A Location-Based Service Using a Server-Side Geographical Database

AbdElRahman Ahmed El-Said, Wen-Chen Hu, and Fatima El Jamiy
Department of Computer Science
University of North Dakota
Grand Forks, ND 58202
abdelrahman.elsaid@und.edu, wenchen@cs.und.edu,
eljamiyfatima@gmail.com

Naima Kaabouch
Department of Electrical Engineering
University of North Dakota
Grand Forks, ND 58202
naima.kaabouch@engr.und.edu

Abstract

One of the most popular apps is location-based services (LBSs) such as navigation, locating local events, and checking location-based advertisements. However, building an LBS is not a simple task because it involves various subjects and techniques like mobile computing, databases, and security and privacy. One of the major LBS components is geographical databases, which are used to store geographic data like locations and functions (such as restaurants and gas stations). A geographical database is usually hosted on a server because of its huge size and should facilitate geographic data storage, indexing, searching, and matching. This paper tries to mitigate the high difficulty of LBS construction by showing the construction step by step with a focus on connecting a mobile device to a server-side geographical database. After reading this article, readers will be able to build an LBS prototype for their research or applications.

1 Introduction

Smartphones are the most popular electronic devices nowadays. People use them to perform daily activities, like managing messages and emails, browsing the Internet, and watching video, anywhere and anytime. One of the popular apps is location-based service (LBS), which is a service based on the geographical position of a mobile handheld device. One of the LBS examples is to find a nearby ethnic restaurant by using a smartphone. According to the market research reports, (i) the global LBSs market was valued at $11,994 million in 2016, and is expected to reach $61,897 million by 2022, supported by a growth rate of 26.6% (Malani, 2017), and (ii) the growing trend for the integration of location-based search with social networking websites will post a growth rate of close to 40% for the global LBS market during 2017-2021 (Technavio, 2017). They show the high importance and popularity of LBSs. However, building an LBS is not a simple task. This research tries to help developers by showing how to build an LBS.

Building a location-based system (LBS) is not easy because various disciplines and subjects are involved. This pper tries to mitigate the high complication and difficulty by showing how to build an LBS from the ground up with a focus on connecting the mobile devices to a server-side geographical database. The five major components of an LBS system and our methods for implementing them are listed below:

- *Mobile handheld devices*: Android emulator (AVD), instead of a smartphone, is used for development.
- *Positioning systems like satellites*: The DDMS (Dalvik Debug Monitor Server) of an AVD is employed to send the mock locations, instead of using actual road testing.
- *Mobile and wireless networks*: URL connection, provided by Android, is used to connect the emulator to the service provider.
- *Service providers*: The LBS is to show a direction between the user and the nearest participant.
- *Geographical databases*: A small geographical database is used by the service provider.

Readers, who are interested in LBS research and applications, can use this paper to build their LBS prototypes. Based on the prototypes, they can realize their LBS research and show the advantages of their proposed methods.

The rest of this paper is organized as follows. Section 2 introduces related software, tools, and services including (i) Android and Android Studio, (ii) MySQL and MySQL Workbench, and (iii) various location-based services. The proposed LBS app is introduced in Section 3, which includes three sub-sections: (i) the proposed system, (ii) the server-side geographical database used by the proposed system, and (iii) the result screenshots. Section 4 explains how to build the proposed system including three sub-section: (i) Android server connection, (ii) client-side code, and (iii) server-side code. The final section summarizes this study and gives possible LBS projects.

2 Related Software, Tools, and Services

This section introduces the software and tools used to build the proposed system including (i) Android and Android Studio and (ii) MySQL databases and MySQL Workbench. In addition, various location-based services are listed to give readers an idea what the market trend of LBSs is.

