CSE3001 - SOFTWARE ENGINEERING

(J Component)

PROJECT REPORT

"SHARE THE FARE"

Submitted By

SHANTANU MAHALE 17BCE1161

SHIKHAR BHARDWAJ 17BCE1250

SHIVAM PANDEY 17BCE1129

Under the Guidance of

Dr S. Ganapathy

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

VIT – CHENNAI Campus, Chennai-127.

OCTOBER 2018

TABLE OF CONTENTS

ABSTRACT

This document is a project report of the Software Engineering Project – Share the Fare. In the introduction, the need of the app is mentioned. Why do we need an app with such functionalities, what are the problems with the existing systems, all of that is discussed in the introduction.

Further, all of the existing systems are elaborated and all the loopholes in the existing systems are highlighted. The drawbacks of the existing systems are also enlisted and the need for better systems with fulfilling functionalities is stated.

In the proposed systems section, all the requirements are listed and accordingly the working of the proposed system is discussed. Most of the problems, that occur while using the existing system that are being solved in the proposed system and the way all the things are managed are mentioned.

The proposed system section also consists of the details regarding communication phase, analysis phase, design phase, implementation phase as well as testing.

At last, the conclusion and the references are mentioned.

CHAPTER 1

INTRODUCTION

Generally, when people travel from one place to another via some public transport, they expect it to reach their destinations on time as well as it must have reasonable fare. Using a bus or local train can be cheap but at the same time one cannot tell that it will serve within a particular time frame.

For example, a train from Chennai to New Delhi won't take 8 hours even if you want that. Because the minimum time of travel may be 36 hours. It can't be any less.

So, as an alternative for local travelling, one prefers taxis and cab services over the public bus transport or the metro trains. But, if slightly longer distance is considered, the time to reach destination is a way much faster than the local buses, but the fare for that ride also becomes too high to bear for that distance.

Here, the very high prices of the taxis become an issue. What if the person wanting to travel by cabs and taxis to catch up with the speed, can afford to pay a fourth or a fifth of the price he/she is currently paying. To make that possible one can cab-pool his cab with other people.

But again, the problem is, how does one know if anyone from his place is willing to go to the same place at the same time. That's exactly what 'Share the Fare' does.

The app takes the details of every traveller and shows all the useful of that to every other traveller, so that, the next time ever if anyone uses the app, one can find fellow passengers who want to travel to the same place.

It is completely different from the share taxis that are operational at present. A typical share taxi is like a local bus (it goes everywhere dropping and picking everyone), plus it will charge more.

CHAPTER 2

EXISTING SYSTEMS

Existing system consist of private taxis/cabs or share taxis.

Let's assume that the distance of travel is 25 km. A private taxi on an average will charge Rs 20.0/km, considering all the 24 hours in a day where unexpected high fares like Rs 30-40/km during the night-time or seasonal rush are also taken into account.

That sums up to Rs 500.0 for the total ride (which is obviously too much if only one person has to travel). It becomes a problem not only for any regular user who wants the service daily or weekly, but also for the ones who travel occasionally.

Let's assume that people are ready to pay even half the price but the full price is too much for them. Even if they get a chance to cab-pool their ride with at least one other guy, both of them will be saving as good as half the fare.

And, if the existing system of private cabs is becoming just too expensive, people won't have any problem in switching to the alternative that we are providing.

Let's assume the case of a share taxi. A typical share taxi will also charge Rs 15.0/km on an average, taking into consideration, all odds. The average can lowest hit to Rs 12.0/km and no less.

So, the minimum and the maximum fares can be Rs 300.0 and Rs 375.0. Moreover, the share taxi is picking and dropping other people also. It can be considered as a small version of local buses. Thus, time becomes a major issue in this case.

If someone wants to catch a flight or a train, he/she is less likely to rely on share taxis and is more likely to opt for private taxis, that again are very costly to be bearable.

Due to such cases happening very often, there arises the need of something that enables the people to cab-pool their rides and hence

effectively save money and insure that they reach the destined location on time.

Cab-pooling can be, in all ways, a better option for the people. It saves time and money of the customers.

A point to be noted in case of share taxis is that, the share taxis will have a route. The taxi will be having one final destination and one final pickup point. It will be moving in that particular route only, but it may take many sub-routes in between. Thus, it will have many pickup and destination points.

Whereas, in cab-pooling, a private cab is booked and the fare is shared among all the passengers travelling. Here, there is only one pickup and one common destination. Thus, the cab won't stop anywhere in between and also would not take any unnecessary sub-routes. This is again another disadvantage of the existing system.

CHAPTER 3

PROPOSED SYSTEM

The basic idea is to cab-pool, so that the fare is shared and the money is saved. But the problem is how will a person be able to find another person going to the same destination and at the same time. It is practically impossible to ask every individual stranger about his destination and time of travel and to know whether that particular individual is willing to share the cab or not.

To counter this problem, an internet-based mobile application can be made where the individuals wanting to share their cab-rides can put up their travel details and they can view other individuals who are also willing to travel to the same destination and at the same time. This way, individuals can find peers for cab-pooling.

3.1 Communication Phase

Here are some functional requirements that our internet-based mobile application named 'Share the Fare' app is going to provide:

- **Registration:** Every user has to first register before logging in into the application. The user registration will be form – based activity that the user will need to feed in the basic information so that he/she can be uniquely identifiable. User will need to fill the following data:
    - **Name**
    - **Mobile Number**
    - **Email**
    - **Password**
- **Login:** Every user has to login before any further procedures. Login form will ask the user his/her registered email address and password to confirm authenticity.
    - **Email**
    - **Password**

- **Pickup:** After logging in, the login session will start, where all the details about the travelling will be collected from the user. One of them is the pickup location. The pickup location can be the current address of the person that can be found out via GPS of his/her handset or if one wants to chose another pickup location, he/she can do that also.
    - **Google Maps Activity:** This activity provides current location of the user. User can choose other destination also.
- **Destination:** This is similar to that of pickup. Another maps activity will enable to feed in the location and get that location as destination location.
- **Date & Time:** A date and time picker facility is provided where the date and time of travelling can be selected.
- **View List Option:** After having filled all the information, one can click here to view other people along with their contact details. Thus, he/she may get the people with whom the ride can be shared.
- **Logout:** It enables users to log out of the system after they are done with the application.

3.2 Analysis Phase

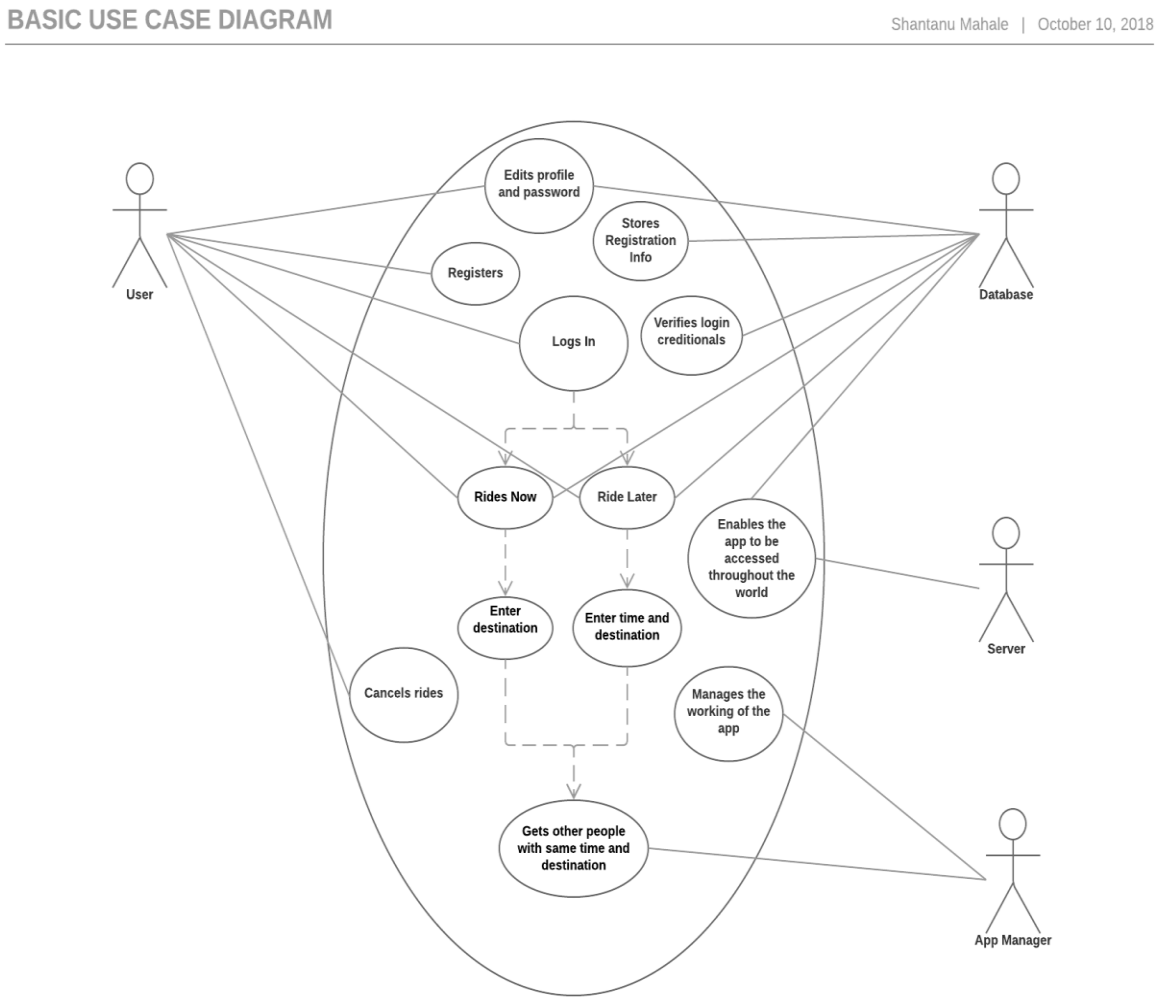
In the analysis phase, all the non-functional requirements are checked upon:

