IslanderGo: A GPS-based Navigator for TAMUCC on Android platform

GRADUATE PROJECT REPORT

Submitted to the Faculty of
the Department of Computing Sciences
Texas A&M University-Corpus Christi
Corpus Christi, Texas

In Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Science

By

Nikitha Pandyala
Spring 2018

Committee Members

Dr. Longzhuang Li
Committee Chairperson

Dr. Ajay K. Katangur
Committee Member
ABSTRACT

A university may be very large having multiple campuses and numerous buildings. Over time, new buildings are built making it difficult for new students, faculty, and visitors to find an office or location in the university. Likewise, some departments may be relocated in the campus which is a problem for existing students and faculty as well. There are no facilities available to easily locate administrative buildings, departments, library, food court, etc. in the campus from user’s current location. To address this issue, location-based services can be used, due to their profound benefits and usage in real-life applications. Based on this study, a Global Positioning System (GPS) based navigation system for Texas A&M University-Corpus Christi (TAMUCC) campus is developed. This Google Map based application on Android platform recognizes specific location, destination, and suggests the best possible way to reach the desired destination thus making it easy to navigate buildings within the campus. Also, this application can be used to broadcast events on the campus map which can reduce the need of physical posters. Consequently, this application helps navigating around the TAMUCC campus in a more convenient and timely way.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>ii</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td>iii</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>v</td>
</tr>
<tr>
<td>1 BACKGROUND AND RATIONALE</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Global Positioning System (GPS)</td>
<td>2</td>
</tr>
<tr>
<td>1.2 Android</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Android Studio</td>
<td>3</td>
</tr>
<tr>
<td>1.4 Google Maps</td>
<td>3</td>
</tr>
<tr>
<td>1.5 Related work</td>
<td>4</td>
</tr>
<tr>
<td>1.6 Solution: IslanderGo</td>
<td>6</td>
</tr>
<tr>
<td>1.6.1 Functionalities of the proposed application</td>
<td>6</td>
</tr>
<tr>
<td>2 SYSTEM DESIGN</td>
<td>8</td>
</tr>
<tr>
<td>2.1 Software Requirements</td>
<td>8</td>
</tr>
<tr>
<td>2.2 System Architecture</td>
<td>9</td>
</tr>
<tr>
<td>2.2.1 Modules Description</td>
<td>11</td>
</tr>
<tr>
<td>2.2.1.1 View/UI</td>
<td>11</td>
</tr>
<tr>
<td>2.2.1.2 REST Service</td>
<td>11</td>
</tr>
<tr>
<td>2.2.1.3 Shortest Path Algorithm</td>
<td>11</td>
</tr>
<tr>
<td>2.2.1.4 Ray casting algorithm</td>
<td>12</td>
</tr>
<tr>
<td>2.2.1.5 Apache Tomcat</td>
<td>12</td>
</tr>
<tr>
<td>2.2.1.6 JSON</td>
<td>12</td>
</tr>
<tr>
<td>2.2.1.7 SQLite</td>
<td>13</td>
</tr>
<tr>
<td>2.2.1.8 Amazon Web Services</td>
<td>13</td>
</tr>
<tr>
<td>2.2.1.9 Google Maps API</td>
<td>13</td>
</tr>
<tr>
<td>2.2.1.10 Google Places API</td>
<td>13</td>
</tr>
<tr>
<td>2.2.1.11 Google Geolocation API</td>
<td>14</td>
</tr>
<tr>
<td>2.3 Use Case Diagram</td>
<td>14</td>
</tr>
<tr>
<td>2.4 Class Diagram</td>
<td>16</td>
</tr>
<tr>
<td>CHAPTER</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>------</td>
</tr>
<tr>
<td>2.5 User Interface</td>
<td>18</td>
</tr>
<tr>
<td>2.5.1 Login Screen</td>
<td>19</td>
</tr>
<tr>
<td>2.5.2 Registration Screen</td>
<td>20</td>
</tr>
<tr>
<td>2.5.3 Home Screen</td>
<td>21</td>
</tr>
<tr>
<td>2.5.4 Menu</td>
<td>22</td>
</tr>
<tr>
<td>2.5.5 Admin Interface</td>
<td>23</td>
</tr>
<tr>