2.1 Android and Android Studio

Android (Android, n.d.a) is the most popular mobile operating system as it is used by more than 80% of smartphones in 2017 (IDC, 2018). It is a software stack based on the Linux kernel for mobile devices that includes an operating system, middleware, emulators, and various key applications like phone, browser, and contacts. It was initially developed by Android, Inc., which was acquired by Google, Inc. in 2005. Since then, Android has been revised and enhanced significantly and constantly for the ever-changing newer features of mobile devices. In the past, Android app development in Eclipse with ADT (Android Development Tools) was highly recommended. Since 2014, Android Studio (Android, n.d.b) became the primary IDE (Integrated Development Environment) for native Android application development. It is freely available under the Apache License 2.0 based on JetBrains' IntelliJ IDEA software. Figure 1 shows a user interface of Android Studio and its major functions are listed as follows:

- *Attributes*, providing controls for the selected view's attributes,
- *Design editor*, displaying the layout in both the design and blueprint views,
- *Gradle console*, a screen for managing the automated build system for the Android source code,
- *Logcat*, a screen showing the logs generated from the running programs,
- *Menu bar*, providing the essential functions of Android Studio,
- *Palette*, providing a list of widgets and layouts that can be dragged into the layout in the editor,
- *Project structure*, showing the view hierarchy of the project, and
- *Toolbar*, providing buttons to configure the layout appearance in the editor and to change some layout attributes.

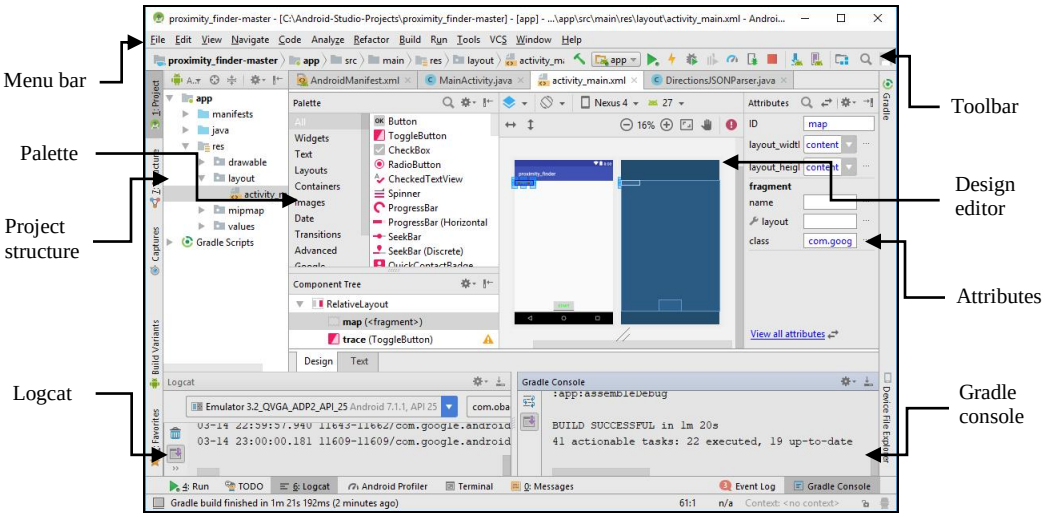


Figure 1: A User Interface of Android Studio.

2.2 MySQL Databases and MySQL Workbench

The proposed geographical database uses MySQL (MySQL, n.d.a), an open-source relational database management system (RDBMS). It is supported by a company named MySQL AB, which is a division of Oracle Corporation now. MySQL is one of the major components of LAMP (Linux, Apache, MySQL, and PHP/Perl/Python) technology for website construction. It is reliable because tens of thousands of websites have used MySQL without incidences and, even better, it is free because it is open source. In addition, most programming languages like PHP and Perl are allowed to embed SQL (Structured Query Language) commands in their programs for adding, accessing, and processing data in MySQL. This research uses Android Java to connect to a server-side PHP program with embedded SQL commands to access the geographical database using MySQL. Other than using MySQL text user interface to manage MySQL, MySQL Workbench (MySQL, n.d.b) is a database IDE (Integrated Development Environment) for the MySQL database systems. It allows a DBA or developer to visually design, develop, manage, and maintain databases. Figure 2 shows a user interface of the MySQL Workbench and its major components are introduced as follows:

- *Hide/show panels*, changing the layout of the Workbench,
- *Menu bar*, providing the essential functions of MySQL Workbench,
- *Navigator*, showing the schema of the user,
- *Object information*, displaying the details of the object,
- *Output*, containing the log of executed commands,
- *Result*, showing output from the SQL commands in the query panel,
- *SQL additions*, including context help and SQL snippets for saving SQL code,
- *SQL query panel*, allowing users to enter SQL commands, and
- *Toolbar*, providing buttons to quickly manage the SQL commands in the query panel.

