LiveTracking

By

Hasanul Sanjid

Id: 2015-1-50-021

Md. Yasir Arafat Sun

Id: 2015-1-50-003

Supervised by

Dr. Mohammad Arifuzzaman

Assistant Professor

Department of Electronics & Communications Engineering

East West University

A project submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Information & Communications Engineering

Department of Electronics & Communications Engineering

East WestUniversity

Dhaka-1212, Bangladesh

April 2019

Declaration

We, Hasanul Sanjid, and MD. Yasir Arafat Sun, hereby, declare that the work presented in this project is the outcome of the investigation perform by us under the supervision of Dr. Mohammad Arifuzzaman, Assistant Professor, Department of Electronics & Communications Engineering, East West University. We also declare that no part of this project has been or is being submitted elsewhere for the award of any degree or diploma.

Countersigned

……

(Dr. Mohammad Arifuzzaman)

Supervisor

Signature of Student

……..

(Hasanul Sanjid)

Id: 2015-1-50-021

..…….

(MD. Yasir Arafat Sun)

Id: 2015-1-50-003

Letter of Acceptance

This project entitled "**LiveTracking**" submitted by Hasanul Sanjid (ID: 2015-1-50-021), Md. Yasir Arafat Sun (ID: 2015-1-50-003) to the Department of Electronics & Communications Engineering, East West University, Dhaka, Bangladesh is accepted by the department in partial fulfillment of requirements for the Award of the Degree of Bachelor of Science in Information and Communication Engineering on Spring, 2019.

Supervisor

_____

Dr. Mohammad Arifuzzaman
Assistant Professor
Dept. of Electronics & Communications Engineering
East West University, Dhaka, Bangladesh.

Chairperson

_____

Dr. Mohammed Moseeur Rahman
Professor & Chairperson
Dept. of Electronics & Communications Engineering
East West University, Dhaka, Bangladesh.

Table of Contents

List of Figures

Abstract

Offline tracking is the way one can be tracked using other information than their online communications data. These offline Tracking is used to discover the area and to record the situation of the individual or gadget at customary interims. The location of a device (tablet, phone) can be easily and accurately identified if it is connected to, for example, to a cell tower. Sometimes, this tracking is technologically impossible to evade. In our App, we use google location API that send the longitude and latitude in a SMS to track the user device location. Thus, we don't need to call the user to track his device or him. By sending a SMS we can track the location of the device or the user. This app can track the location of the user or a device in every Five minutes. It doesn't require any active internet connection to track the location. When user connect the device with internet the tracking data uploaded into the server. In this app user can see his current location in the map and can see his previous location in the history. In this app a user can add as many trusted numbers as he /she wants. When the user is in danger, they can send the alert message to the trusted number by pressing the power button twice consecutively.

Acknowledgement

As it is true for everyone, we have also arrived at this point of achieving a goal in our life through various interactions with and help from other people. we would not like to make effortto find best words to express our thankfulness other than simply listing those people who have contributed to this project itself in an essential way. This work has been carried out in the Department of Electronics & Communications Engineering at East West University, Bangladesh.

First of all, we would like to express our deepest gratitude to the almighty Allah for his blessings on us. Next, our special thanks go to our supervisor Dr. Mohammad Arifuzzaman, Assistant Professor, East West University who gave us this opportunity, initiated us into the field of "Android Application & IOT", and without whom this work would not have been possible and their encouragements, visionaries and thoughtful comments and suggestions and unforgettable support. we would like to thank all Friends who give excellent collaboration during performance evaluation studies for overall support and helpful suggestions in solving tricky technical problems. Last but not the least, I would like to thank Our parents for their unending support, encouragement and prayers.

There are numerous other people too who have shown me their constant support and friendship in various ways, directly or indirectly related to our academic life. We will remember them in our heart and hope to find a more appropriate place to acknowledge them in the future.

MD. Yasir Arafat Sun

Hasanul Sanjid

April, 2019.

Chapter 1

Introduction

Technology becomes essential in our daily life. Technologies and new inventions are changing the world and the daily life of the people. The world currently floats on the sea of technology. Albert Einstein said “It has become appallingly obvious that our technology has exceeded our humanity”. We realize that without technology we can’t live right now. To make a better future we have to use this as a power that can solve all our valid problems and dilemmas. Now a day to track the location of a person or an object we use Global Positioning System (GPS). GPS can pinpoint longitude, latitude, ground speed, and course direction of the target **[1].** To measure this device has to be connecting with the internet where many people in our country don’t know how to connect their mobile device with the internet. And Mobile internet is expensive in our country there comes the part of offline live tracking. As raising the criminal activity in our country offline live tracking become essential for many people and companies to know the exact location of their employees.

The aim of this project is to help people to send their exact location to their relatives if they face any accident or any crime. And to help the law enforcement agencies to capture the criminal and help the victim. It will help Employers to track their employee’s location at the office time. As this project doesn’t need any active data connection it will help people if they face any mishap. With the help of this law enforcement agencies can respond earlier than before.

1.1 Motivation

Current population of Bangladesh is 164.7 million and the rate of crime is increasing here day by day. Sometimes our law enforcement agencies couldn't find the exact place where the crime occurred. In some cases, employees of some companies say that they were attacked on their way to work. In that case employer can check the employee's location of that specific location and time whether the employee was really there or not. Employees can send their location to their sir or manager if they face any kind of problem or criminal activity. We all about Bkash service or their transaction app. In the eve of EID or other festivals we hear the news that bkash agent gets robed from here and there. By using this app bkash agent can send their current location to their trusted contacts by message. The GPS tracker we use it is quite costly further more to track the movement of an entity the device has to be connect with the internet and that is also costly. Our project aim is to give the tracking service at low-cost so that everyone could use it if they want.

1.2 Objectives

The main aim of this project is to help the manager or head of the marketing of a company to find the sales officer. The manager can also see the Whole day place record of the employee to make sure whether the employee is in the right track or not. If the employees face any problem, they can send their location to the manager or other respected person. It also helps the law enforcement agencies to investigate any crime scene if any of the employee relate with the crime. It is also helpful for women if they face any problem, they can give their exact location to their trusted contacts.

1.3 Outline of the Report

In chapter 2, we researched about Tracking Apps and review some technologies used to implement this project; Chapter 3 analyzes the whole architecture of the system; In chapter 4, the database implementation and interfaces are presented and in chapter 5, conclusion and future of the project are mentioned.

Chapter 2

Background study

In this chapter, existing Apps related to offline tracking and technologies used to implement the project have been discussed in details.

2.1 Existing Location Tracking Apps

GPS My Tracks Offline **[1]** This application can record the tracks of your trek dependent on the organize point on GPS framework. This application can work disconnected with the goal that you don't have to depend on the web association constantly. It works in the wood, in the estate, in the mining region, adrift, and all over the place. You can record place, pace, speed, and the course that you have been through. This application is additionally lightweight, it devours not very many of your telephone memory and power.