<td>3 SYSTEM IMPLEMENTATION</td>
<td>25</td>
</tr>
<tr>
<td>3.1 Setting required permissions</td>
<td>25</td>
</tr>
<tr>
<td>3.2 MD5 Hashing of account passwords</td>
<td>26</td>
</tr>
<tr>
<td>3.3 To find location of a user</td>
<td>27</td>
</tr>
<tr>
<td>3.4 Ray Casting Algorithm</td>
<td>28</td>
</tr>
<tr>
<td>3.5 Fetch Google routes</td>
<td>29</td>
</tr>
<tr>
<td>3.6 Recording Routes</td>
<td>30</td>
</tr>
<tr>
<td>3.7 Velocity Finder</td>
<td>31</td>
</tr>
<tr>
<td>3.8 Distance calculation</td>
<td>32</td>
</tr>
<tr>
<td>3.9 A* Search Algorithm</td>
<td>33</td>
</tr>
<tr>
<td>4 EVALUATION AND RESULTS</td>
<td>34</td>
</tr>
<tr>
<td>4.1 Registration page</td>
<td>34</td>
</tr>
<tr>
<td>4.2 Login page</td>
<td>35</td>
</tr>
<tr>
<td>4.3 Search</td>
<td>36</td>
</tr>
<tr>
<td>4.3.1 Record Routes</td>
<td>37</td>
</tr>
<tr>
<td>4.3.2 Routes History</td>
<td>38</td>
</tr>
<tr>
<td>4.3.3 Events</td>
<td>39</td>
</tr>
<tr>
<td>4.3.4 Information</td>
<td>40</td>
</tr>
<tr>
<td>4.3.5 Settings</td>
<td>40</td>
</tr>
<tr>
<td>4.3.6 Logout</td>
<td>41</td>
</tr>
<tr>
<td>4.4 Validating Events</td>
<td>41</td>
</tr>
<tr>
<td>4.5 Validating Information</td>
<td>42</td>
</tr>
<tr>
<td>5 CONCLUSION AND FUTURE WORK</td>
<td>44</td>
</tr>
<tr>
<td>5.1 Conclusion</td>
<td>44</td>
</tr>
<tr>
<td>5.1.1 Future Work</td>
<td>44</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>46</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>System Architecture of IslanderGo</td>
</tr>
<tr>
<td>2</td>
<td>Use Case diagram for User in IslanderGo Application</td>
</tr>
<tr>
<td>3</td>
<td>Use Case diagram for Admin</td>
</tr>
<tr>
<td>4</td>
<td>Class diagram of the application</td>
</tr>
<tr>
<td>5</td>
<td>Welcome Screen</td>
</tr>
<tr>
<td>6</td>
<td>Login screen</td>
</tr>
<tr>
<td>7</td>
<td>Registration screen</td>
</tr>
<tr>
<td>8</td>
<td>Home screen</td>
</tr>
<tr>
<td>9</td>
<td>Menu screen</td>
</tr>
<tr>
<td>10</td>
<td>Admin screen for <em>Events</em></td>
</tr>
<tr>
<td>11</td>
<td>Admin screen for <em>information</em></td>
</tr>
<tr>
<td>12</td>
<td>Setting permissions in AndroidManifest.xml file</td>
</tr>
<tr>
<td>13</td>
<td>MD5 Hashing Algorithm</td>
</tr>
<tr>
<td>14</td>
<td>finding last available location of user</td>
</tr>
<tr>
<td>15</td>
<td>Ray Casting algorithm to find users current location with respect to nearest building</td>
</tr>
<tr>
<td>16</td>
<td>Fetching Google routes</td>
</tr>
<tr>
<td>17</td>
<td>Recording operations</td>
</tr>
<tr>
<td>18</td>
<td>Velocity finder to find users velocity while recording</td>
</tr>
<tr>
<td>No.</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>19</td>
<td>Distance calculations</td>
</tr>
<tr>
<td>20</td>
<td>To find nearest point to fetch routes</td>
</tr>
<tr>
<td>21</td>
<td>Test case with empty field</td>
</tr>
<tr>
<td>22</td>
<td>Test case using same email id</td>
</tr>
<tr>
<td>23</td>
<td>Test case with empty field</td>
</tr>
<tr>
<td>24</td>
<td>Test case with invalid credentials</td>
</tr>
<tr>
<td>25</td>
<td>Search screen before selecting destination</td>
</tr>
<tr>
<td>26</td>
<td>Search screen when destination is selected</td>
</tr>
<tr>
<td>27</td>
<td>During recording</td>
</tr>
<tr>
<td>28</td>
<td>Route recorded</td>
</tr>
<tr>
<td>29</td>
<td>Routes History</td>
</tr>
<tr>
<td>30</td>
<td>Events list</td>
</tr>
<tr>
<td>31</td>
<td>Event indicated with color change</td>
</tr>
<tr>
<td>32</td>
<td>Information list</td>
</tr>
<tr>
<td>33</td>
<td>Settings Screen</td>
</tr>
<tr>
<td>34</td>
<td>Test case with adding an event with no time</td>
</tr>
<tr>
<td>35</td>
<td>Test case for empty field</td>
</tr>
</tbody>
</table>
CHAPTER 1