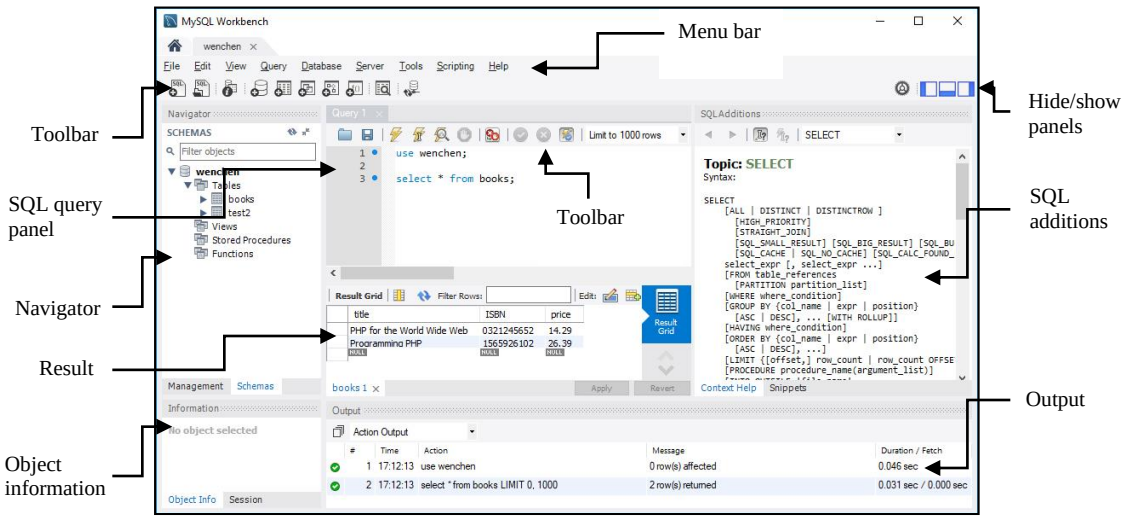


Figure 2: A User Interface of MySQL Workbench.

2.3 Location-Based Services (LBSs)

Various LBSs are available on the app stores and most major IT corporations provide their own LBSs. This sub-section introduces some of them, so readers can get a sense of what LBSs can affect our daily lives and implement their own LBSs based on this research:

- Foursquare (https://foursquare.com/), recommending interesting places like coffee houses, restaurants, and stores to users based on their previous browsing history, purchases, or check-in history.
- Uber (https://www.uber.com/), a form of sharing economy and of which major function is ridesharing among its users.
- Google Maps (https://maps.google.com/), a web mapping service developed by Google. Other than the traditional map services like street maps and route planning for traveling, it also provides various recommendations such as restaurants and hotels.
- Bing Maps (https://www.bing.com/maps/), provided by Microsoft including the features of directions, traffic information, and people, business, and location search.
- iOS – Maps (https://www.apple.com/ios/maps/), a map or location-based service provided by Apple, Inc. including the features of indoor maps, navigation, and proactive suggestions.
- AutoNavi (https://www.autonavi.com/), a provider of digital map, navigation, and LBS in China offering products with features such as route planning, travel navigation, and offline maps from the Alibaba Group.

3 Design of the Proposed Location-Based Service (LBS)

Various LBSs can be found from the app stores. Examples of the services include finding nearby ethnic restaurants, navigation, and recommending interesting places or events. Other than the app user, most participants of the services are static or fixed; i.e., they, like the restaurants and concerts, are not mobile. This research investigates a kind of LBSs of which the participants are mobile. The potential of this kind of LBSs are great, but at the same time, their realization is different from the one of the traditional LBSs. This section introduces the design of the proposed system and implementation of the system will be given in the next section.