All-In-One Offline Maps **[2]** In this App Your genuine area and bearing are plainly shown on the guide, which can be pivoted to coordinate your genuine introduction (relies upon gadget abilities) You can include different things the guide, for example, waypoints, symbols, courses, territories and tracks. You can without much of a stretch oversee them utilizing the amazing SD-Card Landmarks Explorer.

GPS Location Info, SMS Coordinates, Compass + **[3]** This application is a fantastic item included in Google Maps which will give your companions a chance to see precisely where you are, continuously. Exceptionally exact telephone Locator will assist you with using Google Maps in your telephone and see a flag indicating your companion's area in the guide too simply choosing the paired document for Google Maps.

GPS Reader **[4]** GPS Reader is a free application for perusing and putting away area data given by the Global Positioning System (GPS). It enables you to acquire your present area and store it

for future reference. Put away areas can be contrasted with get separation and bearing data and sent out to KML records for bringing into mapping applications, for example, Google Maps.

GPS Coordinates **[5]** GPS Coordinates application to android to get, offer, spare and pursuit map directions of your present area. GPS Coordinates Converter can change over any location to scope and longitude, convert scope longitude to a location, and discover facilitates. Duplicate and offer your present GPS arrange with loved ones utilizing GPS area in latlong configuration, address or both. You can likewise utilize the GPS organizes discoverer to share an alternate location or GPS facilitates.

2.2 Technological Background

To implement the project, some open source tools have been used such as XAMPP, Notepad++, Apache as web server. Android Studio as platform, Java as coding language, for service we use Google Api, MySQL is used as local database server and Firebase as main server

2.2.1Java

Java is a broadly useful programming language that is class-based, object-situated, and explicitly intended to have a couple of usage conditions as could be expected under the circumstances. It is planned to give application engineers "a chance to write once, run anyplace" (WORA), implying that aggregated Java code can keep running on all stages that help Java without the requirement for recompilation. Java applications are commonly aggregated to "bytecode" that can keep running on any Java virtual machine (JVM) paying little respect to the basic PC design. The linguistic structure of Java is like C and C++, however it has less low-level offices than both of them. Starting at 2018, Java was by GitHub a standout amongst the most famous programming dialects being used, especially for customer server web applications, with a detailed 9 million engineers.

2.2.2 Android Studio

Android Studio is the authority incorporated improvement condition (IDE) for Google's working framework, based on JetBrains' IntelliJ IDEA programming and planned specifically for Android advancement. It is accessible for download on Windows, macOS and Linux based working frameworks. It is a trade for the Eclipse Android Development Tools (ADT) as the essential IDE for local Android application development. Android Studio was declared on May 16, 2013 at the Google I/O meeting. It was in early access see arrange beginning from variant 0.1 in May 2013, at that point entered beta stage beginning from adaptation 0.8 which was discharged in June 2014. The principal stable form was discharged in December 2014, beginning from variant 1.0. The current stable adaptation is 3.3, which was discharged in January 2019.

2.2.3 Google API

Google APIs is a lot of use programming interfaces (APIs) created by Google which permit correspondence with Google Services and their combination to different administrations. Instances of these incorporate Search, Gmail, Translate or Google Maps. Outsider applications can utilize these APIs to exploit or expand the usefulness of the current services. The APIs give usefulness like examination, AI as an administration (the Prediction API) or access to client information (when consent to peruse the information is given). Another essential model is an inserted Google map on a site, which can be accomplished utilizing the Static maps API, Places API or Google Earth API.

.

2.2.4 Firebase

Firebase developed from Envolve, an earlier startup established by James Tamplin and Andrew Lee in 2011. Envolve gave engineers an API that empowers the combination of online talk usefulness into their sites. Designers were utilizing Envolve to synchronize application information, for example, amusement state continuously over their clients. Firebase's first item was the Firebase Realtime Database, an API that synchronizes application information crosswise over iOS, Android, and Web gadgets, and stores it on Firebase's cloud. The item helps programming engineers in structure ongoing, communitarian applications. In 2014, Firebase

propelled two items. Firebase Hosting and Firebase Authentication. This situated the organization as a versatile backend as an administration. In October 2014, Firebase was obtained by Google. In October 2015, Google obtained Divshot to blend it with the Firebase group.

2.2.5 Editor (Notepad++)

Notepad++ is a content manager and source code proofreader for use with Microsoft Windows. It underpins selected altering, which permits working with numerous open records in a solitary window. The task's name originates from the C increase administrator. At first Notepad++ is circulated as free programming, facilitated by SourceForge.net, from where it has been downloaded more than 28 million times and twice won the Source Forge Community Choice Award for Best Developer Tool. The undertaking was facilitated on Tux Family (fr) from 2010 to 2015; since 2015 Notepad++ has been facilitated on GitHub. Notepad++ utilizes the Scintilla proofreader part.

Chapter 3

System Analysis and Design

In this chapter, we analyzed the architecture of the whole project. System analysis is a necessary process of defining the architecture, component& data of a system to satisfy specified requirements. By examining a system component parts and their interaction is a method of studying design method of a system.

3.1 Process Requirement

The following process requirements are identified for system:

- After download the app from the play store user have to register with the phone number & a password before user log in the app
- A valid login is required for every registered user. After registration a user can login or access the app and use it.
- After a valid login user can check or edit the information & set the amount of trusted number in the App.
- Following previous steps user have to give the permission of the location & background access to the App. So that App can trace the location.
- The App can trace the location of the user 5 minutes consecutively and upload the data in a server.
- Only an admin can get access to the server to check in tracking information and the activity of several user.

3.2 Process details

3.2.1 Google Location API

Google APIs is a lot of utilization programming interfaces (APIs) created by Google which permit correspondence with Google Services and their combination to different administrations. Instances of these incorporate Search, Gmail, Translate or Google Maps. Outsider applications can utilize these APIs to exploit or expand the usefulness of the current administrations. The Geolocation API restores an area and precision sweep dependent on data about cell towers and WIFI hubs that the versatile customer can recognize.

3.2.2 Get the last known area

Utilizing the Google Play administrations area APIs, our application can demand the last known area of the client's gadget. Much of the time, you are keen on the client's present area, which is typically identical to the last known area of the gadget. In particular, utilize the fused location provider to recover the gadget's last known area. The combined area supplier is one of the area APIs in Google Play administrations. It deals with the basic area innovation and gives a straightforward API so you can determine necessities at an abnormal state, similar to high exactness or low power. It additionally enhances the gadget's utilization of battery control.

3.2.3 Set up Google Play services

To get to the combined area supplier, our application's improvement venture must incorporate Google Play administrations. Download and introduce the Google Play administrations part by means of the SDK Manager and added the library to our project.

3.2.4 Specify app permissions

Applications that utilization area administrations must demand area consents. Android offers two area consents: ACCESS_COARSE_LOCATION and ACCESS_FINE_LOCATION. In our App we use ACCESS_COARSE_LOCATION.