- Access Security
- Accessibility
- Availability
- Confidentiality
- Efficiency
- Integrity
- Reliability
- Safety
- Sustainability
- Flexibility
- Maintainability
- Scalability

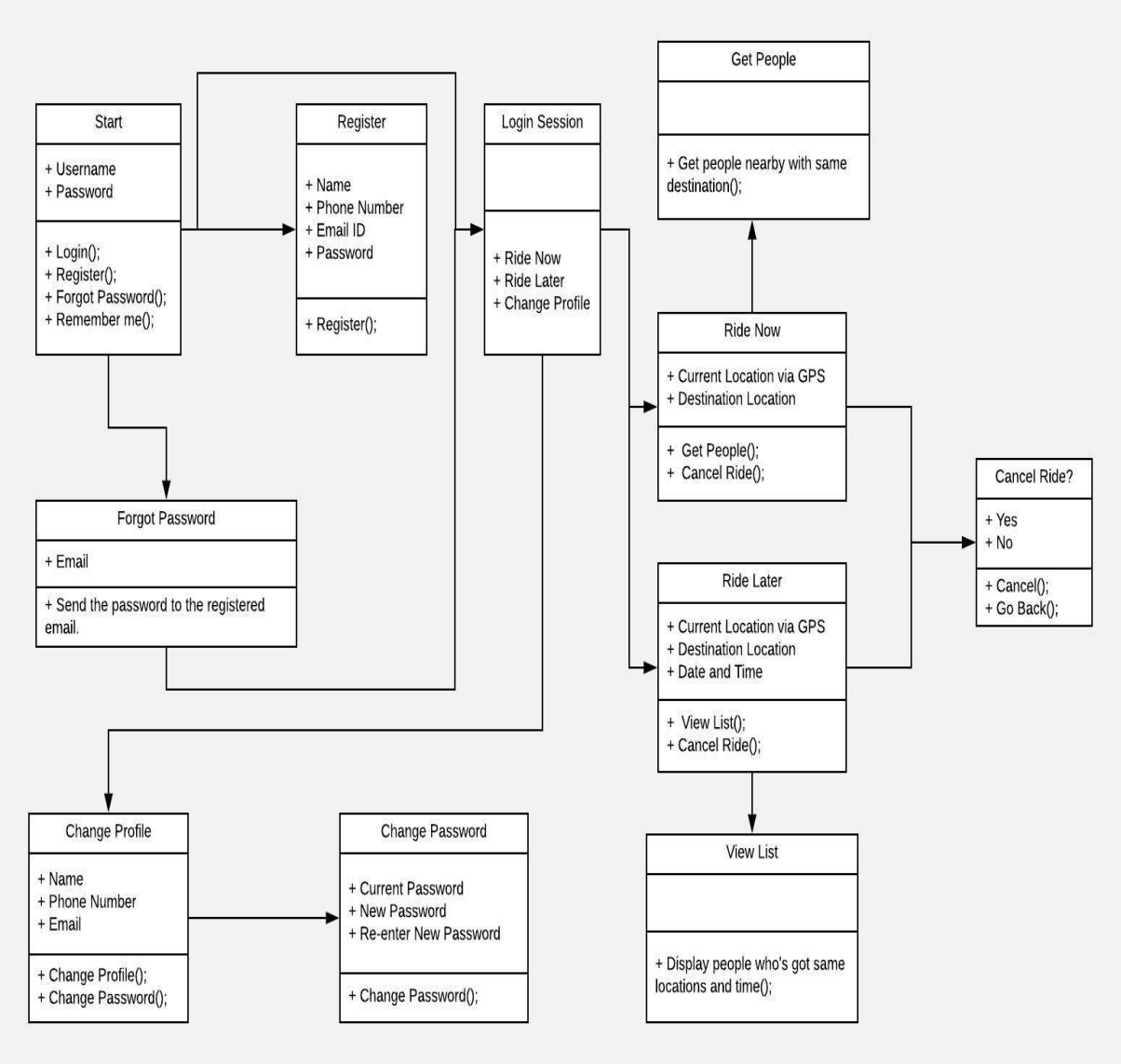
- Portability
- Reusability

3.3 Design Phase

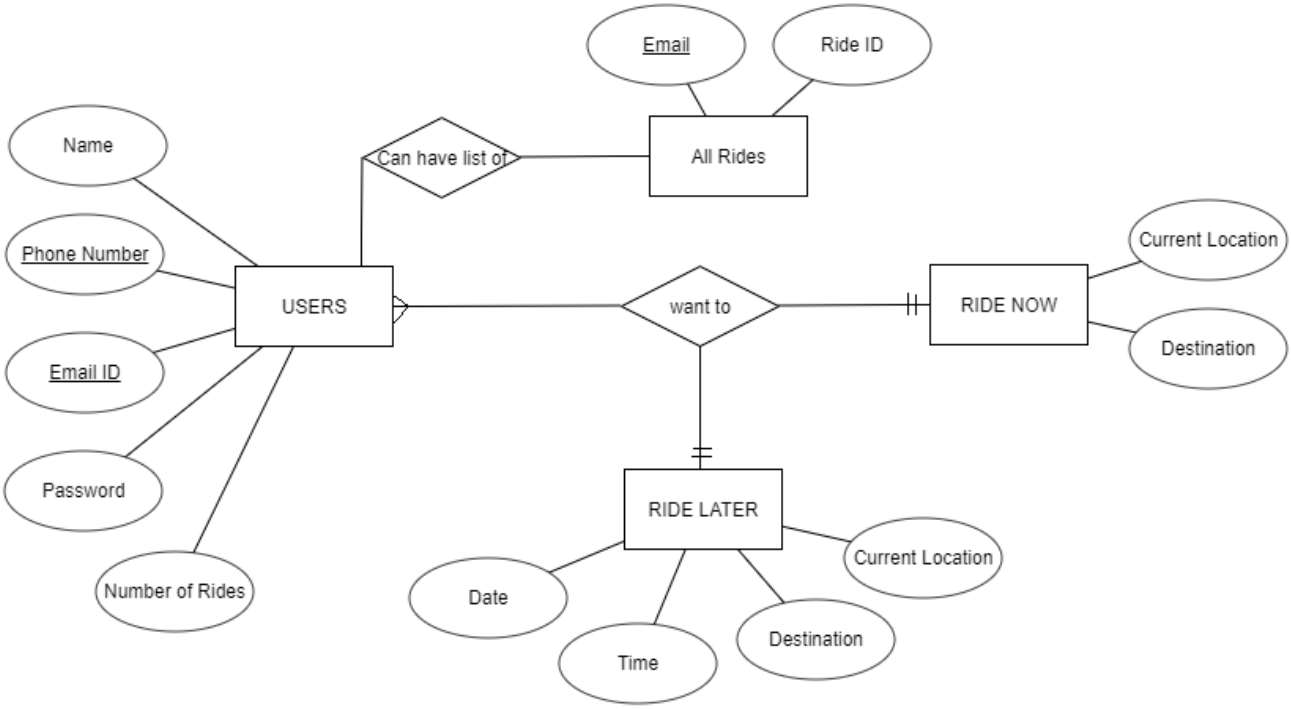
1. Use Case Diagram:



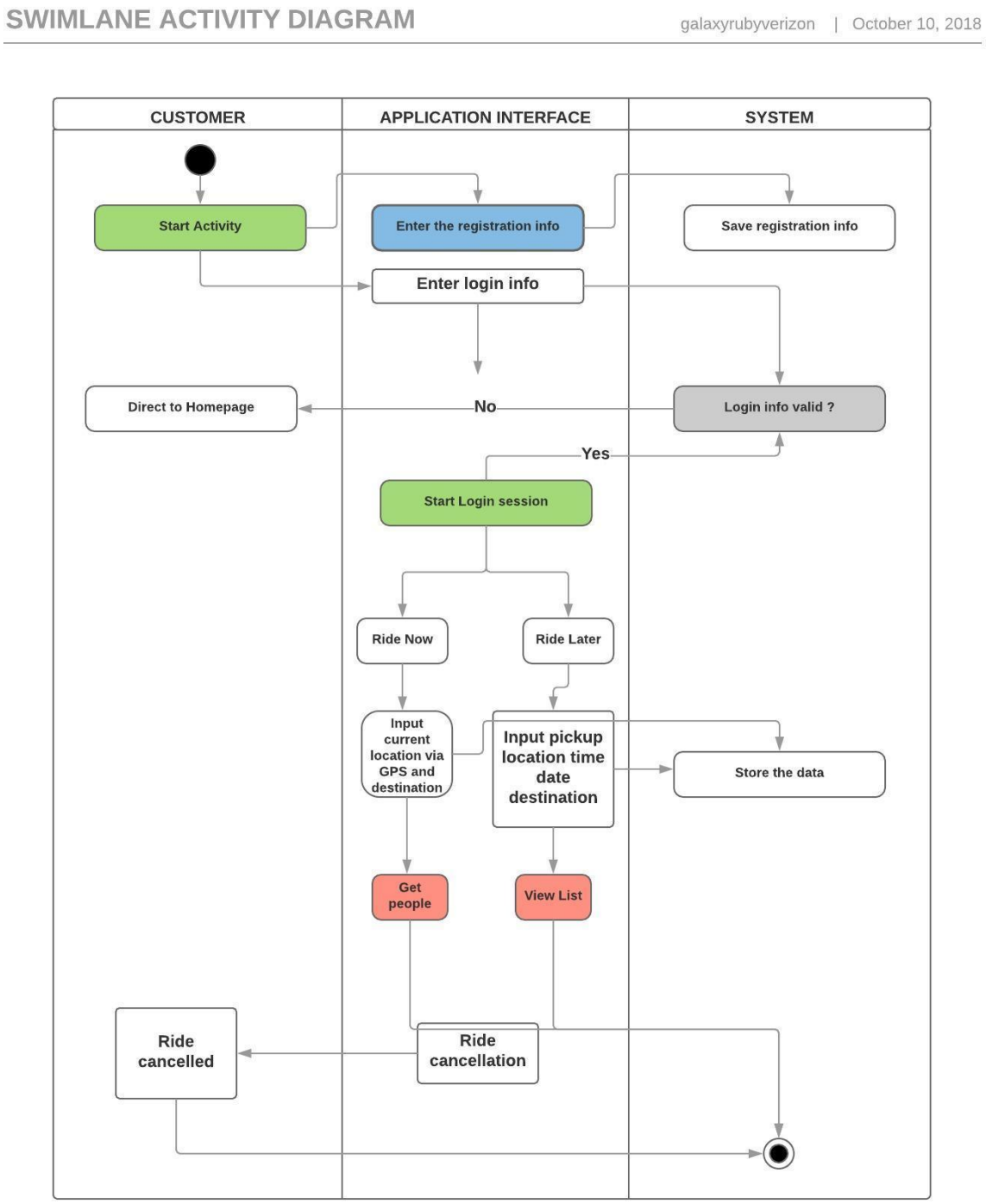
2. Class Diagram:



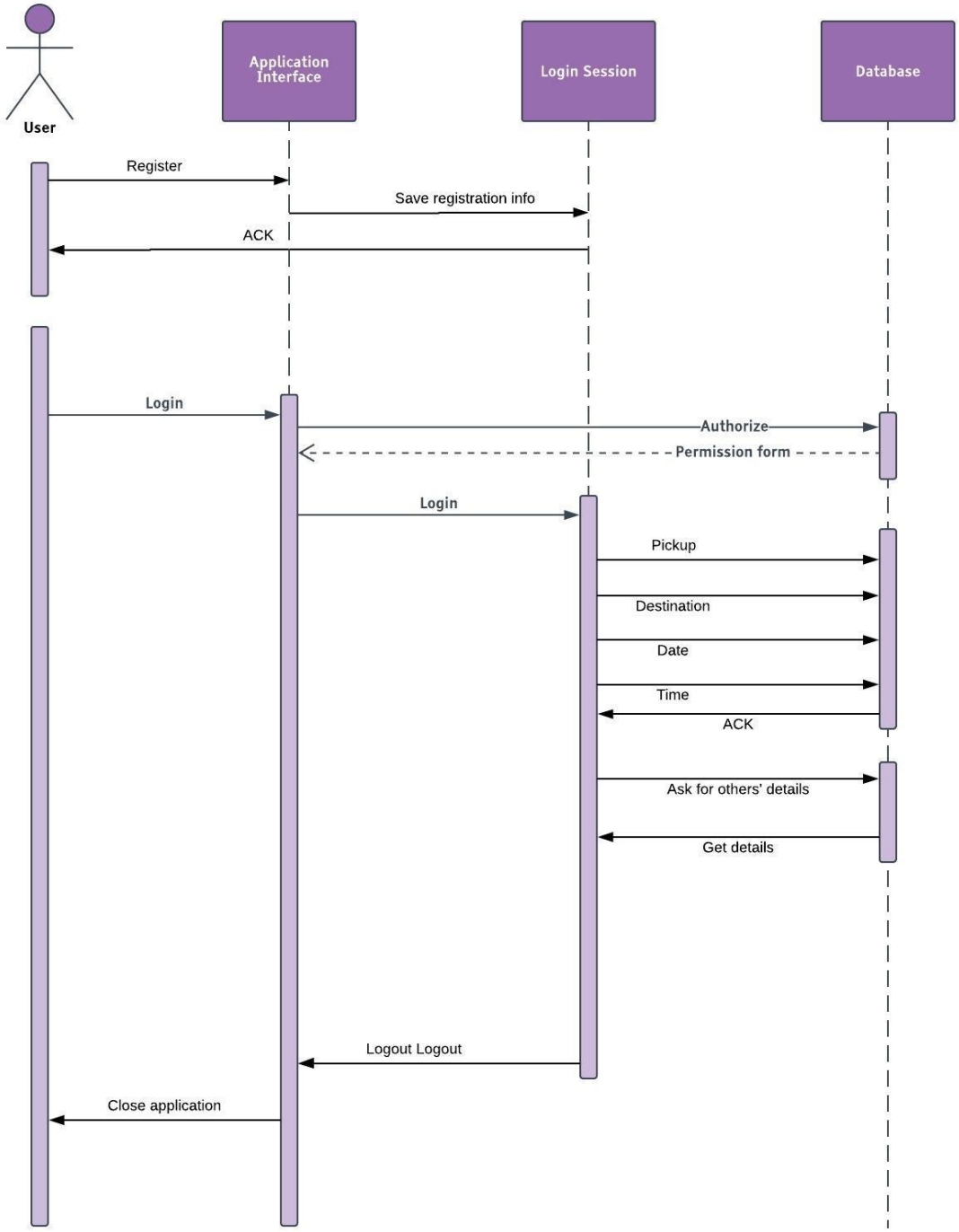
3. Entity – Relationship Diagram:



4. Activity Diagram:



5. Sequence Diagram



3.4 Implementation Phase

1. Home Page: Login

```java
package com.sharethefare.sharethefare;

import android.content.Context;
import android.content.Intent;
import android.os.Vibrator;
import android.support.design.widget.TextInputLayout;
import android.app.Activity;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class start extends AppCompatActivity {

    private Vibrator vib;
    DBH myDb;

    public static EditText email_login;
    EditText pass_login;

    Button login;
    Button register;

    private session session;

    public void init(){
        register.setOnClickListener((view) -> {
                Intent reg = new Intent( packageContext: start.this, registerr.class);
                startActivity(reg);
        });
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_start);

        session = new session( ctx: this);

        email_login = (EditText)findViewById(R.id.email_login);
        pass_login = (EditText)findViewById(R.id.pass_login);
        login = (Button)findViewById(R.id.login_button);
```

```java
        login = (Button)findViewById(R.id.login_button);
        register = (Button)findViewById(R.id.register);

        vib = (Vibrator) getSystemService(Context.VIBRATOR_SERVICE);

        myDb = new DBH( context: this);

        init();

        login.setOnClickListener((view) -> {
                Intent log = new Intent( packageContext: start.this,after_login.class);
                if(!validateEmail(email_login.getText().toString().trim())){
                    vib.vibrate( milliseconds: 120);
                    email_login.setError("Invalid Email");
                    email_login.requestFocus();
                }
                else {
                    if(!validatePassword(pass_login.getText().toString())) {
                        vib.vibrate( milliseconds: 120);
                        pass_login.setError("Invalid Password");
                        pass_login.requestFocus();
                    }
                    else{
                        boolean isInserted = myDb.check(
                                email_login.getText().toString(), pass_login.getText().toString() );
                        if(isInserted == true) {
                            session.setLoggedin(true);
                            startActivity(log);
                        }
                        else
                        {
                            Toast.makeText( context: start.this,  text: "Invalid Credentials!",
                                    Toast.LENGTH_LONG).show();

                        }
                    }
                }
        });

    }

    protected boolean validateEmail(String email) {
        String emailPattern = "^[_A-Za-z0-9-\\+]+(\\.[_A-Za-z0-9-]+)*@"
                + "[A-Za-z0-9-]+(\\.[A-Za-z0-9]+)*(\\.[A-Za-z]{2,})$";

        Pattern pattern = Pattern.compile(emailPattern);
        Matcher matcher = pattern.matcher(email);

        return matcher.matches();
    }

    protected boolean validatePassword(String password) {
        if(password!=null && password.length()>1) {
            return true;
        } else {
            return false;
        }
    }
}
```

2. Register

```java
package com.sharethefare.sharethefare;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.os.Vibrator;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class registerr extends AppCompatActivity{

    private Vibrator vib;
    DBH myDb;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);

        final EditText name = (EditText)findViewById(R.id.names);
        final EditText mobile = (EditText)findViewById(R.id.mobiles);
        final EditText email = (EditText)findViewById(R.id.emails);
        final EditText password = (EditText)findViewById(R.id.pwds);
        final Button register = (Button) findViewById(R.id.register);

        vib = (Vibrator) getSystemService(Context.VIBRATOR_SERVICE);

        myDb = new DBH( context: this);

        register.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View v) {
                if(!validateName(name.getText().toString().trim())){
                    vib.vibrate( milliseconds: 120);
                    name.setError("Invalid Name");
                    name.requestFocus();
                }
                else{
                    if(!validateMobile(mobile.getText().toString().trim())){
                        vib.vibrate( milliseconds: 120);
                        mobile.setError("Invalid Mobile Number");
                        mobile.requestFocus();
                    }
                    else{
                        if(!validateEmail(email.getText().toString())) {
                            vib.vibrate( milliseconds: 120);
                            email.setError("Invalid Email");
                            email.requestFocus();
                        }
                        else{
                            if (!validatePassword(password.getText().toString())) {
                                vib.vibrate( milliseconds: 120);
                                password.setError("Invalid Password");
                                password.requestFocus();
                            }
                            else{
                                boolean isInserted = myDb.insertData(name.getText().toString(),
                                        mobile.getText().toString(),
                                        email.getText().toString(), password.getText().toString() );
                                if(isInserted == true) {
                                    Toast.makeText( context: registerr.this,
                                            text: "You have registered successfully!",
                                            Toast.LENGTH_LONG).show();
                                    Intent inte = new Intent( packageContext: registerr.this, start.class);
```

```java
                                    startActivity(inte);
                                }
                                else
                                {
                                    Toast.makeText( context: registerr.this,
                                            text: "You have not registered successfully!",
                                            Toast.LENGTH_LONG).show();

                                }
                            }
                        }
                    }
                }
            }
        });


    }
    protected boolean validateName(String name) {
        if(name!=null && name.length()>1) {
            return true;
        } else {
            return false;
        }
    }

    protected boolean validateEmail(String email) {
        String emailPattern = "^[_A-Za-z0-9-\\+]+(\\.[_A-Za-z0-9-]+)*@"
                + "[A-Za-z0-9-]+(\\.[A-Za-z0-9]+)*(\\.[A-Za-z]{2,})$";

        Pattern pattern = Pattern.compile(emailPattern);
        Matcher matcher = pattern.matcher(email);

        return matcher.matches();
    }

    protected boolean validatePassword(String password) {
        if(password!=null && password.length()>1) {
            return true;
        } else {
            return false;
        }
    }
}
```

3. Session

```java
package com.sharethefare.sharethefare;

import android.content.Context;
import android.content.SharedPreferences;


public class session {
    SharedPreferences prefs;
    SharedPreferences.Editor editor;
    Context ctx;

    public session(Context ctx){
        this.ctx = ctx;
        prefs = ctx.getSharedPreferences( s: "share_the_fare", Context.MODE_PRIVATE);
        editor = prefs.edit();
    }

    public void setLoggedin(boolean logggedin){
        editor.putBoolean( s: "loggedInmode",logggedin);
        editor.commit();
    }

    public boolean loggedin() { return prefs.getBoolean( s: "loggedInmode",  b: false); }
}
```