3.1 The Proposed System

The proposed location-based service is to draw a direction between the user and the nearest mobile participant. The major difference between the proposed system and other LBS systems is the participants of the former system are mobile. For example, a student is looking for his/her friends to borrow a calculator right before an exam. Traditional way to solve this problem is to find the nearest place where his/her friends live. However, the problem is his/her friends may be on the move and their locations may be different from time to time. The services would not be effective if the mobility feature is not considered. One way to keep track of the participants' locations is having the participants send their locations to the server once they move and the locations are saved in a server-side geographical database. The service then retrieves the current locations from the database and draws a direction between the user and the nearest participant. The following steps show how the proposed service works:

1. Participants sign up the service.
2. Entries are created in tables of a server-side geographical database.
3. Participants send their locations to the server if their locations change.
4. The database is updated according to the received location data.
5. A user submits his/her current location and a request to find a nearest participant.
6. Based on the requester's current location and the location data in the geographical database, the service draws a direction between the user and the nearest participant.

Figure 3 gives another way to explain the proposed service by giving a workflow of the proposed system. It includes the following four steps: (a) participants sending their updated location data to the sever, (b) the user looking for a nearest participant, (c) the server finding the nearest participant according to the locations of user and participants, and (d) drawing a route between the user and the nearest participant.

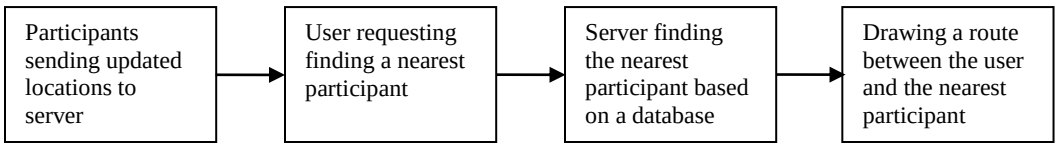


Figure 3: Workflow of the Proposed System.

3.2 The Proposed Server-Side Geographical Database

Geographical databases are databases storing geographic data including spatial data, like heights and locations, and attribute data like names such as McDonalds and functions such as restaurant. They are usually provided by a third party such as GeoNames (n.d.) because huge space is needed to store the data. This section proposes a small geographical database for facilitating geographic data storage, indexing, searching, and matching. This research uses a server-side MySQL (n.d.a) database to host the geographic data with an emphasis on mobile users. The database schema is shown in Table 1. It is made as simple as possible because the focus is on how to connect the mobile client to a server-side geographical database. The major difference between the proposed database and traditional databases is the entity could be mobile; i.e., its location could change from time to time. It consists of two tables: *entities* and *LBS* tables. The table *entities* include a column *time*, which is the time when the location is saved. The duration of the entity at the current location can therefore be found as the difference between the current time and the saved time. One of the functions of the duration is to check whether the location is valid. For example, if the duration of the location is more than a week, then the service may assume the mobile location is not valid. The other table *LBS* is used to find the participants of an instance of the service. One participant may join more than one instance of the service.

PID	Name	Function	Latitude	Longitude	Time
1	Poke Mon	male	47° 55’ 31” N	97° 2’ 57” W	2018-02-09 21:21:40
2	Italian Moon	restaurant	47° 55’ 40” N	97° 1’ 60” W	2010-10-29 06:28:42
3	Super Mario	male	47° 55’ 31” N	97° 1’ 52” W	2017-05-28 10:51:30
4	GI Jane	female	47° 55’ 60” N	97° 1’ 98” W	2018-02-29 11:18:42
5	Valley Dairy	gas station	47° 55’ 33” N	97° 2’ 53” W	2017-03-37 06:21:43
6	Turtle River	park	47° 54’ 72” N	97° 1’ 70” W	1991-12-20 04:40:42
7	GF Bus	transportation	47° 55’ 51” N	97° 1’ 54” W	2018-02-54 10:33:21
8	Florida Sunshine	farmers’ market	47° 55’ 31” N	97° 1’ 97” W	2018-06-29 19:21:44