Solicitation this consent with the uses-permission authorization component in our application show, as the accompanying code bit appears:

```
<manifestxmlns:android="http://schemas.android.com/apk/res/android"
   package="com.google.android.gms.location.sample.basiclocationsample">

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

Figure 3.1: Specify app permissions code

3.2.5 Create location services client

In our action's onCreate() strategy, make a case of the Fused Location Provider Client as the accompanying code piece appears.

```
privateFusedLocationProviderClientfusedLocationClient;
// ..
@Override
protectedvoidonCreate(Bundle savedInstanceState){
   // ...
   fusedLocationClient =LocationServices.getFusedLocationProviderClient(this);
}
```

Figure 3.2: Create location services client code

3.2.6 Get the last known location

When we have made the Location Services customer, we begin to get the last known area of a client's gadget. At the point when our application is associated with these we begin to utilize the melded area supplier's getLastLocation() method to retrieve the device location. The exactness of the area returned by this call is dictated by the consent setting we put in our application show, as portrayed in the Specify App Permissions section of this document. To demand the last known area, call the `getLastLocation()` method. The accompanying code bit represents the solicitation and a basic treatment of the reaction:

```
fusedLocationClient.getLastLocation()
        .addOnSuccessListener(this,newOnSuccessListener<Location>(){
            @Override
            publicvoidonSuccess(Location location){
                // Got last known location. In some rare situations this can be
null.
                if(location !=null){
                    // Logic to handle location object
                }
            }
        });
```

Figure 3.3: Get the last known location code

The getLastLocation () technique restores a Task that we use to get a Location object with the scope and longitude directions of a geographic area. The area article might be invalid in the accompanying circumstances:

- Location is off in the gadget settings. The outcome could be invalid regardless of whether the last area was recently recovered on the grounds that crippling area likewise clears the store.
- The gadget never recorded its area, which could be the situation of another gadget or a gadget that has been reestablished to manufacturing plant settings.
- Google Play benefits on the gadget have restarted, and there is no dynamic Fused Location Provider customer that has mentioned area after the administrations restarted. To stay away from this circumstance one can, make another customer and solicitation area refreshes himself.

Chapter 4

Implementation

4.1-ER -Diagram

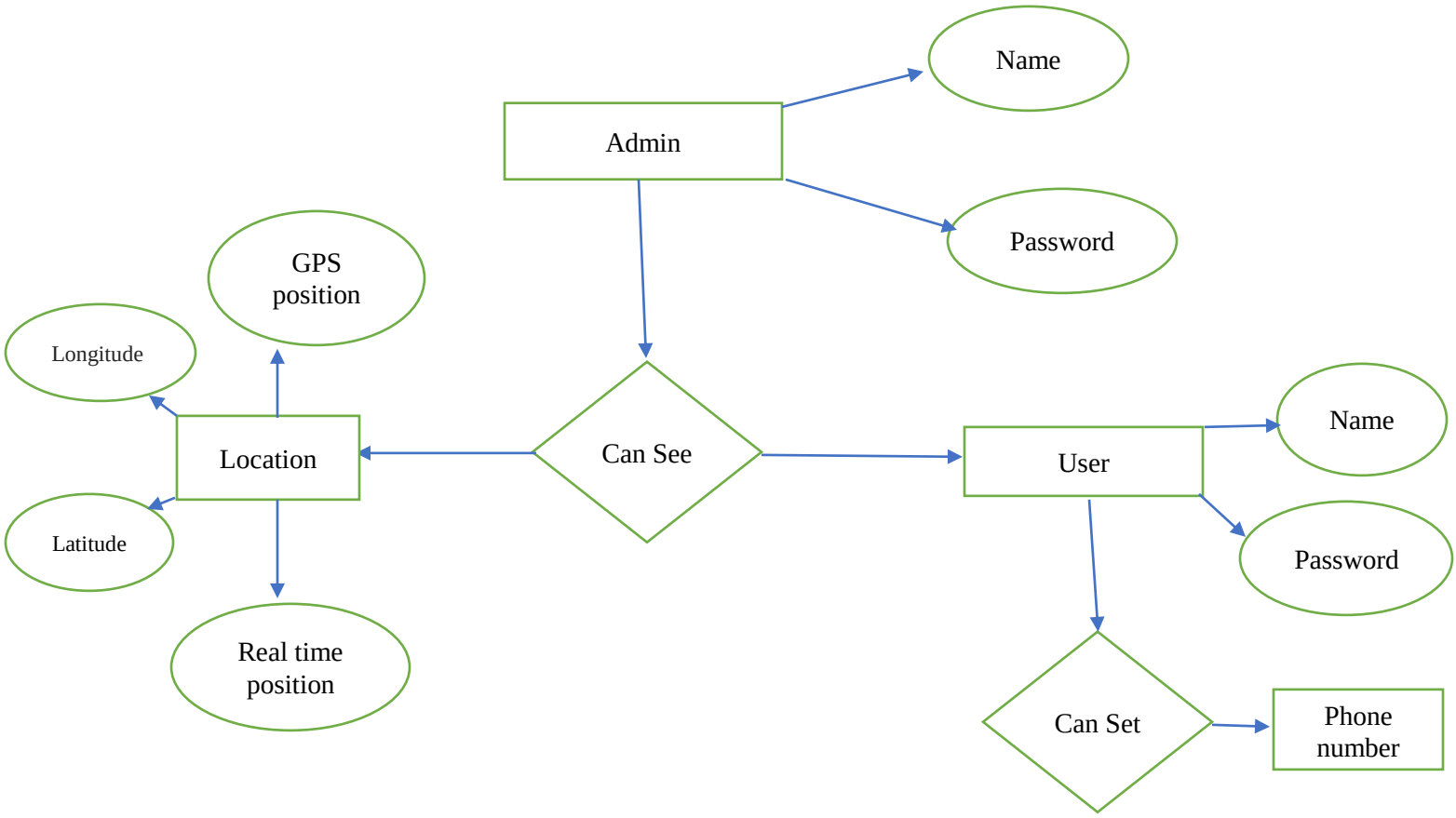

Figure: 4.1 ER- diagram of Database

4.2 Database Implementation

In the wake of getting the prerequisite of a legitimate plan and auxiliary structure of our database, we can move to the execution organize. All in all, actualizing our basic structure includes characterizing the different articles and upholding the requirements on the information connections.

