour or icons.
Since our application is designed for wheelchair users, we pay special attention when designing the style. The following features are the designs for wheelchair users.

1) **Highlighting the staircases**

   Figure 15 shows that all the staircases in our map have been highlighted in red. Climbing through staircases is infeasible for the wheelchair users. Highlighting the staircases can allow users to recognize them easily and get rid of them.

2) **Only Visualizing the MTR exits accessible to the wheelchair users**

   MTR exits without accessibility facilities are useless for users. Therefore, removing irrelevant exits from the map will less mislead the users.

   The exit A is disappeared in Figure 16 as there is no ramp or other accessibility facilities.

   ![Figure 15 Mapbox editor highlighting the staircases in red](image1)

   ![Figure 16 Mapbox editor showing the available MTR exits](image2)
After loading the datasets and designing the style, a complete map is ready to be imported into our Android application. The below figure shows the map after editing.

5.4 Routing algorithm construction

In this session, details of how the routes are being calculated will be discussed. First, there are three modes of routing available for the users: Wheeling Mode, Bus Mode and MTR mode.

1) Wheeling Mode

A very typical routing problem where the users input 2 points: source and destination, and request a shortest route. In our project, Dijkstra Algorithm is being used to solve the shortest path problem. The full working principle of Dijkstra Algorithm will not be discussed in this session. And the purpose of the projection is not to analyse or enhance the Dijkstra Algorithm, but is to make use of the Dijkstra Algorithm to provide solution to the wheelchair users.

Nearest node selection

Nearest node selection is always a typical problem when allowing users to pick any point in map as source/destination. Since, the data points in our map database are limited, not every user’s input have a matched record in our database. Therefore, we must select the nearest point in our database to substitute the user input.
The ‘optimal’ point to select is worth discussed and we have modified and enhanced our NN algorithm several times throughout the development process.

**V.1.0. Selecting the node with shortest distance to user input**
This is the most obvious and straightforward solution to most people, and it works fine in most cases. However, it fails in some street patterns.

Consider the above street diagram. The red points are the nodes stored in the database where the green cross is the user input. For user perspective, it would be more responsible to select the one of the points in the upper street as the source because apparently, the user is located near the upper street. However, the v.1.0 NN algorithm will select the lower red spot as the input as the distance is the shortest. Thus, the whole route calculated will be less meaningful because of the false point selection.

**V.1.1. Selecting closest point from the nearest edge**
This is the enhanced NN algorithm where we first select the nearest edge from the map, we then choose the closer point (either the head or tail) from the edge as the user input. Consider the above example again, the v.1.1. now first select the upper edge, then select the upper left point as the input (because the distance is closer). In this case, the route calculated is more relevant as the user is more likely to pass through the selected point.

![Figure 18 Street illustration for NN v1.0](image)
Be remaindered that the purpose of NN selection is to select a substituted point to aid calculation but still the source from users should not be changed. That is, to complete a route, the application should 1) provide a route for user to walk to the substituted point and 2) the route to the destination which is calculated by Dijkstra Algorithm.

![Figure 19 Street illustration for NN v1.1](image)

However, the v.1.1 NN still cause problems in some routing cases. For instance, the nearest edge itself is included in the shortest route. In this case, the application suggests the user to walk a U-turn path, which is not desirable.

**V.2.0. Selecting the point in the closest edge with concern of the shortest routes**

This is the current NN selection algorithm used in our application. For each user’s source and destination, we find the nearest edge respectively. After obtaining the 4 points (2 for source and 2 for destination), we run the Dijkstra Algorithm for each pair of source destination and take the pair with the shortest cost. The 2 selected points are the substituted source and destination.

This algorithm takes the assumption that the cost between the user input and substituted points is negligible to the overall cost of the path.