4. After logging in

```java
package com.sharethefare.sharethefare;

import java.util.Calendar;

import android.app.DatePickerDialog;
import android.app.TimePickerDialog;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.os.Bundle;
import android.os.Vibrator;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.TimePicker;
import android.widget.Toast;

public class after_login extends AppCompatActivity{

    private Vibrator vib;
    DBH myDb;
    private session session;

    EditText destin,date,time;
    Button dest,date_button,time_button;
    private int mYear, mMonth, mDay, mHour, mMinute;
    String ax,ay;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_logined);

        date_button = findViewById(R.id.date_button);
        time_button = findViewById(R.id.time_button);

         date = (EditText)findViewById(R.id.date_text);
         time = (EditText)findViewById(R.id.time_text);
        final Button add = (Button) findViewById(R.id.add_button);
        final Button view = findViewById(R.id.view_button);
        final String emailz=start.email_login.getText().toString();

        final Button logout = findViewById(R.id.logout);
        vib = (Vibrator) getSystemService(Context.VIBRATOR_SERVICE);
        session = new session( ctx: this);
        if(!session.loggedin()){
                logout();
        }
        myDb = new DBH( context: this);

        EditText pickup = findViewById(R.id.pickup_text);
        Button pick = findViewById(R.id.pickup_button);

        destin = findViewById(R.id.destination_text);
        dest = findViewById(R.id.destination_button);

        pick.setOnClickListener((view) -> {
                Intent reg = new Intent( packageContext: after_login.this, MapsActivity.class);
                startActivity(reg);
        });
        pickup.setText(MapsActivity.strr);

        logout.setOnClickListener((view) -> { logout(); });

        date_button.setOnClickListener((view) -> { getdate(); });

        time_button.setOnClickListener((view) -> { gettime(); });

        add.setOnClickListener((v) -> {
```

```
92                  @Override
93                  public void onClick(View v) {
94                          boolean isInserted = myDb.insertDate(ax,ay, emailz );
95                          if(isInserted == true) {
96                              Toast.makeText( context: after_login.this,  text: "Sorry! Request failed.",
97                                      Toast.LENGTH_LONG).show();
98                          }
99                          else
100                         {
101                             Toast.makeText( context: after_login.this,
102                                     text: "Your date and time of travel has been registered.",
103                                     Toast.LENGTH_LONG).show();
104
105                         }
106                     }
107         });
108
109         view.setOnClickListener((view) → {
112                 Intent vi = new Intent( packageContext: after_login.this, viewlist.class);
113                 startActivity(vi);
114         });
116
117         dest.setOnClickListener((view) → {
120                 Intent gantavya = new Intent( packageContext: after_login.this,MapsActivity2.class);
121                 startActivity(gantavya);
122         });
124         destin.setText(MapsActivity2.des_loc);
125     }
126
127     private void logout(){
128         session.setLoggedin(false);
129         finish();
130         startActivity(new Intent( packageContext: after_login.this,start.class));
131     }
132
133     private void getdate(){
134             // Get Current Date
135             final Calendar c = Calendar.getInstance();
136             mYear = c.get(Calendar.YEAR);
137             mMonth = c.get(Calendar.MONTH);
138             mDay = c.get(Calendar.DAY_OF_MONTH);
139
140
141             DatePickerDialog datePickerDialog = new DatePickerDialog( context: this,
142                     (view, year, monthOfYear, dayOfMonth) → {
147
148                             date.setText(dayOfMonth + "-" + (monthOfYear + 1) + "-" + year);
149                             ax=dayOfMonth + "-" + (monthOfYear + 1) + "-" + year;
150
151                     }, mYear, mMonth, mDay);
153             datePickerDialog.show();
154     }
155
156     private void gettime(){
157         final Calendar c = Calendar.getInstance();
158         mHour = c.get(Calendar.HOUR_OF_DAY);
159         mMinute = c.get(Calendar.MINUTE);
160
161         // Launch Time Picker Dialog
162         TimePickerDialog timePickerDialog = new TimePickerDialog( context: this,
163                 (view, hourOfDay, minute) → {
168
169                         time.setText(hourOfDay + ":" + minute);
170                         ay=hourOfDay + ":" + minute;
171                 }, mHour, mMinute,  is24HourView: false);
173         timePickerDialog.show();
174     }
175     public void showMessage(String title,String Message){
176         AlertDialog.Builder builder = new AlertDialog.Builder( context: this);
177         builder.setCancelable(true);
178         builder.setTitle(title);
179         builder.setMessage(Message);
180         builder.show();
181     }
182 }
```

5. Pickup

```java
package com.sharethefare.sharethefare;

import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Address;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.support.v4.app.ActivityCompat;
import android.support.v4.app.FragmentActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;

import java.io.IOException;
import java.util.List;

public class MapsActivity extends FragmentActivity implements OnMapReadyCallback {

    private GoogleMap mMap;
    LocationManager locationManager;
    DBH myDb;

    public static String strr;

    @Override
    public void onCreate(Bundle savedInstanceState) {

        myDb = new DBH( context: this);

        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps);
        // Obtain the SupportMapFragment and get notified when the map is ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync( onMapReadyCallback: this);
        locationManager = (LocationManager) getSystemService(LOCATION_SERVICE);
        if (ActivityCompat.checkSelfPermission( context: this,
                android.Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED
                && ActivityCompat.checkSelfPermission( context: this,
                android.Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED)
        {
            // TODO: Consider calling
            //    ActivityCompat#requestPermissions
            // here to request the missing permissions, and then overriding
            //   public void onRequestPermissionsResult(int requestCode, String[] permissions,
            //                                          int[] grantResults)
            // to handle the case where the user grants the permission. See the documentation
            // for ActivityCompat#requestPermissions for more details.
            return;
        }



        locationManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER,
                minTime: 0,  minDistance: 0, new LocationListener() {
            @Override
            public void onLocationChanged(Location location) {
                double latitude = location.getLatitude();
                double longitude = location.getLongitude();
                LatLng latLng = new LatLng(latitude, longitude);
                List<Address> address = null;
                Geocoder geocoder = new Geocoder(getApplicationContext());
```

```
74                  try {
75                      List<Address> addressList = geocoder.getFromLocation(latitude, longitude, maxResults: 1);
76                      strr = addressList.get(0).getAddressLine( index: 0);
77                      mMap.addMarker(new MarkerOptions().position(latLng).title(strr));
78                      mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(latLng, v: 16.5f));
79
80
81                      Log.d( tag: "abcd",strr);
82                  } catch (IOException e) {
83                      e.printStackTrace();
84                  }
85
86              }
87
88              @Override
89              public void onStatusChanged(String provider, int status, Bundle extras) {
90
91              }
92
93              @Override
94              public void onProviderEnabled(String provider) {
95
96              }
97
98              @Override
99              public void onProviderDisabled(String provider) {
100
101             }
102
103         });
104         final Button ok = (Button) findViewById(R.id.button1);
105
106         ok.setOnClickListener((view) -> {
109                 Intent reg = new Intent( packageContext: MapsActivity.this, after_login.class);
110                 String emailz=start.email_login.getText().toString();
111                 boolean isInserted = myDb.insertData1(strr,emailz);
112
113                 if(isInserted == true)
114                     Toast.makeText( context: MapsActivity.this,
115                             text: "Your current location could not be accessed!", Toast.LENGTH_LONG).show();
116                 else
117                 {
118                     Toast.makeText( context: MapsActivity.this,
119                             text: "Your current location has been accessed!", Toast.LENGTH_LONG).show();
120
121                 }
122                 startActivity(reg);
123         });
125     }
126
127
128     /**
129      * Manipulates the map once available.
130      * This callback is triggered when the map is ready to be used.
131      * This is where we can add markers or lines, add listeners or move the camera. In this case,
132      * we just add a marker near Sydney, Australia.
133      * If Google Play services is not installed on the device, the user will be prompted to install
134      * it inside the SupportMapFragment. This method will only be triggered once the user has
135      * installed Google Play services and returned to the app.
136      */
137     @Override
138     public void onMapReady(GoogleMap googleMap) {
139         mMap = googleMap;
140
141         // Add a marker in Sydney and move the camera
142 //        LatLng sydney = new LatLng(-34, 151);
143 //        mMap.addMarker(new MarkerOptions().position(sydney).title("Marker in Sydney"));
144 //        mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
145     }
146 }
147
```

7. Destination

```java
package com.sharethefare.sharethefare;

import android.Manifest;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Address;
import android.location.Geocoder;
import android.location.Location;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.KeyEvent;
import android.view.View;
import android.view.WindowManager;
import android.view.inputmethod.EditorInfo;
import android.widget.AdapterView;
import android.widget.AutoCompleteTextView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.GooglePlayServicesNotAvailableException;
import com.google.android.gms.common.GooglePlayServicesRepairableException;
import com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.common.api.PendingResult;
import com.google.android.gms.common.api.ResultCallback;
import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.location.places.AutocompletePrediction;
import com.google.android.gms.location.places.Place;
import com.google.android.gms.location.places.PlaceBuffer;
import com.google.android.gms.location.places.Places;
import com.google.android.gms.location.places.ui.PlacePicker;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.LatLngBounds;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

/**
 * Created by User on 10/2/2017.
 */

public class MapsActivity2 extends AppCompatActivity implements OnMapReadyCallback,
        GoogleApiClient.OnConnectionFailedListener{

    @Override
    public void onConnectionFailed(@NonNull ConnectionResult connectionResult) {

    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        Toast.makeText( context: this,  text: "Map is Ready", Toast.LENGTH_SHORT).show();
        Log.d(TAG,  msg: "onMapReady: map is ready");
        mMap = googleMap;