4.2.1 Database:

Database stored all the information of the user along with the amount of the user and their details.

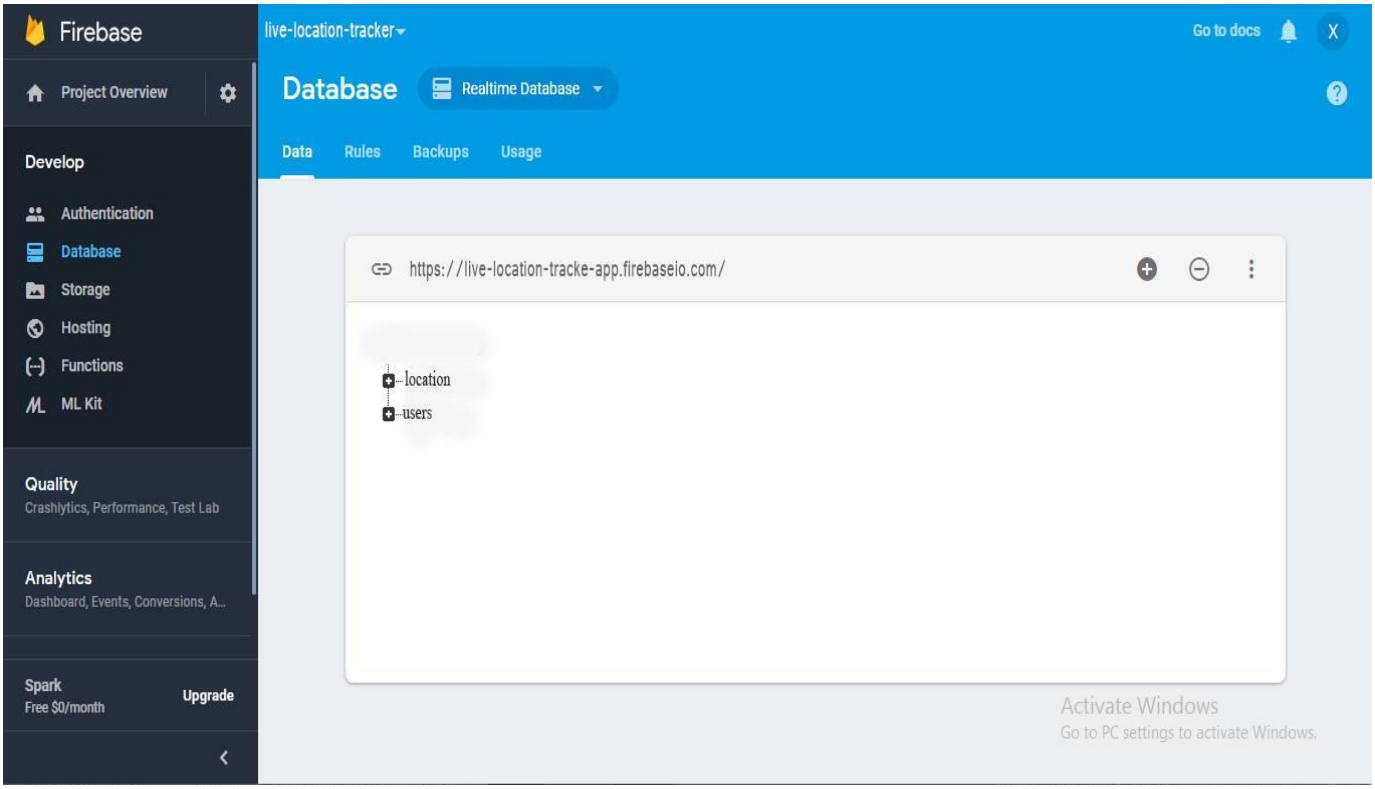

Figure 4.1: Main Databases

4.2.2 Database of user:

Database of users store all listed user's data such as name, address, mobile number, location. Figure 4.2 shows the database of users.

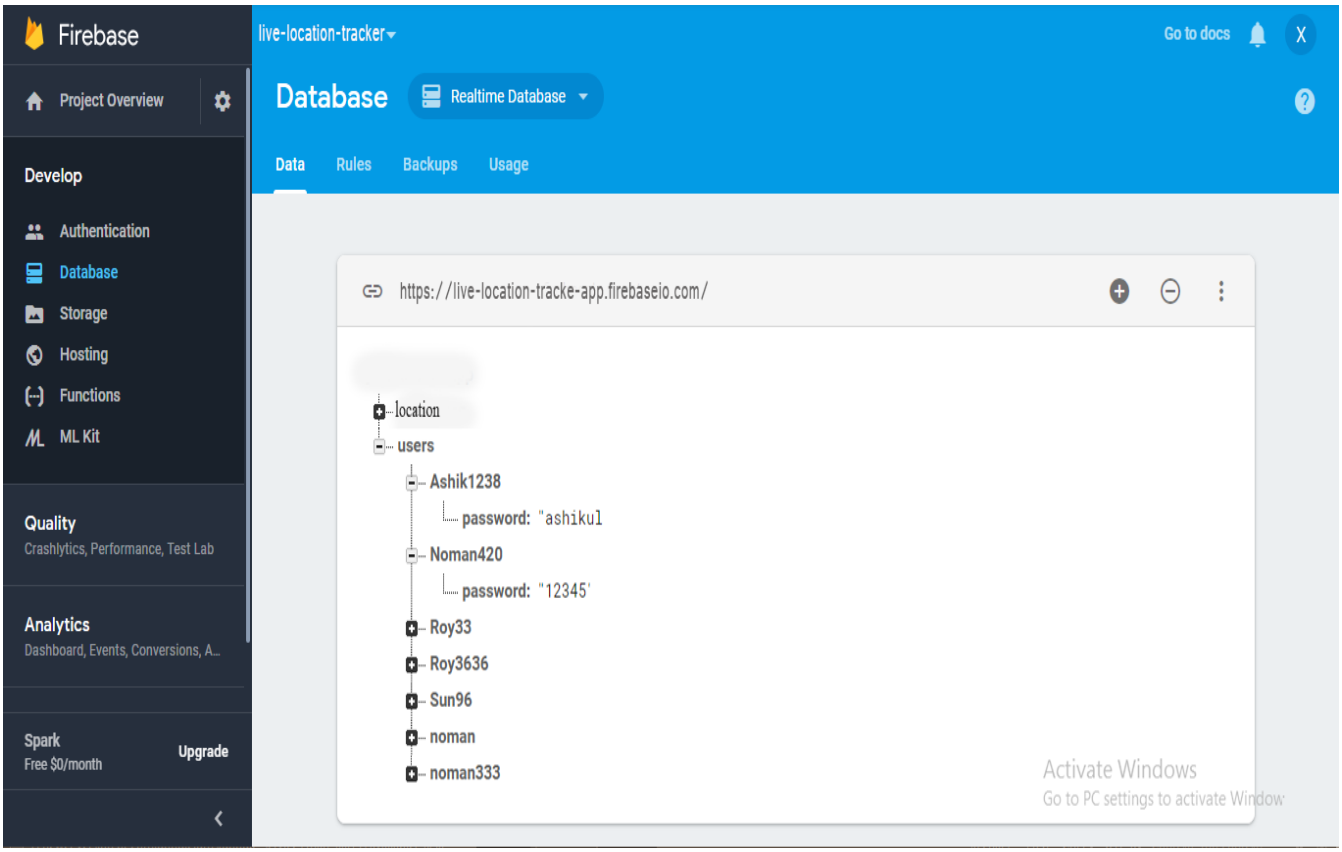

Figure 4.2: Databases of users

4.3 Interface

A basic part of frameworks configuration is to make the UI to the new framework. Info and yield configuration center around the substance of that interface – the particular fields that ought to be incorporated into screens and reports that are seen by the clients. When the substance is resolved, the configuration for human-PC association (HCI) is resolved. The (UI) is the manner in which the framework converses with the clients, utilizing screens/structures, reports, and blunder messages. Amid interface structure designers recognize strategies for every framework movement and the required contributions for those exercises. These required data sources become screens or structures. Client association is basic amid these structure exercises.

4.3.1 Home page:

This is the home page for LiveTracking application. In the top menu there is start service and stop service by pressing those user give the permission of the background activity to the app. In the middle there is Service on or off by pressing this user can activate the location tracker and map, history etc.4.3 shows the Home page of the system.

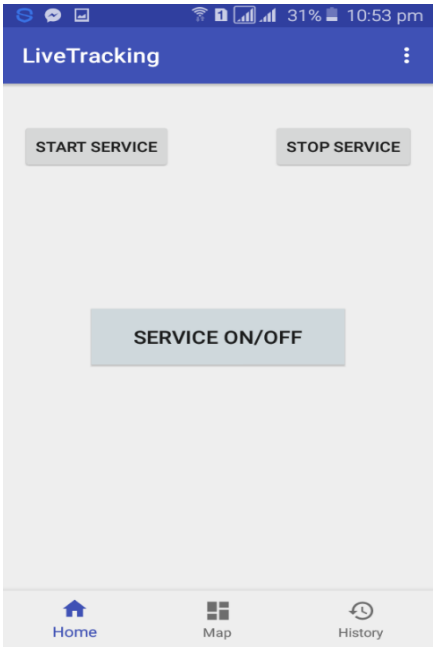

Figure 4.3: Home page.

4.3.2 Registration page:

This is the account registration page for LiveTracking application. Here user can register an account to access the service of the application. User must need to give requirement information for registration. User fill up the form with require all info which is complete the registration. If a person fills the form and submits, then he/ she become a user of the site. Figure 4.4 shows the Registration page of the system.

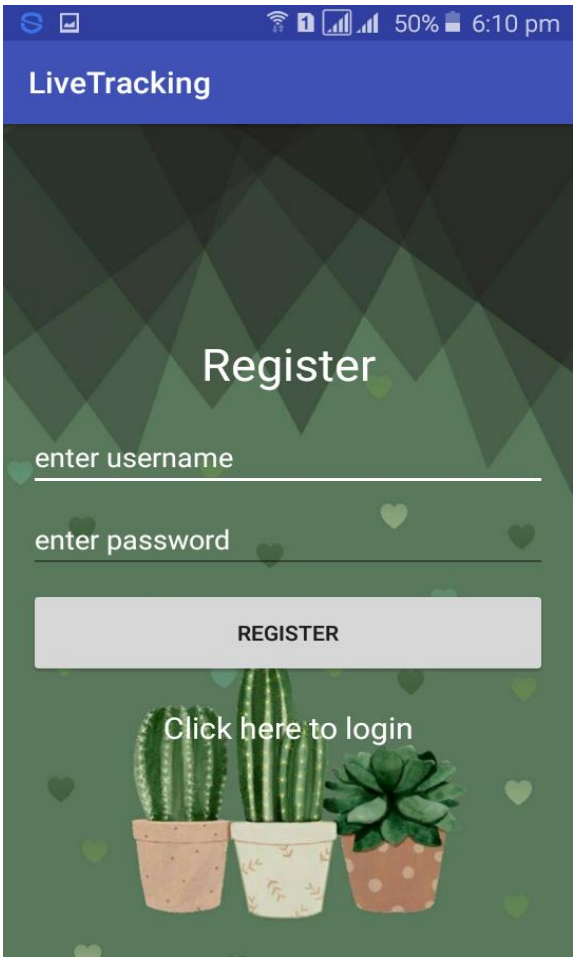

Figure 4.4: User Registration page.

4.3.3 Account Log in page:

This is the account login page for LiveTracking application. Here user can login to access his/her account. User must need to register for login. User fills up the form with registered mobile number and valid password which is used at registration. Figure 4.5 shows the login page of the system.

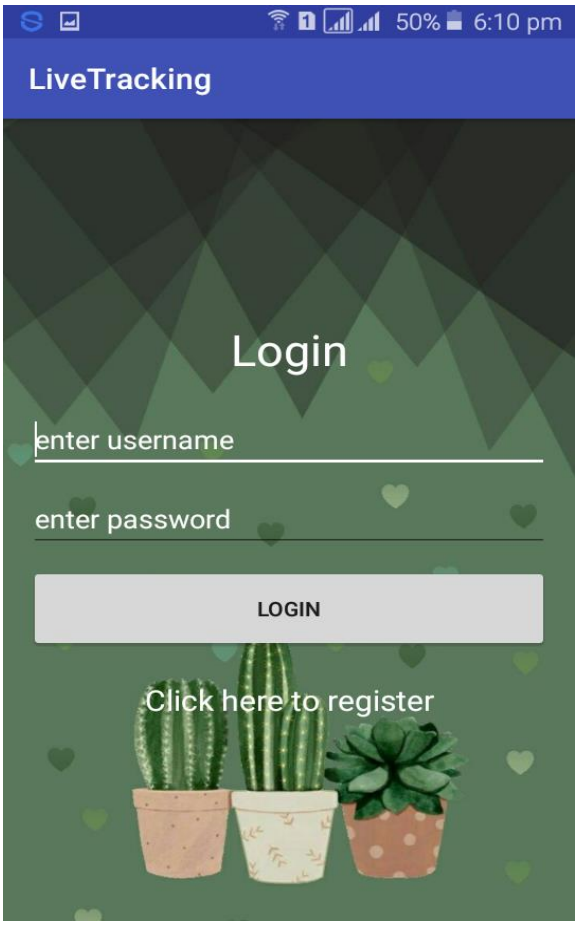

Figure 4.5: User Account log in page.

4.3.4 Add Number page:

This is the account login page for LiveTracking application. Here user can add the number of their trusted person from whom they can expect help. The amount of number that a user can add is unlimited. Figure 4.6 shows the Add number page of the system.

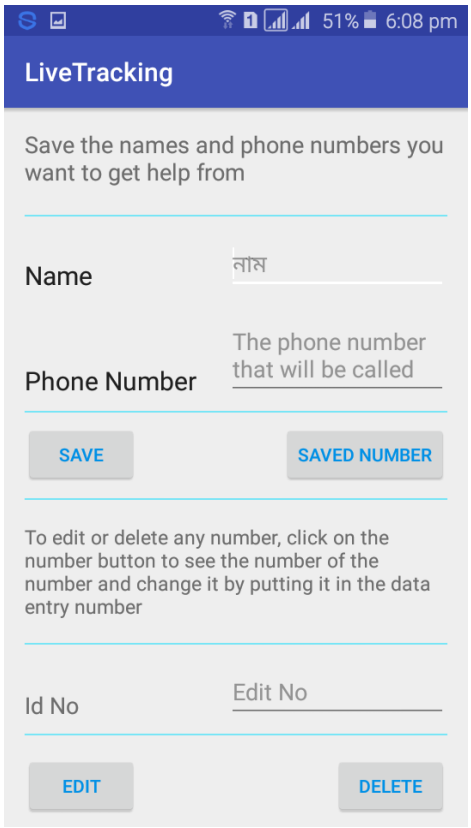

Figure 4.6: Add number page

4.3.5 User Location page:

This is the Location page for LiveTracking application. Here user can see location that consecutively updated in 5 minutes. Here location is shown in Longitude and latitude. Figure 4.7 shows the user location page of the system.

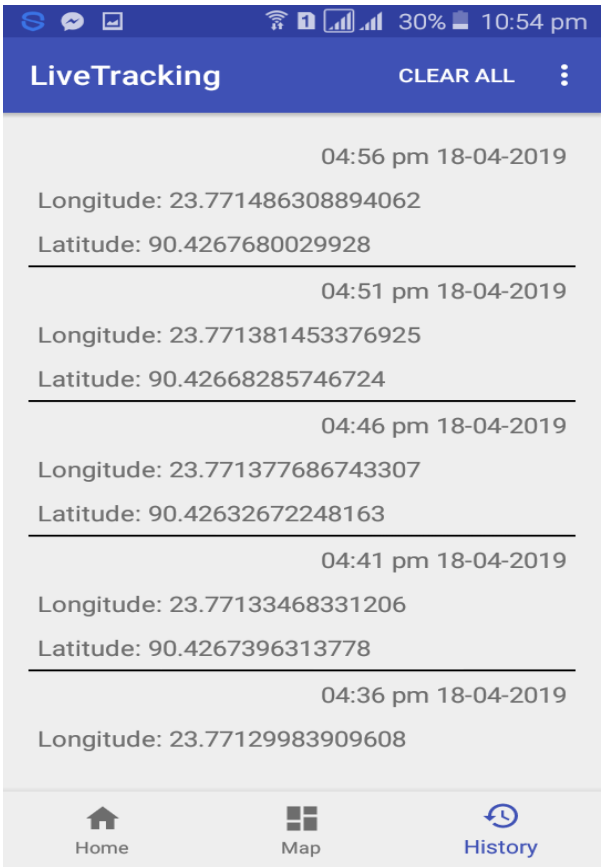

Figure 4.7: user location page

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In the end, we have to consider that no work or project can be 100% perfect. To make a perfect project, we had to give our 99%. In this report, we have developed an offline location tracking system to track the position of the person. To make flexible system improvements on network problems, offline location tracking provides good results. Google Location API based tracking using network provider had been used for location tracking but without using internet. The "Offline Live Tracking" project is successfully designed and developed to fulfill the necessary requirements, and the system is very much user friendly. We tried our best to make this project less expensive and more user friendly. The present project has been developed to meet the aspirations indicated in the modern age. It is very helpful to locate and trace our most dear person or employees of an organization if they are any danger. It will also be helpful for the law-enforcement agencies to track criminal activity. When the user in Danger he/she can simply send alert message to their chosen number. A database server stores all the activity of the user if they need it later.

5.2 Future Work

In this project we tried to make this Application more user friendly and less expensive than usual tracking Application. So, after executing the debug and upgrade plan we are going to focus on the future of this application. After releasing the next update version, we are going to overview user's review as we stored those on the database. Then we going to evaluate those and find solution if there are any major problems or user demands. After this we could release the developer version. Therefore, we try to do some future work, those are given below:

- In future addition we are thinking to release the IOS version of our Application. So that every smart device user can get our service.
- The GPS tracker that we use for vehicle tracking is expensive in our country. In future we are thinking to implement our "LiveTracking" Application into small devices to trace the vehicles.

References

1. https://whatis.techtarget.com/definition/GPS-tracking
2. https://www.eetimes.com/document.asp?doc_id=1278363
3. https://www.gpswox.com/en/blog/useful-information/the-advantages-of-gps-tracking-system
4. https://developers.google.com/maps/documentation/geolocation/intro
5. https://en.wikipedia.org/wiki/Java_(programming_language)
6. https://play.google.com/store/apps/details?id=gpstracksoffline.ttsberita.www&hl=en
7. https://play.google.com/store/apps/details?id=net.psyberia.offlinemaps&hl=en
8. https://play.google.com/store/apps/details?id=com.gpsdragon.gpslocation&hl=en
9. https://play.google.com/store/apps/details?id=com.latitudelongitude.gpscoordinates&hl=en
10. https://developer.android.com/studio/intro
11. https://en.wikipedia.org/wiki/Google_APIs
12. https://firebase.google.com/

Appendix

Codes