Under v.2.0. approach, we can guarantee that the nearest edge will not be included in the Dijkstra result and thus, the application will not suggest the user to walk redundant street.
2) **Bus Mode**

Bus mode is a more complicated routing problem than routing through wheeling. In high level abstraction, the users first look around the nearby bus stop, look for the suitable bus route, take the bus and get off at the nearest bus stop. If the distance is still too far away from the destination after taking the first bus trip, we will repeat the steps and look for new bus route.

Our routing algorithm operates the same way. And all the information required for routing is stored in the database. The details of dataset related to bus routes and the data manipulation can refer to the Map Data Construction Session. Here we focus on how we make use of the data to compute the suitable routes in detail.

1) **Finding nearby bus stops**

This is done by conducting a range search. For every user’s input (source), the application first conducts a range search to get all the nearby bus stop. In this project, we have set the range bound to 500m. The purpose of setting range bound is that we try not to let the user to walk too far away to get to the suitable bus stop.
After obtaining all the bus stops in the selected range, the application further lists the all the bus routes included in those bus stops and then find a bus stop such that it has the shortest distance to the destination.

```sql
with temp1 as (with temp as (select stop_id, lat, lon, dist from stop_bus,
   ST_distance(geometry(ST_GeomFromText('POINT("'+origin.getLongitude()+'" 
   "'+origin.getLatitude()+'"',4326))::geography,geom)as dist where dist <500
   order by dist)select distinct route_id, lat, lon, temp.stop_id, stop_seq,
   route_seq from route_stop, temp where temp.stop_id =
   route_stop.stop_id )select route_bus.route_name, temp1.route_seq,
   temp1.route_id,temp1.stop_id as sstop_id,temp1.lat as slat, temp1.lon as 
   slon, stop_bus.lat as elat, stop_bus.lon as elon, stop_bus.stop_id as 
   estop_id, dist from route_bus, temp1, stop_bus, route_stop,
   ST_distance(geometry(ST_GeomFromText('POINT("'+destination.getLongitude()+'" 
   "'+destination.getLatitude()+'"',4326))::geography,geom)as dist where
   route_stop.route_id = temp1.route_id and route_stop.stop_id =
   stop_bus.stop_id and temp1.route_seq = route_stop.route_seq and
   temp1.stop_seq <= route_stop.stop_seq and route_bus.route_id =
   route_stop.route_id order by dist limit 1;
```

2) Find all the intermediate bus stops travelled

```sql
select lat, lon, stop_name from stop_bus, route_stop where stop_bus.stop_id
   = route_stop.stop_id and route_seq = "+route_seq+" and route_id =
   "+route_id+" and stop_seq >= "+sstop_seq+" and stop_seq <= "+estop_seq+" 
   order by route_seq, stop_seq;
```

3) Find the shortest route from source to the starting bus stop (like Wheeling Mode)

4) If the distance from shortest route from ending bus stop is < 500m
   A) Find the shortest route from ending bus stop (like Wheeling Mode)
   B) Repeat step 1, 2 and 3
Figures below demonstrates how the Bus routes will be visualized in the application.

![Bus route visualization](image1.png)

**Figure 21 an example of bus route**

3) **MTR mode**

The routing principle of MTR is similar to the routing through bus. The application searches both the nearest MTR station exit for the source and destination respectively. However, unlike the bus network, all stations in MTR are connected, i.e., any station can get to another station by travelling various railway lines. This makes the computation of suitable MTR routes much easier. In other words, the computation of the nearest exit to source should be independent with the computation of nearest exit to the destination. Therefore, the two calculations can be done concurrently. In practice, our application, when receive user request, sends two HTTP requests to the server concurrently for the computation of the exits for source and destination. Since, the application need not to wait for the first response to send the second request, the overall response time for the application is shortened.