```

```java
        if (mLocationPermissionsGranted) {
            getDeviceLocation();

            if (ActivityCompat.checkSelfPermission( context: this,
                    Manifest.permission.ACCESS_FINE_LOCATION)
                    != PackageManager.PERMISSION_GRANTED &&
                    ActivityCompat.checkSelfPermission( context: this,
                    Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED)
            {
                return;
            }
            mMap.setMyLocationEnabled(true);
            mMap.getUiSettings().setMyLocationButtonEnabled(false);

            init();
        }
    }

    private static final String TAG = "MapActivity";

    private static final String FINE_LOCATION = Manifest.permission.ACCESS_FINE_LOCATION;
    private static final String COURSE_LOCATION = Manifest.permission.ACCESS_COARSE_LOCATION;
    private static final int LOCATION_PERMISSION_REQUEST_CODE = 1234;
    private static final float DEFAULT_ZOOM = 15f;
    private static final int PLACE_PICKER_REQUEST = 1;
    private static final LatLngBounds LAT_LNG_BOUNDS = new LatLngBounds(
            new LatLng( v: -40,  v1: -168), new LatLng( v: 71,  v1: 136));

    DBH myDb;
    //widgets
    private AutoCompleteTextView mSearchText;
    private ImageView mGps, mInfo, mPlacePicker;


    //vars
    private Boolean mLocationPermissionsGranted = false;
    private GoogleMap mMap;
    private FusedLocationProviderClient mFusedLocationProviderClient;
    private PlaceAutocompleteAdapter mPlaceAutocompleteAdapter;
    private GoogleApiClient mGoogleApiClient;
    private PlaceInfo mPlace;
    private Marker mMarker;


    public static String des_loc;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps2);

        myDb = new DBH( context: this);

        mSearchText = (AutoCompleteTextView) findViewById(R.id.input_search);
        mGps = (ImageView) findViewById(R.id.ic_gps);
        mInfo = (ImageView) findViewById(R.id.place_info);
        mPlacePicker = (ImageView) findViewById(R.id.place_picker);
        final Button mok = findViewById(R.id.ok2);
        getLocationPermission();
        mok.setOnClickListener((view) -> {
                Intent reg = new Intent( packageContext: MapsActivity2.this, after_login.class);
                startActivity(reg);
                String emailz=start.email_login.getText().toString();
                    boolean isInserted = myDb.insertData2(des_loc,emailz);

                    if(isInserted == true)
                        Toast.makeText( context: MapsActivity2.this,
                                text: "Your destination location could not be accessed!",
                                Toast.LENGTH_LONG).show();
                    else
                    {
                        Toast.makeText( context: MapsActivity2.this,
                                text: "Your destination location has been accessed!",
                                Toast.LENGTH_LONG).show();
```

```
149
150                     }
151                 //}
152             });
154
155         }
156
157         private void init(){
158             Log.d(TAG,  msg: "init: initializing");
159
160             mGoogleApiClient = new GoogleApiClient
161                     .Builder( context: this)
162                     .addApi(Places.GEO_DATA_API)
163                     .addApi(Places.PLACE_DETECTION_API)
164                     .enableAutoManage( fragmentActivity: this,  onConnectionFailedListener: this)
165                     .build();
166
167             mSearchText.setOnItemClickListener(mAutocompleteClickListener);
168
169             mPlaceAutocompleteAdapter = new PlaceAutocompleteAdapter( context: this, mGoogleApiClient,
170                     LAT_LNG_BOUNDS,  filter: null);
171
172             mSearchText.setAdapter(mPlaceAutocompleteAdapter);
173
174             mSearchText.setOnEditorActionListener((textView, actionId, keyEvent) -> {
177                     if(actionId == EditorInfo.IME_ACTION_SEARCH
178                             || actionId == EditorInfo.IME_ACTION_DONE
179                             || keyEvent.getAction() == KeyEvent.ACTION_DOWN
180                             || keyEvent.getAction() == KeyEvent.KEYCODE_ENTER){
181
182                         //execute our method for searching
183                         geoLocate();
184                     }
185
186                     return false;
187             });
189
190             mGps.setOnClickListener((view) -> {
193                     Log.d(TAG,  msg: "onClick: clicked gps icon");
194                     getDeviceLocation();
195             });
197
198             mInfo.setOnClickListener((view) -> {
201                     Log.d(TAG,  msg: "onClick: clicked place info");
202                     try{
203                         if(mMarker.isInfoWindowShown()){
204                             mMarker.hideInfoWindow();
205                         }else{
206                             Log.d(TAG,  msg: "onClick: place info: " + mPlace.toString());
207                             mMarker.showInfoWindow();
208                         }
209                     }catch (NullPointerException e){
210                         Log.e(TAG,  msg: "onClick: NullPointerException: " + e.getMessage() );
211                     }
212             });
214
215             mPlacePicker.setOnClickListener((view) -> {
218
219                     PlacePicker.IntentBuilder builder = new PlacePicker.IntentBuilder();
220
221                     try {
222                         startActivityForResult(builder.build( activity: MapsActivity2.this),
223                                 PLACE_PICKER_REQUEST);
224                     } catch (GooglePlayServicesRepairableException e) {
225                         Log.e(TAG,  msg: "onClick: GooglePlayServicesRepairableException: "
226                                 + e.getMessage() );
227                     } catch (GooglePlayServicesNotAvailableException e) {
228                         Log.e(TAG,  msg: "onClick: GooglePlayServicesNotAvailableException: "
229                                 + e.getMessage() );
230                     }
231             });
233
234
235
236             hideSoftKeyboard();
```

```java
    }

    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        if (requestCode == PLACE_PICKER_REQUEST) {
            if (resultCode == RESULT_OK) {
                Place place = PlacePicker.getPlace( context: this, data);
                //des_loc = place.getName()+" "+place.getAddress();

                PendingResult<PlaceBuffer> placeResult = Places.GeoDataApi.getPlaceById
                        (mGoogleApiClient, place.getId());
                placeResult.setResultCallback(mUpdatePlaceDetailsCallback);
            }
            else {
                super.onActivityResult(requestCode, resultCode, data);
            }
        }
        else {
            super.onActivityResult(requestCode, resultCode, data);
        }
    }

    private void geoLocate(){
        Log.d( tag: "awe",  msg: "geoLocate: geolocating");

        String searchString = mSearchText.getText().toString();

        Geocoder geocoder = new Geocoder( context: MapsActivity2.this);
        List<Address> list = new ArrayList<>();
        try{
            list = geocoder.getFromLocationName(searchString,  maxResults: 1);
        }catch (IOException e){
            Log.e(TAG,  msg: "geoLocate: IOException: " + e.getMessage() );
        }

        if(list.size() > 0){
            Address address = list.get(0);

            Log.d( tag: "awer",  msg: "geoLocate: found a location: " + address.toString());
            //Toast.makeText(this, address.toString(), Toast.LENGTH_SHORT).show();
            //des_loc = list.get(0).getAddressLine(0);
            moveCamera(new LatLng(address.getLatitude(), address.getLongitude()), DEFAULT_ZOOM,
                    address.getAddressLine( index: 0));
        }
    }

    private void getDeviceLocation(){
        Log.d(TAG,  msg: "getDeviceLocation: getting the devices current location");

        mFusedLocationProviderClient = LocationServices.getFusedLocationProviderClient( activity: this);

        try{
            if(mLocationPermissionsGranted){

                final Task location = mFusedLocationProviderClient.getLastLocation();
                location.addOnCompleteListener((task) -> {
                        if(task.isSuccessful()){
                            Log.d(TAG,  msg: "onComplete: found location!");
                            Location currentLocation = (Location) task.getResult();

                            moveCamera(new LatLng(currentLocation.getLatitude(),
                                            currentLocation.getLongitude()),
                                    DEFAULT_ZOOM,
                                    title: "My Location");

                        }else{
                            Log.d(TAG,  msg: "onComplete: current location is null");
                            Toast.makeText( context: MapsActivity2.this,
                                    text: "unable to get current location", Toast.LENGTH_SHORT).show();
                        }
                });
            }
        }catch (SecurityException e){
            Log.e(TAG,  msg: "getDeviceLocation: SecurityException: " + e.getMessage() );
        }
    }
```

```java
    private void moveCamera(LatLng latLng, float zoom, PlaceInfo placeInfo){
        Log.d(TAG, msg: "moveCamera: moving the camera to: lat: " +
                latLng.latitude + ", lng: " + latLng.longitude );
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(latLng, zoom));


        mMap.clear();

        mMap.setInfoWindowAdapter(new CustomInfoWindowAdapter( context: MapsActivity2.this));

        if(placeInfo != null){
            try{

                String snippet = "Address: " + placeInfo.getAddress();
                //des_loc = placeInfo.getName()+" "+placeInfo.getAddress();
                MarkerOptions options = new MarkerOptions()
                        .position(latLng)
                        .title(placeInfo.getName())
                        .snippet(snippet);
                mMarker = mMap.addMarker(options);

            }catch (NullPointerException e){
                Log.e(TAG, msg: "moveCamera: NullPointerException: " + e.getMessage() );
            }
        }else{
            mMap.addMarker(new MarkerOptions().position(latLng));
        }

        hideSoftKeyboard();
    }

    private void moveCamera(LatLng latLng, float zoom, String title){
        Log.d(TAG, msg: "moveCamera: moving the camera to: lat: " + latLng.latitude
                + ", lng: " + latLng.longitude );
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(latLng, zoom));


        if(!title.equals("My Location")){
            MarkerOptions options = new MarkerOptions().position(latLng).title(title);
            mMap.addMarker(options);
        }

        hideSoftKeyboard();
    }

    private void initMap(){
        Log.d(TAG, msg: "initMap: initializing map");
        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager().
                findFragmentById(R.id.map);

        mapFragment.getMapAsync( onMapReadyCallback: MapsActivity2.this);
    }

    private void getLocationPermission(){
        Log.d(TAG, msg: "getLocationPermission: getting location permissions");
        String[] permissions = {Manifest.permission.ACCESS_FINE_LOCATION,
                Manifest.permission.ACCESS_COARSE_LOCATION};

        if(ContextCompat.checkSelfPermission(this.getApplicationContext(),
                FINE_LOCATION) == PackageManager.PERMISSION_GRANTED){
            if(ContextCompat.checkSelfPermission(this.getApplicationContext(),
                    COURSE_LOCATION) == PackageManager.PERMISSION_GRANTED){
                mLocationPermissionsGranted = true;
                initMap();
            }else{
                ActivityCompat.requestPermissions( activity: this,
                        permissions,
                        LOCATION_PERMISSION_REQUEST_CODE);
            }
        }else{
            ActivityCompat.requestPermissions( activity: this,
                    permissions,
                    LOCATION_PERMISSION_REQUEST_CODE);
        }
```

```java
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
                                           @NonNull int[] grantResults) {
        Log.d(TAG, msg: "onRequestPermissionsResult: called.");
        mLocationPermissionsGranted = false;

        switch(requestCode){
            case LOCATION_PERMISSION_REQUEST_CODE:{
                if(grantResults.length > 0){
                    for(int i = 0; i < grantResults.length; i++){
                        if(grantResults[i] != PackageManager.PERMISSION_GRANTED){
                            mLocationPermissionsGranted = false;
                            Log.d(TAG, msg: "onRequestPermissionsResult: permission failed");
                            return;
                        }
                    }
                    Log.d(TAG, msg: "onRequestPermissionsResult: permission granted");
                    mLocationPermissionsGranted = true;
                    //initialize our map
                    initMap();
                }
            }
        }
    }
```

```java
                Log.d(TAG, msg: "onResult: Place query did not complete successfully: "
                        + places.getStatus().toString());
                places.release();
                return;
            }
            final Place place = places.get(0);

            try{
                mPlace = new PlaceInfo();
                mPlace.setName(place.getName().toString());
                Log.d(TAG, msg: "onResult: name: " + place.getName());
                mPlace.setAddress(place.getAddress().toString());
                Log.d(TAG, msg: "onResult: address: " + place.getAddress());
//                mPlace.setAttributions(place.getAttributions().toString());
//                Log.d(TAG, "onResult: attributions: " + place.getAttributions());
                mPlace.setId(place.getId());
                Log.d(TAG, msg: "onResult: id:" + place.getId());
                mPlace.setLatlng(place.getLatLng());
                Log.d(TAG, msg: "onResult: latlng: " + place.getLatLng());
                mPlace.setRating(place.getRating());
                Log.d(TAG, msg: "onResult: rating: " + place.getRating());
                mPlace.setPhoneNumber(place.getPhoneNumber().toString());
                Log.d(TAG, msg: "onResult: phone number: " + place.getPhoneNumber());
                mPlace.setWebsiteUri(place.getWebsiteUri());
                Log.d(TAG, msg: "onResult: website uri: " + place.getWebsiteUri());

                Log.d(TAG, msg: "onResult: place: " + mPlace.toString());

                des_loc= place.getName().toString() + " " + place.getAddress().toString();
                Log.d( tag: "45",des_loc);

                Log.d( tag: "kl",des_loc);

            }catch (NullPointerException e){
                Log.e(TAG, msg: "onResult: NullPointerException: " + e.getMessage() );
            }

            moveCamera(new LatLng(place.getViewport().getCenter().latitude,
                    place.getViewport().getCenter().longitude), DEFAULT_ZOOM, mPlace);

            places.release();
        };
    }
```