```
public class GPSTracker extends Service implements LocationListener {

    // Get Class Name
private static String TAG = GPSTracker.class.getName();

    private final Context mContext;

    // flag for GPS Status
booleanisGPSEnabled = false;

    // flag for network status
booleanisNetworkEnabled = false;

    // flag for GPS Tracking is enabled
booleanisGPSTrackingEnabled = false;

    Location location;
    public double latitude;
    public double longitude;

    // How many Geocoder should return our GPSTracker
```

```
intgeocoderMaxResults = 1;

    // The minimum distance to change updates in meters
    private static final long MIN_DISTANCE_CHANGE_FOR_UPDATES = 10; //
10 meters

    // The minimum time between updates in milliseconds
    private static final long MIN_TIME_BW_UPDATES = 1000 * 60 * 1; // 1
minute

    // Declaring a Location Manager
    protected LocationManagerlocationManager;

    // Store LocationManager.GPS_PROVIDER or
LocationManager.NETWORK_PROVIDER information
    private String provider_info;

publicGPSTracker(Context context) {
this.mContext = context;
getLocation();
    }

    /**
     * Try to get my current location by GPS or Network Provider
     */
    @SuppressLint("MissingPermission")
public void getLocation() {
```

```
        try {
locationManager = (LocationManager)
mContext.getSystemService(LOCATION_SERVICE);

            //getting GPS status
isGPSEnabled =
locationManager.isProviderEnabled(LocationManager.GPS_PROVIDER);

            //getting network status
isNetworkEnabled =
locationManager.isProviderEnabled(LocationManager.NETWORK_PROVIDER);

            // Try to get location if you GPS Service is enabled
            if (isGPSEnabled) {
this.isGPSTrackingEnabled = true;

Log.d(TAG, "Application use GPS Service");

                /*
                 * This provider determines location using
                 * satellites. Depending on conditions, this provider may take a while to
return
                 * a location fix.
                 */

provider_info = LocationManager.GPS_PROVIDER;
```

```
        } else if (isNetworkEnabled) { // Try to get location if you Network Service
is enabled
this.isGPSTrackingEnabled = true;

Log.d(TAG, "Application use Network State to get GPS coordinates");

            /*
             * This provider determines location based on
             * availability of cell tower and WiFi access points. Results are retrieved
             * by means of a network lookup.
             */
provider_info = LocationManager.NETWORK_PROVIDER;

        }

        // Application can use GPS or Network Provider
if (!provider_info.isEmpty()) {
locationManager.requestLocationUpdates(
provider_info,
                MIN_TIME_BW_UPDATES,
                MIN_DISTANCE_CHANGE_FOR_UPDATES,
                this
            );

if (locationManager != null) {
                location = locationManager.getLastKnownLocation(provider_info);
```