LBS	PID
1	1
12	3
2	4
1	32
4	120
2	3
1	3
…	…

(b)

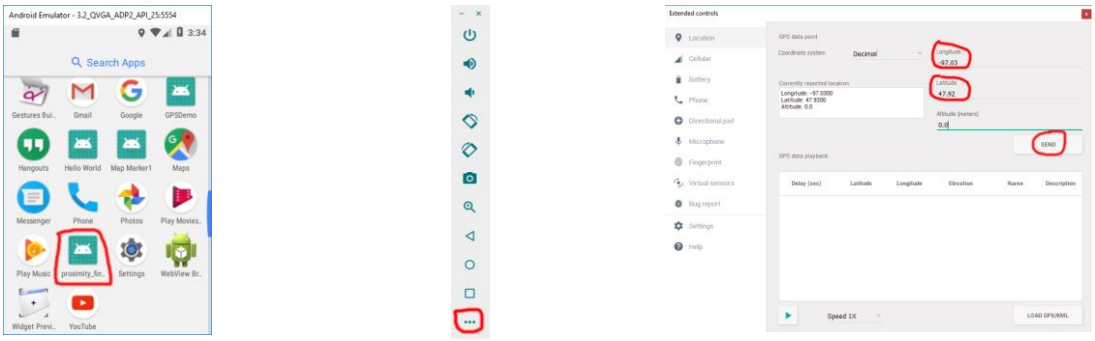
...	...	...	...	...	...

(a)

Table 1: Sample Values of the Proposed Geographical Database Schema Including (a) *entities* Table and (b) *LBS* Table.

3.3 The Result Screenshots

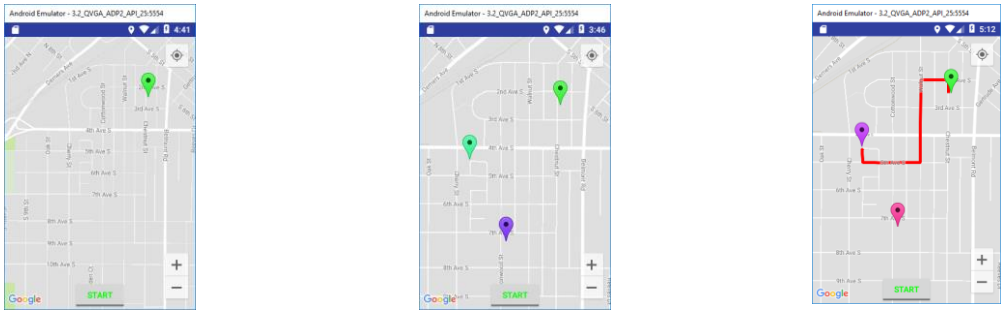
This sub-section shows some of the screenshots from this service. It is to help readers understand what the service can achieve before they study the implementation details given in the next section. Instead of doing actual road tests, this research uses Android AVD (Android Virtual Device) to simulate the testing. Figure 4 shows three screenshots from this app. Figure 4.a shows the app in the Android Launcher, Figure 4.b is the control panel associated with the emulator, and Figure 4.c is the extended controls after clicking the item at the bottom of the control panel. The extended controls include a location function, which allows users to enter location data including latitude, longitude, and attitude and submit it to the app. Using this function is similar to the app receiving location data from the GPS (Global Positioning System).



(a) (b) (c)