BACKGROUND AND RATIONALE

With the increase in technological advancements, mobile devices have progressed significantly with respect to performance parameters including memory capacities, advanced processing power etc. Within this context, the industry of geolocation using mobile devices has been enormously growing in the last decade. In fact, Location-based services (LBS) use real-time geolocation data from a mobile device or smart-phone to provide information, entertainment, and security [16][13].

In general, Location-based services (LBS) provide personalized services to the mobile users according to their current location. LBS uses Geographical Information System (GIS) to provide all the valuable features and indeed, people can track their current location and navigate from one location to another easily. Location can be traced via many technologies like cell identification, GPS, various radio-location systems, accelerometers, and electronic compass etc[9]. Out of all, GPS is the most accurate technique in measuring coordinates of a location[1] [15]. The navigation becomes easier with the help of Google Maps on GPS enabled mobile devices. GPS application allows users to search for a destination based on their current location. However, it shows significant performance only in outdoor environments.

Likewise, universities involve complex structure of buildings and pathways which is difficult for new students, faculty, and visitors to search for places. In fact, even most of the students who are familiar with the campus find it difficult to understand the entire picture of campus buildings in detail. There are many applications and commercial devices that provide driving directions and navigation such as Waze Navigator, Google Maps, Garmin Navigation devices, but their capabilities
are limited on a university campus [14].

Timely information is important in today’s life. For example, traffic information, information about accidents, road maintenance are known in advance. Similarly, for a Digital campus, timely information plays crucial role like where and when a soccer match will be played, place, time and topic of a seminar etc. These features are unavailable in any of the existing navigation applications. Therefore, its important to have a campus navigation application which can be accessed through mobile devices for navigation purpose and also for broadcasting various events taking place on the campus. In later sections, GPS is discussed in detail followed by Android, Android studio and Google maps.

1.1 Global Positioning System (GPS)

Global Positioning System is a network of orbiting satellites that sends precise details of the position of objects space in relevance to earth. The signals are obtained by GPS receivers and are used to interpolate the exact location, speed and time of any object/vehicles position. The GPS receivers must receive the signals from at least four satellites in an orbit. Each of the satellites will transmit the microwave signals to the GPS receiver in order to receive the distance between them. The triangulation technique is then used to increase the accuracy of the position estimation.

1.2 Android

Android mobile is the most popular in the smart phone market as it is an open source mobile operating system based on Linux with java support[4]. Android is a software stack for mobile devices that includes an operating system, middleware, and key applications which is developed and maintained by Google. It is one of the most
used mobile operating systems in today's world. As it is an open source operating system, developers most likely prefer Android over iOS to develop applications. In contrast, iOS is specific only to the iPhone users and is not an open-source platform. Many versions of Android have been released to date, each having several updates. Each new version is named after a dessert, going in alphabetical order: cupcake (Version 1.5) followed by Donut (Version 1.6), clair (Version 2.0 - 2.1), Froyo (Version 2.2), Gingerbread (Version 2.3), Honeycomb (Version 3.0 - 3.2), Ice Cream Sandwich (Version 4.0), Jelly Bean (Version 4.1- 4.3), KitKat (Version 4.4), Lollipop (Version 5.0 - 5.1), Marshmallow (6.0-6.0.1), Nougat (7.0 - 7.1.1) and latest version being Oreo (8.0-8.1). The system as such comes with many built-in APIs and because of it being open source, various applications are being developed depending on the needs and are being put on the Play Store for everyone to use[4].

1.3 Android Studio

Android Studio is Android’s official integrated development environment (IDE) based on IntelliJ IDEA and available for download on Windows, macOS and Linux. It is built for Android to accelerate development and help in building the highest-quality applications for every Android device. It offers tools custom-tailored for Android developers, including rich code editing, debugging, testing, and profiling tools.

1.4 Google Maps

Google Maps is a web mapping service developed by Google which is also available both on Android and iOS platform. It offers satellite imagery, street maps, 360 panoramic views of streets (Street View), real-time traffic conditions (Google Traf-
fic), and route planning for traveling by foot, car, bicycle, or public transportation[5]. The Google Maps application on Android and iOS have many features, including turn-by-turn navigation and public transit information.

1.5 Related work

Trend of location based navigation with the help of Google Map in android platform is growing faster. An application named Guide-My-Tour [17] has been developed by the researchers. This map indicates and tracks a users current location and heading direction. It can be zoomed in and out, or rotated in any direction. It combines a traditional paper map image of a locality with a satellite map image. While user walking information updated regularly following the user movement.

A Campus Assistant Application is designed and developed for Boca Raton Campus of Florida Atlantic University on Android Platform [3]. The application provides facility to choose starting and destination location and it gives shortest driving and walk able path and it also provides suitable parking lots for the user. The application provides rerouting if the user departs from the projected path. They have designed a Map Editor tool to edit and manage campus maps. Another research team added contextual information in the path of destination from users current location for more assistance of user with this application.

An application named Mobile Campus was designed and developed for SRM University campus [2]. This campus tour guide application will work on NFC (Near field communication) enabled smart phones on Android platform. It can be used by visitors, students and parents. This application includes basic functionality.

A comprehensive guiding and navigation services on Smart Phone has been developed for 2013 Taiwan Lantern Festival [12]. This application provides guidance
and navigation service with a custom map, not simple Google map. This application gives content information of 3D animation which is a digitized version of a water painting. It converts public map onto custom map in graphic format. The custom map is stored in KMZ format, which is a compressed format of KML files. When the visitor is interested in the POI (Point of Interest), he/she may activate the guiding service.

A map navigation system is developed to solve travelling salesman problem on android using Google map and Google GeoCoder API [7]. A campus spatial information service system is developed using Google map. The System is mainly the combination of Google Maps and MySQL database. The system provides facility of interactive information such as the picture, description, link and the useful measurement tool. User can add his POI and information about POI to database and it provides option of taking feedback from user.