![MTR route visualization](image2.png)

**Figure 22 an example of MTR route**
The SQL query used to find the nearest MTR exit for the source point.

```
"select latitude, longitude, station_name, exit_name, facility, facility2
from mtr_exit order by
ST_Distance(ST_GeomFromText('POINT("+origin.getLatitude()+"+
"+origin.getLongitude()+"'),4326), geom ) asc limit 1;"
```

The SQL query used to find the nearest MTR exit for the destination point.

```
"select latitude, longitude, station_name, exit_name, facility, facility2
from mtr_exit order by
ST_Distance(ST_GeomFromText('POINT("+destination.getLatitude()+"+
"+destination.getLongitude()+"'),4326), geom ) asc limit 1;"
```

We can observe the 2 SQL are independent of each other, therefore the 2 SQL can be sent to the server concurrently.

**Extreme cases handling**

A number of bugs presented when the 3 modes of routing were first implementation. Especially when the system receives abnormal input from users. This subsection discusses how our system handles different extremely cases.

1) **Extremely short distance between the source and destination**

If the distance between the source and destination given by the users are extremely short, e.g., the 2 points are located in the same edge. Recall the working principle of the nearest node v2.0, the application will select the same node for both source and destination (since the cost of 2 identical points is always zero). The route therefore return from the server is null. This becomes problematic as we can solve such a naïve routing problem.

In order to solve the above problem, the application is modified to handle this special case. Before sending the route request to the server, our application first check whether the 2 nodes are in the same edge. This can be done by calling the NN algorithm v2.0.. Whenever the edges are identical, our application recognize that the 2 nodes close to each other. Instead of sending routes request, the application will skip the step (since the result must be null) and directly draw route to connect the 2 nodes.
2) **Extremely short distance between the source and destination (scenario 2)**

Another extremely example of extremely close source and destination input. But this time, the 2 nodes do not have the same nearest edge. Yet, their closest edges are connected. Recall the NN algorithm v.2.0., the routing, the server again will return an empty route (since the cost is always 0).

In this case, the application refuse to directly connect the 2 points, since this makes the route displayed less accurate. Consider, the case of a L-shaped road, and the source and the destination are the 2 ends. If the application directly connects 2 points, the route displayed is not accurate. Therefore, our application will further send another request to server requesting the middle point. And after obtaining the point, our application connects the 3 points and draws the route.

![Figure 23.1 an example showing 2 points on an edge](image)

![Figure 23.2 an example showing 2 points on two adjacent edges](image)

3) **Source/destination locates in abnormal areas**

The abnormal areas here are referring to inaccessible places like the center of the sea. The application will select to nearest accessible point as replacement and calculate the routes. Our application will not reject any abnormal input.

5.5 **User experience enhancement**

After the routing algorithm had been constructed, I shifted my work/focus into enhancing user experience. This session will mention my work done to enhance the user experience.
**Zoom in/out function**

A function which is relative easy to implement, yet greatly facilitate the users when interacting with the map. The zoom function allows users to enlarge or shrink the map view by simply clicking buttons. We believed that dragging the screen with 2 fingers to zoom in/out is sometimes not preferred as it blocks the information presented to users (the route). Therefore, the zoom in/out buttons are placed on the right side of the screen so that, the users can clearly observe the routes even when they press the zoom buttons.

The below figures show different zoom levels after pressing the Zoom in/out buttons. The left one is the original zoom level, while the zoom level in the middle is added by 1 (after pressing the zoom in button) and the right figure shows the zoom level after minus 1 (after pressing the zoom out button).

![Figure 24 a demonstration of zoom in/out function](image)

**Response time handling**

This is a typical problem when the application needs to send request to the server, and wait for the response. During the response time, how should the application behave? The problem becomes critical especially when the response time is relatively long.

After the application sent the HTTP request to the web server, it wait for the response. When users send the request again with different input without waiting the response, how should the application behave?
In the early development stage, we did not intentionally set constraint to control this event. As a result, the application will first draw the first route and then immediately replace the route with the new user input.