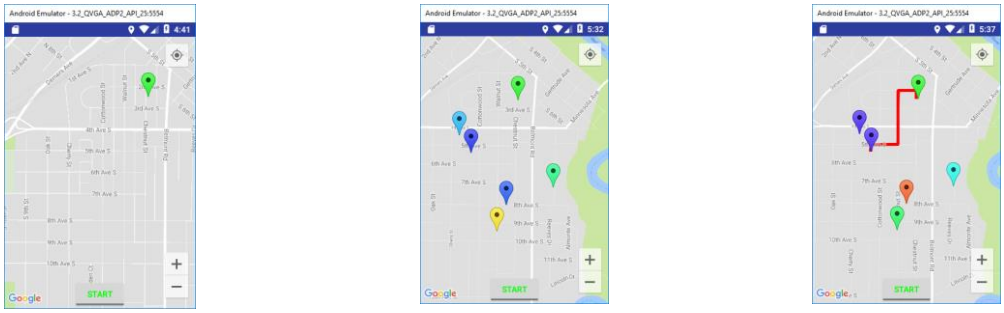
Figure 4: (a) The Proposed App Shown in the Android Launcher, (b) the Control Panel, and (c) the Extended Controls.

Figure 5 shows an example of an application of this service, where the user is marked in Figure 5.a, Figure 5.b shows two more participants joining the service, and a direction between the user and the nearest participant is drawn in Figure 5.c. Another example is given in Figure 6, which is similar to the one in Figure 5, but with five participants.



(a) (b) (c)

Figure 5: (a) User Being Marked, (b) Two More Participants Joining the Service, (c) Direction Between the User and the Nearest Participant.



(a) (b) (c)

Figure 6: Another Example: (a) User Being Marked, (b) Five More Participants Joining the Service, and (c) Direction Between the User and the Nearest Participant.

4 Implementation of the Proposed System

Design of the proposed system is given in the previous section and this section discusses the implementation part. Several programming languages are used in this implementation where Android Java is mainly used on the client-side for direction drawing, and PHP and Python are used on the server-side for accessing the geographical database and calling Google Maps Directions API (Application Programming Interface) (Google, n.d.) to find the shortest distance. Figure 7 shows a workflow of the implementation of the proposed system including (i) a user requesting a direction between him/her and the nearest participant, (ii) location data of the participants retrieved from the geographical database, (iii) the distances between the user and the participants calculated by using Google Maps Directions API, (iv) the data of the nearest participant sent to the user, and (v) the shortest direction drawn by using Google Directions API. Brief discussion of the implementation is given in this section. Further references for the related implementation can be found from the articles by the authors (Hu, 2018; Hu, Kaabouch, Yang, & Wang, 2014; Hu, Kaabouch, & Yang, 2014).

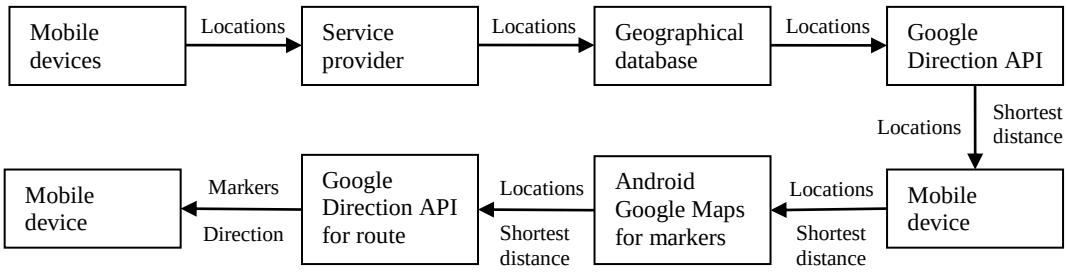


Figure 7: Workflow of the Proposed System Implementation.

4.1 Android Server Connection

There are a couple of ways allowing Android devices to connect to the servers. Three of the classes for the connections are given as follows:

- `URLConnection`, which represents a communications link between the application and a URL,
- `HttpClient`, which is an interface for an HTTP client and was deprecated in android 5.1 and is removed from the Android SDK (Software Development Kit) in Android 6.0, and
- `HttpURLConnection`, which is the preferred method to connect to a remote server through HTTP.