A useful model for the future digital campus of Wuhan University of Technology is implemented [8]. This is a kind of GIS based campus navigation system. The system is mainly developed for school teachers and students. It integrates schools teaching resources, infrastructure, services and other information using GIS technology and thus provides digital and intelligent information services to teachers and students.

A location based nearest ATM search is developed by a research team. Its an application of GPS based location tracking service [6]. In this application, a new algorithm for location tracking has been proposed and implemented. NFC has some shortcomings like it works in a distance of 4cm or less.

In order to overcome such limitations, the proposed application is designed in such a way that user can get event information from any distance, any place. All of
the existing system follows complex mechanism for storing and updating of locations which is resolved in this application. It provides a very rich and customized menu option to find location and shortest path from current location.

1.6 Solution: IslanderGo

As mentioned above, the primary intend of this application is to effectively navigate and know information on timely basis in a University. To enumerate, a GPS based navigation system for TAMUCC campus is developed on android platform. This application senses users current location, allows user to search for any destination and provides directions/navigation with an optimal path from source to destination by using A* search algorithm [11]. Recording functionality can be used to record paths which are alternate to existing paths. Also, it can be used to display information about various buildings and can also broadcast various events that take place on the campus.

1.6.1 Functionalities of the proposed application

- IslanderGo application effectively recognizes user’s current location on campus to pave the way for better navigation on a campus environment.

- Allows user to search for any destination on campus and provides best possible path from source to destination.

- User can record paths which are alternate to existing paths. This recorded path can be used by all the users using the application.

- This application returns more comprehensive information about the indicated buildings such as its name, description including short history, contact details,
working hours etc.

- Helps broadcasting various events that take place on the campus which reduces the need of physical posters.
CHAPTER 2

SYSTEM DESIGN

2.1 Software Requirements

The following system requirements are necessary to develop this application.

1. Operating System

• Windows (or)

• Macintosh OS X

2. Database

• Amazon Web Services: Amazon Web Services (AWS) is a web service and cloud computing resource provided by Amazon. It is an on-demand computing platform. AWS provides huge computing capacity faster and cheaper to many client companies.

• MySQL: MySQL is an open-source relational database management system. It was established in May 1995 to provide various database operations. It acts as a support for various applications where data retrieval plays a key role.

3. Integrated Development Environment

• Android Studio 2.3: Android Studio provides Gradle based support, Lint tools for performance analysis, rich layout editor for drag-and-drop UI components and an Android Virtual Device emulator.
• **Spring Tool Suite**: The Spring Tool Suite is an Eclipse-based development environment that is customized for developing Spring applications. It provides a ready-to-use environment to implement, debug, run, and deploy any Spring applications. Also, the Spring Tool suite supports application targeting to local, virtual and cloud-based servers.

4. Programming Language

• **JAVA SE 8**: Java is a general-purpose computer programming language that is an object oriented. It is platform independent that works on Java Virtual Machine (JVM). It includes features of Lambda Expressions, Annotations and Date and Time API.

5. Hardware

• 1GB RAM

• 4GB Internal storage

• Dual-core processor

• A mobile phone with android operating system version 5.0 or greater.

2.2 System Architecture

The system architecture of the application is shown in figure 1. The application comprises of the android user interface, SQLite database management system, Shortest path algorithm, REST services, JSON, MySQL database and Amazon Web Services.

In this application, the web server is implemented using REST web service and JSON parser is used for parsing web service. JSON objects are sent across
Figure 1. System Architecture of IslanderGo

communication channels for application in mobile devices. For data input, the client has an android user interface to perform activities. Service layer is used for getting the result from GSON and will perform various algorithms like Ray cast and A* and then sends the result to view. Apache HTTP client of PHP and Java REST client are used for writing code for connections. All requests that are received by the client are sent to the controller of web server through REST service. Service layer in the web server is used for implementing logics to perform distance calculations. DAO layer is used for triggering the database. The database actions are performed using MYSQL and SQLite databases. Finally, the data obtained by the JSON object is
displayed in the user interface of Android.

2.2.1 Modules Description

2.2.1.1 View/UI

The view is a user interface that is visible to the user and has an ability to perform some functionalities. Views are developed by making use of XML, activities, fragments, and adapters. Requests can be sent, and responses are visible to users with the help of view.

2.2.1.2 REST Service

Representational State Transfer also commonly known as the RESTful service is a web service used to provide the interoperability between the various computer systems available on the internet. In a RESTful web service, the requests made to a resources URI will accept any request whether it is in XML, HTML, JSON or some other defined formats. Such a stateless, RESTful service aims at achieving fast performance, reliability, and the ability to grow without affecting the system on its own.

2.2.1.3 Shortest Path Algorithm

A* is a search algorithm [11], or a best-first search, that solves problems by searching among all possible paths to the solution (destination) for the one that incurs the smallest cost (least distance travelled, shortest time, etc.), and among these paths it first considers the ones that appear to lead most quickly to the solution [11], it constructs a tree of paths starting from that node, expanding paths one step at a time, until one of its paths ends at the predetermined goal node. This algorithm is
implemented in this application to determine the best path from the set of recorded routes and the routes obtained from google maps.