However, we believe that provides an opportunity for users to exhaust the server by massively sending request without waiting the server reply. Besides, we believe it is unnecessary to allow users to interact with the application until the route is drawn. Therefore, the application is designed to temporarily disable the User interface during the waiting time.

The following code disable the User interactive:

```java
getActivity().getWindow().setFlags(WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE)
```

The following code enable the User interactive:

```java
getActivity().getWindow().clearFlags(WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE);
```

However, the users might be misled that the application is hanged if the application user interface is disable. They can distinguish whether the application is still waiting the server response or it is already hanged. Therefore, a progress bar is implemented to notify the users.

![Figure 25](image)
In Android, there is a View class called Progress bar. The progress bar is first embedded in the map layout. Set the initial state to GONE.

```
loading.setVisibility(View.GONE);
```

Upon receiving user request and waiting for server response, set the progress bar state to VISIBLE. At the same time, disable the UI.

```
loading.setVisibility(View.VISIBLE);
```

After the response is received and the route is drawn, re-set the progress bar status to GONE. At the same time, enable the UI.
Section VI – Conclusion

This report listed out all the information of our final year project: WheelGo. The project finally comes to the end and I personally gained a lot from doing the final year project.

First of all, I gained solid experience in Android development. During the project, I actively took part in the Android back-end development, for examples, using the Retrofit API to allow communication with the server., creating class or object handling storing the data objects and also writing the backend routing algorithms for wheelchair users. Besides, I also got involved in little user experience enhancement work. I got opportunities in front-end development such as adding the zoom in/out buttons and the progress bar.

Other than gaining experience in Android development, I learnt several new languages and tools during the project. For examples, Node.js - a JavaScript-like server side programming language, PostgreSQL - a spatial database, OSM - an open source map data provider and Mapbox API – a powerful map API originated from OSM. Learning a tool is not an easy job, and I am very proud that I could successfully understand and use all these tools in my final year project and help me to accomplish my goal. I am now feeling confident in learning any other new tools in my future career.

Last but not least, I appreciate my final year project but it still has a lot to improve before it is readily serve the best to the wheelchair users. I sincerely hope that I can continue to develop my application and expand its scope and service.
Section VII – References


**Section VIII – Appendix I**

The table below shows the test areas as well as the passing criteria the testing phase will test against:

<table>
<thead>
<tr>
<th>Test area</th>
<th>Description</th>
<th>Passing criteria</th>
</tr>
</thead>
</table>
| Routing Accuracy           | Test whether the route knows to take public transportation when users input a cross-district source/destination pair. | 1. no staircases shown  
2. the route takes public transportation instead of walking  
3. Pass 30 test cases of random source/destination pair |
| Routing Accuracy           | Test whether the route knows to take public transportation when users input a source/destination pair with travel time > 20 minutes. | 1. no staircases shown  
2. the route takes public transportation if any  
3. Pass 30 test cases of random source/destination pair |
| Feedback                   | Test whether the users can successfully send feedback to the server.          | 1. the users can successfully send feedback to the server.  
2. Pass 5 test cases from 5 different Android phones |
| Turn-by-turn routing       | Test whether the turn-by-turn routing instructions can be correctly displayed. | 1. The instructions are properly shown on the screen  
2. The turn-by-turn paths are feasible for wheelchair users to walk through  
3. Pass 30 test cases of random source/destination pair |
| Voice guidance             | Test whether the voice guidance system pronoun all the words appears on the turn-by-turn instructions. | 1. All the words on the routing instructions (including the address name) can be pronounced  
2. Pass 30 test cases of random source/destination pair |
| Voice guidance             | Test whether the voice instructions can switch automatically after it detected the position changes of users. | 1. The delay of the voice instructions is not more than ±10 seconds  
2. Pass 30 test cases of random source/destination pair |
| Accessibility information  | Test whether the accessibility information is located at the correct corresponding locations. | 1. The accessibility information of each place is properly located  
2. Pass 5 test cases of random source/destination pair |