This research connects a mobile client to the server via the `HttpURLConnection` class. The connection allows the client to send its location data to the server and receive the information in a JSON format the server offers. It is also used to connect to the Google Maps Directions server to request a route between the user and the nearest participant. Program 1 shows a code snippet used to connect to the server and receive data from the server.

```
private ArrayList<LatLng> downloadUrl( LatLng user_location ) {
  HttpURLConnection urlConnection = null;
  URL url = new URL( "ADDRESS_TO_SERVICE_SERVER_WITH_DEVICE_LOCATION" );
  // Creating an HTTP connection to communicate with the URL
  urlConnection = (HttpURLConnection) url.openConnection( );
  // Connecting to the URL
  urlConnection.connect( );
  // Reading data from the URL
  while( ( line = br.readLine( ) ) != null ) {
    String[ ] latlong =  line.split( "," );
    double latitude  = Double.parseDouble( latlong[0] );
    double longitude = Double.parseDouble( latlong[1] );
    LatLng loc       = new LatLng( latitude, longitude );
    data.add( loc );
}
```

Program 1: Code Snippet for Server Connection.

JSON stands for *JavaScript Object Notation*. It, much like XML, is syntax for storing and exchanging text information. This application parses JSON files returned from Google Maps Directions via web services and extracts map data from it. Program 2 shows a code snippet used to connect to the Google Maps Directions service for drawing a direction between the user and the nearest participant. A URL for web services is built to request and retrieve the Google Maps Directions data in a JSON format. An example is given as follows:

```
https://maps.googleapis.com/maps/api/directions/json
    ?origin=37.422006,-122.084095
    &destination=37.431098,-122.234912&sensor=false
```

```
private String downloadJSON( LatLng user_location, LatLng other_user_location ) {
  HttpURLConnection urlConnection = null;
  URL url = new URL( "https://maps.googleapis.com/maps/api/directions/json?origin=" +
    user_location.latitude         + "," + user_location.longitude + "&destination=" +
    other_user_location.latitude   + "," + other_user_location.longitude +
    "&key=GOOGLE_API_KEY" );
  // Creating an HTTP connection to communicate with the URL
  urlConnection = (HttpURLConnection) url.openConnection( );
  // Connecting to the URL
  urlConnection.connect( );
  // Reading data from the URL
  while ( ( line = br.readLine( ) ) != null ) { sb.append( line ); }
```

Program 2: Code Snippet of Using Web Service for Google Maps Directions.

4.2 The Client-Side Code

Program 3 shows a client-side code snippet for the following two major functions:

- Receive data from the server.
- Use the data to draw the direction between the user and the nearest participant and mark all participants.

The `doInBackground` method is used to request and retrieve the Google Maps Directions data in a JSON format by using web services. The `onPostExecute` method is used to parse the JSON string returned from the `doInBackground` method.

```
private class ChangeLoc extends AsyncTask <Location, Void, String> {
  @Override
  protected String doInBackground( Location... params ) {
    mLatitude  = params[0].getLatitude( );
    mLongitude = params[0].getLongitude( );
    final LatLng point = new LatLng( mLatitude, mLongitude );
    runOnUiThread( new Runnable( ) {
      @Override
      public void run( ) {
        mGoogleMap.moveCamera(CameraUpdateFactory.newLatLng( point ) );
        mGoogleMap.animateCamera(CameraUpdateFactory.zoomTo( 12 ) );
      }
    } );
```

```
    ArrayList<LatLng> data = new ArrayList( );
    // Fetching the information of other users from web service
    data = downloadUrl( point );
    data.add( new LatLng( mLatitude, mLongitude ) );

    // Drawing the marker, if destination location is not set
    final ArrayList<LatLng> finalData = data;

    runOnUiThread( new Runnable( ) {
      @Override
      public void run( ) { drawMarker( data ); }
    });
    LatLng other_user_point = new LatLng( data.get(0).latitude, data.get(0).longitude );
    String results = "";
    // Fetching the route from web service (Google Maps Directions)
    results = downloadJSON( point, other_user_point );
    return results;
  }
  // Executing in UI thread, after the execution of doInBackground( )
  @Override
  protected void onPostExecute( String result ) {
    super.onPostExecute( result );
    ParserTask parserTask = new ParserTask( );
    // Invoking the thread for parsing the JSON data.
    parserTask.execute( result );
  }  // End of onPostExecute
}
```

Program 3: Client-Side Code Snippet.