2.2.1.4 Ray casting algorithm

Ray casting algorithm [10] is used to find whether the point (user) is inside or outside a simple polygon (building). It works on testing how many times a ray, starting from the point and going in any fixed direction, intersects the edges of the polygon. If the point is on the outside of the polygon the ray will intersect its edge an even number of times. If the point is on the inside of the polygon then it will intersect the edge an odd number of times [10]. This is how we identify if the user is inside/outside a building and to which building is his current location closest to.

2.2.1.5 Apache Tomcat

Apache Tomcat also known as Tomcat Server, is an open source Java Servlet Container developed by the Apache Software Foundation. This is used to implement several Java Enterprise Edition specifications including Java Servlets, Java Server Pages and Web Sockets. It works on HTTP web server technology.

2.2.1.6 JSON

JSON, JavaScript Object Notation, is an open standard format that is used to transmit the data in the form of key-value pairs. It is the most common data format used for asynchronous browser/server communication.
2.2.1.7 SQLite

SQLite is a relational database management system. In contrast to many other database management systems, SQLite is not a client-server database engine. Rather, it is embedded into the end program for faster retrieval of data.

2.2.1.8 Amazon Web Services

Amazon Web Services (AWS) is a subsidiary of the leading online shopping company, Amazon.com. It offers a high-level Cloud Computing platform. Their services can be used for various services such as computing, analytics, deployment and mobile developer tools.

2.2.1.9 Google Maps API

The Google Maps JavaScript API lets developers to customize maps with their own content and imagery for display on web pages and mobile devices. The Google Maps JavaScript API features four basic map types (road-map, satellite, hybrid, and terrain) which can be modified using layers and styles, controls and events, and various services and libraries.

2.2.1.10 Google Places API

The Google Places Web Service is a service that returns information about a "place": an establishment, a geographic location, or prominent point of interest using an HTTP request. Place requests specify locations as latitude/longitude coordinates. Two basic place requests are available: a Place Search request and a Place Details request.
2.2.1.11 Google Geolocation API

The Google Maps Geolocation API returns a location and accuracy radius based on information about cell towers and WiFi nodes that the mobile client can detect. The current location of a user can be detected using this.

2.3 Use Case Diagram

A use case diagram shows the user interaction with the application system. The use case diagram shown in figure 2 represents the typical cases seen in the application. Overall it is used to describe functionalities of the system in the form of a graphical overview. The oval shapes show the features that are provided to the user.

Any user can use the application once they register successfully by providing basic information like Name, E-mail etc. The login functionality allows users to login into the application with the help of username/email and password which are set by the user during registration. The logout functionality is available to the user after they login. The user needs to grant permission to access user’s location and media files. User has access to more comprehensive information about the indicated buildings such as its name, description including short history, contact details, working hours etc with the information functionality. The search functionality displays a list of all buildings on the campus, so a user can choose the destination. Once the destination is set, all the available google routes and recorded routes are fetched. Recording functionality can be used to record specific routes which are alternate to existing paths. Three best routes are calculated from these set of routes using A* search algorithm and displayed to the user.
Figure 2. Use Case diagram for User in IslanderGo Application

The use case diagram for the functionalities of admin are represented in figure 3. Admin can add/update/delete building information based on the changes made, admin can also add/update/delete events happening in the campus. All this information will be displayed to the user through the application interface.
2.4 Class Diagram

Figure 4 describes the structure of the system by making use of classes of the system, their attributes, methods, and relation between the objects.
Figure 4. Class diagram of the application
2.5 User Interface

User interface is where the user interacts with the system. Figure 5 shows the welcome screen when a user clicks on the application icon. There are two options for the user, Signup or Login. The overall execution flow of the developed IslanderGo application is described in this section.

Figure 5. Welcome Screen
2.5.1 Login Screen

The login screen is used for user authentication to prevent unauthorized application use. Figure 6 shows the users login screen. The user must login with the email and password set during registration to login to the application. When user logins with the right credentials, access is granted to the application.

![Login Screen](image)

**Figure 6.** Login screen
2.5.2 Registration Screen

Figure 7 shows the required fields for registering a new user in the application. If the user is using the application for the first time, the user needs to register in the application by providing basic details such as Name, Email id and set unique username and password combination.

![Registration Screen]

Figure 7. Registration screen
2.5.3 Home Screen

Figure 8 shows the main screen/first screen when the user logs into the application. The Home Screen shows a detailed campus map of the university. It displays all the buildings on the campus which are responsive to touch. If a user touches on specific building, he can see list of options like information, and routes to reach that building.

The float button at the bottom right corner, helps to refresh the current location of a user and sets the source field. When a destination is picked, it is shown in the destination field. The hamburger button on the upper left corner displays the menu of the application.

Figure 8. Home screen
2.5.4 Menu

Figure 9 shows some of the functionalities of the application. At the top of the menu, the username of the user is displayed and all the functionalities are listed below. *Search* displays a list of all the buildings on the campus from where a user can choose a destination. *Record Routes* allows user to record alternate paths from a source to destination. *Routes History* displays statistics of recorded routes by the user. *Events* contain the information and place of the event, change in color of the building indicates an event. In the *information* icon, more details about the buildings are provided. *Settings* is useful to provide users with functionalities to switch map type, and to monitor their search history.