8. Database Helper

```java
package com.sharethefare.sharethefare;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import java.util.ArrayList;

public class DBH extends SQLiteOpenHelper {

    public static final String DATABASE_NAME = "Userx.db";
    public static final String TABLE_NAME = "user_info";

    public static final String COL_1 = "ID";
    public static final String COL_2 = "Name";
    public static final String COL_3 = "Mobile";
    public static final String COL_4 = "Email";
    public static final String COL_5 = "Password";
    public static final String COL_6 = "Pickup";
    public static final String COL_7 = "Destination";
    public static final String COL_8 = "Date";
    public static final String COL_9 = "Time";
    public static final String COL_10 = "Confirm";

    ArrayList<String> arr = new ArrayList<~>();
    public DBH(Context context) { super(context, DATABASE_NAME, factory: null, version: 1); }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("create table " + TABLE_NAME +" (" + COL_1 +" " +
                "INTEGER PRIMARY KEY AUTOINCREMENT," + COL_2 +" TEXT,"
                + COL_3 +" TEXT," + COL_4 +" TEXT," + COL_5 + " TEXT,"+ COL_6
                + " TEXT,"+ COL_7 + " TEXT,"+ COL_8 +" TEXT,"
        + COL_9 + " TEXT,"+ COL_10 + " TEXT)");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS "+TABLE_NAME);
        onCreate(db);
    }

    public boolean insertData(String name,String mobile,String email, String password) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues contentValues = new ContentValues();
        contentValues.put(COL_2,name);
        contentValues.put(COL_3,mobile);
        contentValues.put(COL_4,email);
        contentValues.put(COL_5,password);
        long result = db.insert(TABLE_NAME, nullColumnHack: null ,contentValues);
        if(result == -1)
            return false;
        else
            return true;
    }

    public boolean check(String email, String password){
        String str;
        SQLiteDatabase db = this.getWritableDatabase();
        Cursor res = db.rawQuery( sql: "select "+ COL_4+" from "+TABLE_NAME+" where "
                +COL_4+" ="+"'"+email+"' and "+COL_5+" ="+"'"+password+"'", selectionArgs: null);

        if(res.moveToFirst()){
            return true;
        }
        else
            return false;

    }

    public boolean insertData1(String pickup, String email) {
```

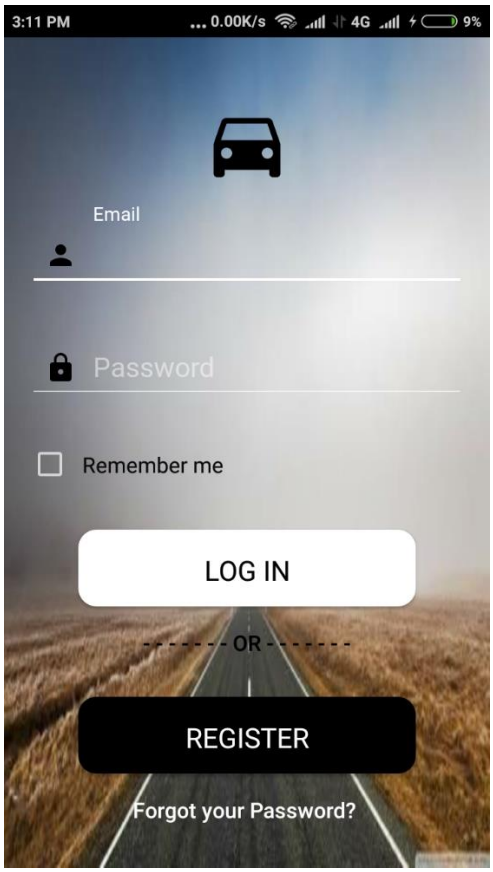
```java
        else
            return false;

    }

    public boolean insertDate(String date,String time, String email) {
        SQLiteDatabase db = this.getWritableDatabase();
        Cursor res = db.rawQuery( sql: "update  "+TABLE_NAME+" set " +COL_8 +
                "="+"'"+date+"',"+COL_9 +
                "="+"'"+time+"' where "+COL_4+" ="+"'"+email+"'", selectionArgs: null);
        if(res.moveToFirst()){
            return true;
        }
        else
            return false;

    }

    public boolean insertData2(String desti, String email) {

        SQLiteDatabase db = this.getWritableDatabase();
        Cursor res = db.rawQuery( sql: "update  "+TABLE_NAME+" set " +COL_7 +
                "="+"'"+desti+"' where "+COL_4+" ="+"'"+email+"'", selectionArgs: null);
        if(res.moveToFirst()){
            return true;
        }
        else
            return false;

    }


    public Cursor getAllData() {
        SQLiteDatabase db = this.getWritableDatabase();
        Cursor res = db.rawQuery( sql: "select "+ COL_2 + "," + COL_3 + "," + COL_6
                + "," + COL_7 + "," + COL_8 + "," + COL_9 + " from "+TABLE_NAME, selectionArgs: null);
        return res;
    }
}
```