```
updateGPSCoordinates();
                }
            }
        }
        catch (Exception e)
        {
            //e.printStackTrace();
Log.e(TAG, "Impossible to connect to LocationManager", e);
        }
    }

    /**
     * Update GPSTracker latitude and longitude
     */
public void updateGPSCoordinates() {
if (location != null) {
latitude = location.getLatitude();
longitude = location.getLongitude();
        }
    }

    /**
     * GPSTracker latitude getter and setter
     * @return latitude
     */
public double getLatitude() {
if (location != null) {
```

```
latitude = location.getLatitude();
        }

        return latitude;
    }

    /**
     * GPSTracker longitude getter and setter
     * @return
     */
public double getLongitude() {
if (location != null) {
longitude = location.getLongitude();
        }

        return longitude;
    }

    /**
     * GPSTrackerisGPSTrackingEnabled getter.
     * Check GPS/wifi is enabled
     */
publicbooleangetIsGPSTrackingEnabled() {

returnthis.isGPSTrackingEnabled;
    }
```

```
    /**
     * Stop using GPS listener
     * Calling this method will stop using GPS in your app
     */
public void stopUsingGPS() {
if (locationManager != null) {
locationManager.removeUpdates(GPSTracker.this);
        }
    }

    /**
     * Function to show settings alert dialog
     */
public void showSettingsAlert() {
AlertDialog.BuilderalertDialog = new AlertDialog.Builder(mContext);

        //Setting Dialog Title
alertDialog.setTitle(R.string.GPSAlertDialogTitle);

        //Setting Dialog Message
alertDialog.setMessage(R.string.GPSAlertDialogMessage);

        //On Pressing Setting button
alertDialog.setPositiveButton(R.string.action_settings, new
DialogInterface.OnClickListener() {

            @Override
```

```
public void onClick(DialogInterface dialog, int which)
            {
                Intent intent = new
Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
mContext.startActivity(intent);
            }
        });

        //On pressing cancel button
alertDialog.setNegativeButton(R.string.cancel, new
DialogInterface.OnClickListener() {

            @Override
public void onClick(DialogInterface dialog, int which)
            {
dialog.cancel();
            }
        });

alertDialog.show();
    }

    /**
     * Get list of address by latitude and longitude
     * @return null or List<Address>
     */
public List<Address>getGeocoderAddress(Context context) {
```

```
if (location != null) {

Geocodergeocoder = new Geocoder(context, Locale.ENGLISH);

        try {
            /**
             * Geocoder.getFromLocation - Returns an array of Addresses
             * that are known to describe the area immediately surrounding the given latitude and longitude.
             */
            List<Address> addresses = geocoder.getFromLocation(latitude, longitude, this.geocoderMaxResults);

            return addresses;
        } catch (IOException e) {
            //e.printStackTrace();
Log.e(TAG, "Impossible to connect to Geocoder", e);
        }
    }

    return null;
  }

  /**
   * Try to get AddressLine
   * @return null or addressLine
   */
```

```
public String getAddressLine(Context context) {
        List<Address> addresses = getGeocoderAddress(context);

if (addresses != null &&addresses.size() > 0) {
            Address address = addresses.get(0);
            String addressLine = address.getAddressLine(0);

            return addressLine;
        } else {
            return null;
        }
    }

    /**
     * Try to get Locality
     * @return null or locality
     */
public String getLocality(Context context) {
        List<Address> addresses = getGeocoderAddress(context);

if (addresses != null &&addresses.size() > 0) {
            Address address = addresses.get(0);
            String locality = address.getLocality();

            return locality;
        }
        else {
```

```
            return null;
        }
    }

    {
        List<Address> addresses = getGeocoderAddress(context);
if (addresses != null &&addresses.size() > 0) {
            Address address = addresses.get(0);
            String countryName = address.getCountryName();

            return countryName;
        } else {
            return null;
        }
    }

    @Override
public void onLocationChanged(Location location) {
    }

    @Override
public void onStatusChanged(String provider, int status, Bundle extras) {
    }

    @Override
public void onProviderEnabled(String provider) {
    }
```

```
   @Override
public void onProviderDisabled(String provider) {
   }

   @Override
publicIBinderonBind(Intent intent) {
       return null;
   }
}
```

```
public class IncomingSms extends BroadcastReceiver
{ PersonDatabasepsdb;
   @Override
public void onReceive(Context context, Intent intent)
   {

final Bundle bundle = intent.getExtras();
       try {
if (bundle != null)
           {
final Object[] pdusObj = (Object[]) bundle.get("pdus");
for (inti = 0; i<pdusObj .length; i++)
               {
```

```
SmsMessagecurrentMessage = SmsMessage.createFromPdu((byte[])
pdusObj[i]);
                String phoneNumber =
currentMessage.getDisplayOriginatingAddress();
                String senderNum = phoneNumber ;
                String message = currentMessage .getDisplayMessageBody();
                try
{ psdb= new PersonDatabase(context);
final Cursor res2 = psdb.getAllData();
                   while (res2.moveToNext()) {
                      final String num = res2.getString(2);
                      if (senderNum.equals(num)) {
if (message.equals("position")) {

MyServiceSms = new MyService();
GPSTrackergpsTracker = new GPSTracker(context);
                         String latitude = String.valueOf(gpsTracker.latitude);
                         String longitude = String.valueOf(gpsTracker.longitude);
Sms.recivedSms(num, latitude, longitude);
}elseToast.makeText(context, message, Toast.LENGTH_LONG).show();
                      } //else Toast.makeText(context, "not number matching",
Toast.LENGTH_LONG).show();
                   }
                }
catch(Exception e){}

             }
```

```
            }

        } catch (Exception e)
        {

        }
    }

}



public class PersonDatabase extends SQLiteOpenHelper {

    public static final String Database_NAME = "Member.db";
public static final String Table_NAME =  "Member_Table";
public static final String Col_1 =  "ID";
public static final String Col_2 =  "NAME";
public static final String Col_3 =  "SURNAME";

publicPersonDatabase(Context context) {
super(context, Database_NAME, null, 1);

    }

    @Override
public void onCreate(SQLiteDatabasedb) {
```

```
db.execSQL("create table " + Table_NAME + "(ID INTEGER PRIMARY KEY
AUTOINCREMENT, NAME TEXT, SURNAME TEXT)");
    }

    @Override
public void onUpgrade(SQLiteDatabasedb, intoldVersion, intnewVersion) {
db.execSQL("DROP TABLE IF EXIST"+Table_NAME);
onCreate(db);
    }

publicbooleaninsertdata(String name, String surname){
SQLiteDatabasedb = this.getWritableDatabase();
ContentValuescontentValues = new ContentValues();
contentValues.put(Col_2,name);
contentValues.put(Col_3,surname);
long result= db.insert(Table_NAME, null, contentValues);
        if (result == -1)

            return false;

        else
            return true;
    }

public Cursor getAllData(){

            }
```

```
public Boolean updateuserData(String id, String name, String surname){
SQLiteDatabasedb = this.getWritableDatabase();
ContentValuescontentValues = new ContentValues();
contentValues.put(Col_1,id);
contentValues.put(Col_2,name);
contentValues.put(Col_3,surname);
db.update(Table_NAME , contentValues , "ID = ?",new String[] { id });
        return true;