![Menu screen](image)

**Figure 9.** Menu screen
2.5.5 Admin Interface

Administrator can add, update, or delete any event in the IslanderGo web page as shown in fig 10. As soon as administrator updates an event, it gets displayed in the events list of the user interface. Similarly, administrator can also add, update and delete information about the buildings on campus which is displayed in information screen as shown in figure 11.

![Admin screen for Events](image)

**Figure 10.** Admin screen for Events
Figure 11. Admin screen for information
CHAPTER 3

SYSTEM IMPLEMENTATION

3.1 Setting required permissions

Figure 12 shows the permissions that are to be set from the AndroidManifest.xml file. This application requires internet to connect to web server, location access to track the accurate location of the user and to write and read to external storage for obtaining and storing the building information.

```xml
<uses-permission android:name="com.example.canquemap.permission.MAPS_RECEIVE" />
<uses-permission android:name="android.permissionINTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
</!

The following two permissions are not required to use
Google Maps Android API v2, but are recommended.

-->
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />

<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />

Figure 12. Setting permissions in AndroidManifest.xml file
3.2 MD5 Hashing of account passwords

The most important security aspect of the application lies with securing the account passwords of the users. To secure these passwords, MD5 Hashing algorithm is implemented to store the passwords in the database as shown in figure 13. The MD5 hashing algorithm is a one-way cryptographic function that accepts a message of any length as input and returns as output a fixed-length digest value to be used for authenticating the original message.

```java
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

public class Md5 {
    public static String hashPassword(String password) {
        MessageDigest md = null;
        try {
            md = MessageDigest.getInstance("MD5");
        } catch (NoSuchAlgorithmException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        md.update(password.getBytes());
        byte byteData[] = md.digest();
        StringBuffer sb = new StringBuffer();
        for (int i = 0; i < byteData.length; i++) {
            sb.append(Integer.toString((byteData[i] & 0xff) + 0x100, 16).substring(1));
        }
        return sb.toString();
    }
}
```

**Figure 13.** MD5 Hashing Algorithm
3.3 To find location of a user

This method is used to find user’s last available location using location provider API as shown in figure 14. Location provider uses GPS to

```java
private Location getLastKnownLocation() {
    lm = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
    List<Provider> providers = lm.getProviders();
    Location bestLocation = null;
    for (String provider : providers) {
        if (ContextCompat.checkSelfPermission(this, android.Manifest.permission.ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(this, new String[]{android.Manifest.permission.ACCESS_FINE_LOCATION},
                    PackageManager.PERMISSION_GRANTED);
        }
        Location l = lm.getLastKnownLocation(provider);
        if (l != null) {
            continue;
        }
        if (bestLocation == null || l.getAccuracy() < bestLocation.getAccuracy()) {
            // Found best last known location is "l".
            bestLocation = l;
        }
    }
    return bestLocation;
}
```

**Figure 14.** finding last available location of user
3.4 Ray Casting Algorithm

Ray casting algorithm is used to find whether the point (user) is inside or outside a simple polygon (building). It works on testing how many times a ray, starting from the point and going in any fixed direction, intersects the edges of the polygon. If the point is on the outside of the polygon the ray will intersect its edge an even number of times. If the point is on the inside of the polygon then it will intersect the edge an odd number of times. This is how we identify if the user is inside/outside a building and to which building is his current location closest to and display this as a toast to user as shown in figure 15.

```java
public boolean pointIsInPolygon(LatLng touchPoint) {
    int i;
    int j;
    boolean result = false;
    System.out.println("BUILDING SET" + buildingSet.values().size());
    for (Building b : buildingSet.values()) {
        LatLng[] bp = b.getPoints();
        for (i = 0, j = bp.length - 1; i < bp.length; j = i++) {
            if ((bp[i].longitude > touchPoint.longitude) != (bp[j].longitude > touchPoint.longitude) &&
                (touchPoint.longitude - bp[i].longitude) * (touchPoint.latitude - bp[i].latitude) +
                (bp[j].longitude - bp[i].longitude) * bp[i].latitude) {
                result = !result;
            }
        } if (result) {
            currentTouchedBuilding = b;
            return result;
        }
    }
    return result;
}
```

**Figure 15.** Ray Casting algorithm to find users current location with respect to nearest building
3.5 Fetch Google routes

The following figure 16 illustrates how Places API is called to fetch the available Google routes.

```java
public Document getDocument(LatLng start, LatLng end, String mode) {
    String url = "http://maps.googleapis.com/maps/api/directions/xml?";
    + "origin=" + start.latitude + "," + start.longitude
    + "&destination=" + end.latitude + "," + end.longitude
    + "&sensor=false&units=metric&mode=" + mode;

    System.out.println(url);
    try {
        URL weburl = new URL(url);
        InputStream in = weburl.openStream();
        DocumentBuilder builder = DocumentBuilderFactory.newInstance()
            .newDocumentBuilder();
        Document doc = builder.parse(in);
        return doc;
    } catch (Exception e) {
        e.printStackTrace();
        return null;
    }
}
```

**Figure 16.** Fetching Google routes
3.6 Recording Routes

As shown in figure 17 is a utility code for recording operations. User’s point of location will be captured for every three seconds. These points will be processed further for various calculations and will be saved in the server.