4.3 The Server-Side Code

The server-side code includes the following two functions:

- Retrieve the participants' locations from the geographical database by using PHP.
- Find the shortest distance by using Python shown in Program 4.

Python and PHP are used because Python is convenient in text and array processing and PHP is commended for web processing. The shortest distance is found by finding the distance between the user and each participant and comparing the distances.

```
users_locs = sys.argv[1].split(':');  # Receiving other users' locations
## Iterating for user locations
for distan in users_locs:
  url = 'https://maps.googleapis.com/maps/api/directions/json?origin='
    + origin + '&destination=' + distan + '&key=' + api_key
  ## Receiving the JSON object from Google Maps Directions
  f = urllib.urlopen( url )
  string = f.read( )
  ## Parsing the JSON object
  j = J.loads( string )
  ## Receiving the distance
  dist = int( j["routes"][0]['legs'][0]['distance']['value'] )
  Distances.append( dist )
  ## Returning the user locations in an ascending order
  for i in sorted( Distances ):
    print users_locs[ Distances.index( i ) ]
```

Program 4: Server-Side Code Snippet.

5 Summary

Mobile applications, especially location-based services, have already made a great impact on our daily lives. It could be used for many purposes such as location-based social networks, network routing, or human behavior study and recognition, to name a few. This research will have broader and important impacts on location-based research and services, as well as computer science education at the University of North Dakota. Curriculum development and research initiatives will benefit from this research. This project involves a variety of topics and could be used to enhance curriculum development and for class projects. For example, the Data Science degree recently approved by the University of North Dakota will benefit from this project because mobile data is intensively used in this research. Research initiatives like the PhD program in scientific computing from the Computer Science Department will use this paper to enhance their research. Readers may apply the knowledge and technologies learned from this article to the following suggested research or applications:

- Finding the interesting nearby places such as ethnic restaurants and movie theaters,
- Providing various recommendations such as concerts and museums including driving/walking routes,
- Using social networks to connect people on the move within a short distance,
- Location-based promotions and coupons such as sales and discounts, and
- Indoor positioning and navigation used to help users have a better visiting experience.

References

Android. (n.d.a). *Android*. Retrieved February 18, 2018, from https://www.android.com/

Android. (n.d.b). *Android Studio: The official IDE for Android*. Retrieved March 03, 2018, from https://developer.android.com/studio/index.html

GeoNames. (n.d.). *GeoNames*. Retrieved March 10, 2018, from http://www.geonames.org/

Google. (n.d.). *Google Maps Directions API*. Retrieved February 16, 2018, from https://developers.google.com/maps/documentation/directions/

Hu, W.-C. (2018). *Data Engineering and Management*. Retrieved January 24, 2018, from http://udncemcs01.und.edu/~wen.chen.hu/course/515/

Hu, W.-C., Kaabouch, N., & Yang, H.-J (2014). Fundamental technologies and methodologies for mobile/handheld app development. In Mehdi Khosrow-Pour, editor, *Encyclopedia of Information Science and Technology*, ISBN13: 9-781-466-65888-2, ISBN10: 1-466-65888-6, IGI Global.

Hu, W.-C., Kaabouch, N., Yang, H.-J., & Wang, X. (2014, April 25-26). Essential Android technologies and Google Maps APIs for location-based services. In *Proceedings of the Midwest Instruction and Computing Symposium (MICS 2014)*, Verona, Wisconsin, USA.

IDC. (2018). *Smartphone OS*. Retrieved March 02, 2018, from https://www.idc.com/promo/smartphone-market-share/os

Malani, G. (2017, January). *Location-Based Services Market*. Allied Market Research, Portland, Oregon.
MySQL. (n.d.a). *MySQL*. Retrieved March 05, 2018, from https://www.mysql.com/
MySQL. (n.d.b). *MySQL Workbench*. Retrieved February 16, 2018, from https://www.mysql.com/products/workbench/
Technavio. (2017, May). *Global Location-Based Services (LBS) Market 2017-2021*. Technavio, Toronto, Ontario.