    }
public Integer deleteData(String id) {
SQLiteDatabasedb = this.getWritableDatabase();
returndb.delete(Table_NAME, "ID = ?",new String[]{id} );



    }
}

public class History extends Fragment implements
AdapterView.OnItemClickListener {
ArrayList<LocationClass>arrayList;
LocationAdapterlocationAdapter;
ListViewlistView;

SQLiteDatabasedb;
DbHelper_gpshelperDb;
```

```
    Cursor cursor;


public History() {
        // Required empty public constructor
    }


    @Override
public View onCreateView(LayoutInflaterinflater, ViewGroup container,
                         Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        View rootView = inflater.inflate(R.layout.fragment_history, container, false);

listView = (ListView) rootView.findViewById(R.id.listView_id);
helperDb=new DbHelper_gps(getContext());
listView.setOnItemClickListener(this);
initialize();

        return rootView;
    }

public void initialize(){
arrayList = new ArrayList<>();

db=helperDb.getReadableDatabase();
helperDb.create_table(db);
```

```
        cursor=helperDb.getAllInfo(db);

if(cursor.moveToFirst()){
do{
                int id;
doublelongi,lati;
                long time;

id=cursor.getInt(0);
longi=cursor.getDouble(1);
lati=cursor.getDouble(2);
time=cursor.getLong(3);

arrayList.add(new LocationClass(id,longi,lati,time));
}while(cursor.moveToNext());
        }

db.close();

Collections.sort(arrayList, new Comparator<LocationClass>() {
publicint compare(LocationClass m1, LocationClass m2) {
returnLong.compare(m1.getTime(), m2.getTime());
            }
        });

    }
```

```
public class Home extends Fragment implements View.OnClickListener{
   Button btn_service;

SharedPreferencessp;
SharedPreferences.Editor editor;

   Button bstrat,bend;

public Home() {
      // Required empty public constructor
   }


   @Override

{
if(v==btn_service){

         // INITIALIZE INTENT
         Intent intent=new Intent(getContext(),Alarm_broadcast.class);
         //PASS CONTEXT,YOUR PRIVATE REQUEST CODE,INTENT
OBJECT AND FLAG
PendingIntent pi= PendingIntent.getBroadcast(getContext(),0,intent,0);
AlarmManageralarmManager= (AlarmManager)
getContext().getSystemService(ALARM_SERVICE);
```

```
        //will trigger every5 minnutes

if(sp.getString("service","").equals("on")){
            //if on then turn off
btn_service.setBackgroundColor(parseColor("#cfd8dc"));

alarmManager.cancel(pi);

editor=sp.edit();
editor.putString("service","off");
editor.apply();

Toast.makeText(getContext(), "Service stopped",
Toast.LENGTH_SHORT).show();

            ((Main)getActivity()).call_by_home();

        }
        else {
            //if off then turn on
btn_service.setBackgroundColor(parseColor("#00c853"));

alarmManager.setInexactRepeating(AlarmManager.RTC_WAKEUP,System.curre
ntTimeMillis()+2000,2*60000,pi);

editor=sp.edit();
editor.putString("service","on");
```

```
editor.apply();

Toast.makeText(getContext(), "Service started",
Toast.LENGTH_SHORT).show();
          }
       }
   }
}
public class Map extends Fragment implements OnMapReadyCallback{
   private GoogleMapmMap;

       }

   @Override
public void onViewCreated(@NonNull View view, @Nullable Bundle
savedInstanceState) {
super.onViewCreated(view, savedInstanceState);
SupportMapFragmentmapFragment =
(SupportMapFragment)getChildFragmentManager().findFragmentById(R.id.map1
);
mapFragment.getMapAsync(this);
   }

   @Override
public void onMapReady(GoogleMapgoogleMap) {
mMap = googleMap;
```

```
LatLngsydney;

if(sp.getString("map_from","").equals("history")){

        Double temp_longi = Double.longBitsToDouble(sp.getLong("longi",
Double.doubleToLongBits(0)));
        Double temp_lati = Double.longBitsToDouble(sp.getLong("lati", 0));

sydney = new LatLng(temp_lati,temp_longi);
mMap.addMarker(new MarkerOptions().position(sydney).title("Your History"));
mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
CameraPositioncameraPosition = CameraPosition.builder()
                .target(sydney)
                .zoom(13)
                .bearing(90)
                .build();

        // Animate the change in camera view over 2 seconds

mMap.animateCamera(CameraUpdateFactory.newCameraPosition(cameraPosition
),
                2000, null);
      }
else if(sp.getString("map_from","").equals("direct")){
get_current_location();
      }
else if(sp.getString("map_from","").equals("home")){
```

```
get_last_location();
        }

    }

public void get_last_location(){
        //it is recent location

DbHelper_gpshelperDb=new DbHelper_gps(getContext());
SQLiteDatabasedb =helperDb.getReadableDatabase();
setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);
if(session.isLoggedIn()){
loadDashboard();
        }
register = (TextView)findViewById(R.id.register);
username = (EditText)findViewById(R.id.username);
password = (EditText)findViewById(R.id.password);
loginButton = (Button)findViewById(R.id.loginButton);

register.setOnClickListener(new View.OnClickListener() {
            @Override
public void onClick(View v) {
startActivity(new Intent(Login.this, Register.class));
            }
        });

loginButton.setOnClickListener(new View.OnClickListener() {
```

```
        @Override
public void onClick(View v) {
user = username.getText().toString();
pass = password.getText().toString();

if(user.equals("")){
username.setError("can't be blank");
            }
else if(pass.equals("")){
password.setError("can't be blank");
            }
else{
                String url = "https://yo-bro-chat-app.firebaseio.com/users.json";
finalProgressDialogpd = new ProgressDialog(Login.this);
pd.setMessage("Loading...");
pd.show();

StringRequest request = new StringRequest(Request.Method.GET, url, new
Response.Listener<String>(){
                    @Override
public void onResponse(String s) {
if(s.equals("null")){
Toast.makeText(Login.this, "user not found", Toast.LENGTH_LONG).show();
                        }
else{
                            try {
JSONObjectobj = new JSONObject(s);
```

```
if(!obj.has(user)){
Toast.makeText(Login.this, "user not found", Toast.LENGTH_LONG).show();
                        }
else if(obj.getJSONObject(user).getString("password").equals(pass)){
UserDetails.username = user;
UserDetails.password = pass;
startActivity(new Intent(Login.this, Main.class));
session.loginUser(user);

                        }
                        else {
Toast.makeText(Login.this, "incorrect password",
Toast.LENGTH_LONG).show();
                        }
                    } catch (JSONException e) {
e.printStackTrace();
                    }
                }
```