```java
public void processRecord_kalman_filter(String beta, boolean firstTime) {
    try {
        String tmpF;
        if (firstTime) {
            tmpF = filePath;
        } else {
            tmpF =filePath p;
        }
        BufferedReader = new BufferedReader(new FileReader(tmpF));
        String line = bufferReader.readLine();
        file_p_Initialization( extension: "txt", fileName, beta, append: false); // save as
        while (line != null) {
            String[] tmp = line.split(" \t");
            VelocityFinder kf = new VelocityFinder();
            Location_Tamucc current;
            for (int i = 0; i < tmp.length; i++) {
                current = new Location_Tamucc(tmp[i]);
                kf.calculate(current.getX(), current.getY(), accuracy: 1,
                current.getZ());
                appendDataToFile_p(kf.toString());
            }
            line = bufferReader.readLine();
        }
        System.out.println("Processed Kalman Filter!");
        bufferReader.close();
        bufferWriter_p.close();
        fileWriter_p.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

Figure 17. Recording operations
3.7 Velocity Finder

It is important to find velocity of a user during recording so it can be used to compare against the Google provided average walking velocity to find optimum paths. In order to find the velocity of the moving object, we divide the change in position by the change in time with respect to the direction moved as shown in figure 18.

```java
public void calculate(double lat_measurement, double lng_measurement,
                      float accuracy, long unixTimeStamp) {
    float minimum = 1;
    if (accuracy < minimum)
        accuracy = minimum;
    if (variance < 0) {
        this.unixTimeStamp = unixTimeStamp;
        lat = lat_measurement;
        lng = lng_measurement;
        variance = accuracy * accuracy;
    } else {
        long timeinMillis = unixTimeStamp
           - this.unixTimeStamp;
        if (timeinMillis > 0) {
            variance += timeinMillis * metresPerSecond
                        * metresPerSecond / 1000;
            this.unixTimeStamp = unixTimeStamp;
        }
        float K = variance / (variance + accuracy * accuracy);
        lat += K * (lat_measurement - lat);
        lng += K * (lng_measurement - lng);
        variance = (1 - K) * variance;
    }
```

Figure 18. Velocity finder to find users velocity while recording
3.8 Distance calculation

As shown in figure 19 is used to calculate distance between different points consisting latitude and longitude.

```java
private double pointToLineDistance() {
    double normalLength = Math.sqrt((closest_2.latitude - closest_1.latitude) * (closest_2.latitude - closest_1.latitude) + (closest_2.longitude - closest_1.longitude) * (closest_2.longitude - closest_1.longitude));
    return Math.abs((origin.latitude - closest_1.latitude) * (closest_2.longitude - closest_1.longitude) - (origin.longitude - closest_1.longitude) * (closest_2.latitude - closest_1.latitude)) / normalLength;
}

private double cosDegree(double origin_closest1, double origin_closest2, double closest1_closest2) {
    double cosD = (Math.pow(origin_closest1, 2) + Math.pow(closest1_closest2, 2) - Math.pow(origin_closest2, 2)) / (2 * origin_closest1 * closest1_closest2);
    return cosD;
}

private double getEDistance(LatLng a, LatLng b) {
    double result = Math.pow(a.latitude - b.latitude, 2) + Math.pow(a.longitude - b.longitude, 2);
    result = Math.sqrt(result);
    return result;
}
```

Figure 19. Distance calculations
3.9 A* Search Algorithm

A* is a search algorithm used to determine the best paths from the set of recorded routes and the routes obtained from Google maps as shown in figure 20.

```java
public int getNearestPointForTwo(LatLng myLatlng, ArrayList<LatLng> rroute,
                                BuildingDrawing bd) {

double min = Double.MAX_VALUE;
LatLng tmpLL;
int minIndex = Integer.MAX_VALUE;
for (int i = 0; i < rroute.size() / 2; i++) {
    tmpLL = rroute.get(i);
    double current = getDistance(myLatlng, tmpLL);

    if (!bd.pointIsInPolygon(tmpLL)) {
        if (current < min) {
            min = current;
            minIndex = i;
        }
    }
}
return minIndex;
}
```

**Figure 20.** To find nearest point to fetch routes
CHAPTER 4

EVALUATION AND RESULTS

This chapter includes testing of all the functionalities and validation of the application in various scenarios.

4.1 Registration page

The user needs to fill all the fields in the registration page to login to the application and failure to do so will result in warning text indicating required field to be filled as shown in figure 21. If a user tries to register second time with the same email id, a message will be prompted stating that the email id already exists as shown in figure 22.

![Figure 21. Test case with empty field](image1)

![Figure 22. Test case using same email id](image2)
4.2 Login page

If the user is using the application on the same device, the user will not be asked to login until the user logs out of the application as it has an option to save the credentials. If the user happens to logout, he again needs to login using the email and password. If either of the fields are left incomplete, a warning message is displayed as shown in figure 23. Figure 24 illustrates the test case when user enters incorrect credentials, a message will be displayed prompting the user to try again with correct set of credentials.

**Figure 23.** Test case with empty field

**Figure 24.** Test case with invalid credentials
4.3 Search

*Search* displays a list of all the buildings on the campus as shown in figure 25, so a user can easily choose the destination from there. If the user is sure about the destination, he can directly type in the field at the top, or the user can pick from the list of buildings by clicking on the *search* button next to the building name. Once the destination is set, three best routes are suggested to the user along with the time required for each route as shown in figure 26. User can pick any one of the three routes and start walking towards the destination.

![Figure 25. Search screen before selecting destination](image1)

![Figure 26. Search screen when destination is selected](image2)
4.3.1 Record Routes

*Record Routes* can be used to record specific routes which are alternate to existing paths. When a user hits record button as shown in figure 27, Ray-casting algorithm recognizes if the user is inside/outside a building and detects the nearest building based on users current location and displays that information as a toast to the user.

![Figure 27. During recording](image)

![Figure 28. Route recorded](image)

In addition, Android services are used to capture coordinates and time for every three seconds while the user is walking towards the destination. This collected information is used to calculate displacement, total time, and average velocity. Furthermore, the average velocity computed is compared with the Google average walking velocity to calculate the optimum time required to travel from source to destination. All the average velocities and paths from various users are stored in
Android file using Android file API. The recorded path is displayed to the user once the user hits *stop recording* as shown in figure 28 and the data is uploaded to the server using MIME type so it can be useful to all the other users as well.

4.3.2 Routes History

When a user records routes, all the routes will be displayed to the user in the form of a list in the *Routes History* as shown in figure 29. This provides the user with a feature to monitor their recorded routes and navigate using that route. The user can also delete the set of recorded routes. The option to delete recorded routes is provided in the settings screen of the user interface.

![Figure 29. Routes History](image)
4.3.3 Events

Any event taking place on the campus can be broadcasted in the developed application which can reduce the need of physical posters. As shown in fig 31, a change in color of the building indicates that an event is taking place in that particular building. If the user clicks on that building, more information regarding the event can be found.

If the user wants to know about events which had already happened or the events which will take place in future, they can go to the Events screen which displays a list of all events that are scheduled for the day and also retains information of the events that took place earlier as shown in figure 30.

![Figure 30. Events list](image1)

![Figure 31. Event indicated with color change](image2)
4.3.4 Information

The information screen provides more comprehensive information about the indicated buildings such as its name, description including short history, contact details, working hours etc as shown in figure 32. This feature helps new students, visitors to know more about the college as well as reduces number of tour guides on campus.

Figure 32. Information list

4.3.5 Settings

The settings screen has options to clear users search query history, and to delete all routes history. The user has an option to switch map type to Satellite, Hybrid and Roadmap. Figure 33 shows the Settings screen.
4.3.6 Logout

The user can click on log out button present in the menu to logout of the application.

4.4 Validating Events

Admin has the privileges to add, update and delete events in the web page of the application. Admin needs to fill in all the fields of the Event page like Event name, building, date and time. If one of the fields is left empty, it will result in failure and a warning text will be prompted asking the admin to fill the required fields as shown in figure 34.
4.5 Validating Information

Admin has the privileges to add, update and delete information regarding various buildings in the campus through the web page of the application. Admin needs to fill in all the fields of the Add Info page such as building and its information. If one of the fields is left empty, it will result in failure and a warning text will be prompted asking the admin to fill the required fields as shown in figure 35.
Figure 35. Test case for empty field
CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

IslanderGo is an application developed for Android devices. It is a Global Positioning System (GPS) based navigation system for Texas A&M University-Corpus Christi (TAMUCC) campus. This Google Map based application provides more comprehensive information of any building in the campus, recognizes current location of a user, allows user to search for any destination in the campus, and suggests the best possible way to reach the desired destination thus making it easy to navigate from building to building within the campus. Also, this application can be used to display event information and updates on the map view of the application which can reduce the need of physical posters. Consequently, the developed application helps navigating around the TAMUCC campus in a more convenient and timely way.

5.1.1 Future Work

By adding the following functionalities, this application can be improved further.

- Audio Controls: If speech activated control is added to this application, visually impaired people can be benefited. Frequent audio alerts on changing location can direct them to proper locations. This can reduce their stress for searching a place or to reach any place.

- Multimedia based Advertisements: Rich multimedia-based advertisements like audio/video promotion of any event can be added in the map with proper
location of the event.

- **Warning system**: Helps broadcasting/receiving alert texts during emergency situations on campus, based on user’s current location.

- **Improved search list**: Buildings can be categorized and grouped into the respective departments which helps a prospective student to take a campus tour at his field of interest at ease.
REFERENCES


[8] Huang, J., Zhan, Y., Cui, W., Yuan, Y., and Qi, P. Development of a campus information navigation system based on gis. In *Computer Design and