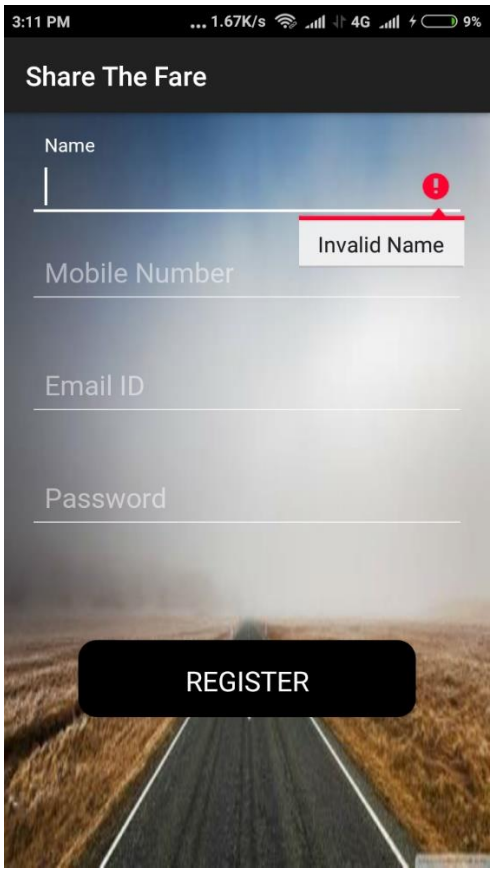
9. View List

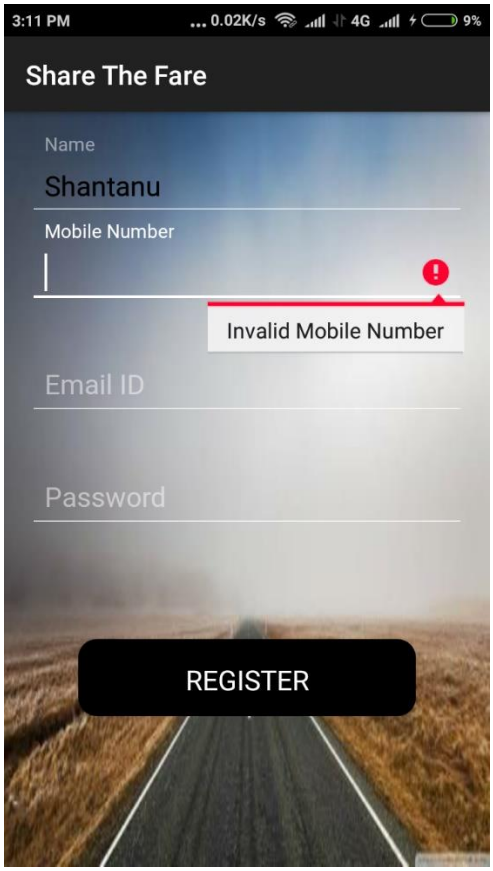
```java
package com.sharethefare.sharethefare;

import android.content.Intent;
import android.database.Cursor;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class viewlist extends AppCompatActivity {
    private session session;
    DBH myDb;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_viewlist);
        final TextView vieww = findViewById(R.id.view_list);
        final Button logout = findViewById(R.id.logoutx);
        session = new session( ctx: this);
        if(!session.loggedin()){
            logout();
        }
        myDb = new DBH( context: this);
```

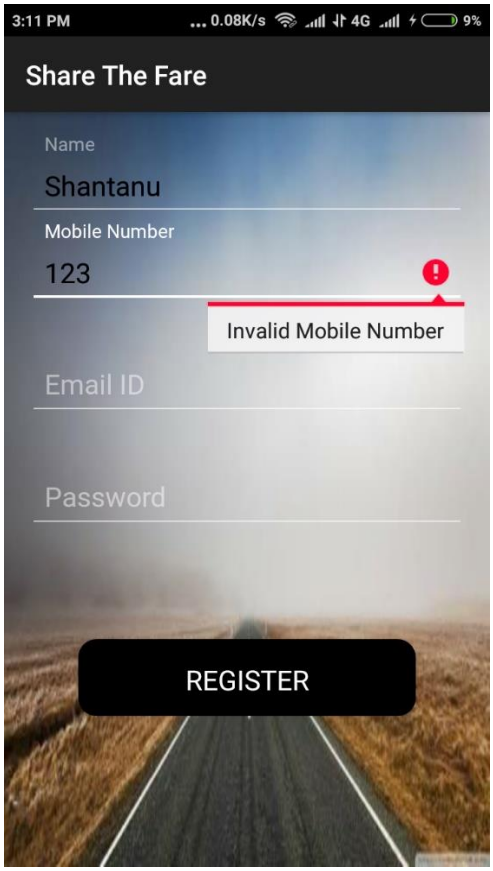
```java
        logout.setOnClickListener((view) -> { logout(); });

        Cursor res = myDb.getAllData();
        if(res.getCount() == 0) {
            // show message
            showMessage( title: "Error", Message: "Nothing found");
            return;
        }

        StringBuffer buffer = new StringBuffer();
        while (res.moveToNext()) {
            buffer.append("Name :"+ res.getString( i: 0)+"\n");
            buffer.append("Mobile :"+ res.getString( i: 1)+"\n");
            buffer.append("Pickup :"+ res.getString( i: 2)+"\n");
            buffer.append("Destination :"+ res.getString( i: 3)+"\n");
            buffer.append("Date :"+ res.getString( i: 4)+"\n");
            buffer.append("Time :"+ res.getString( i: 5)+"\n\n\n");
        }

        // Show all data
        //showMessage("Data",buffer.toString());
        vieww.setText(buffer.toString());

    }
    private void logout(){
        session.setLoggedin(false);
        finish();
        startActivity(new Intent( packageContext: viewlist.this,start.class));
    }

    public void showMessage(String title,String Message){
        AlertDialog.Builder builder = new AlertDialog.Builder( context: this);
        builder.setCancelable(true);
        builder.setTitle(title);
        builder.setMessage(Message);
        builder.show();
    }

}
```

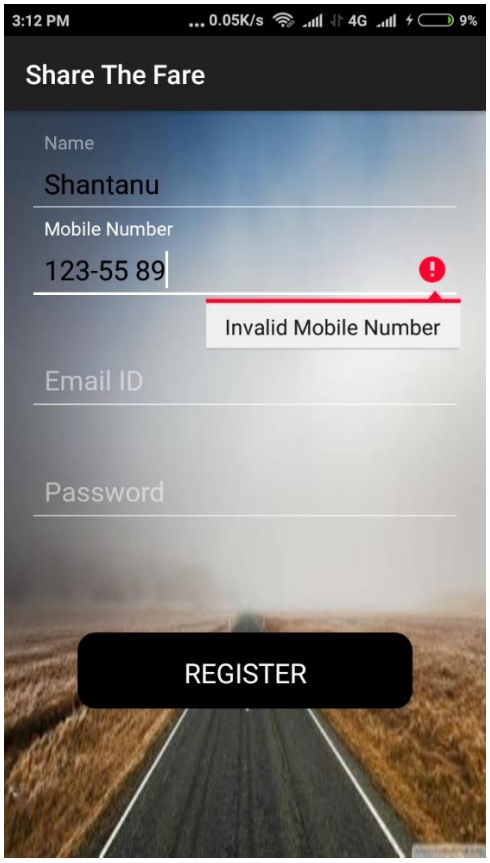
3.5 Testing

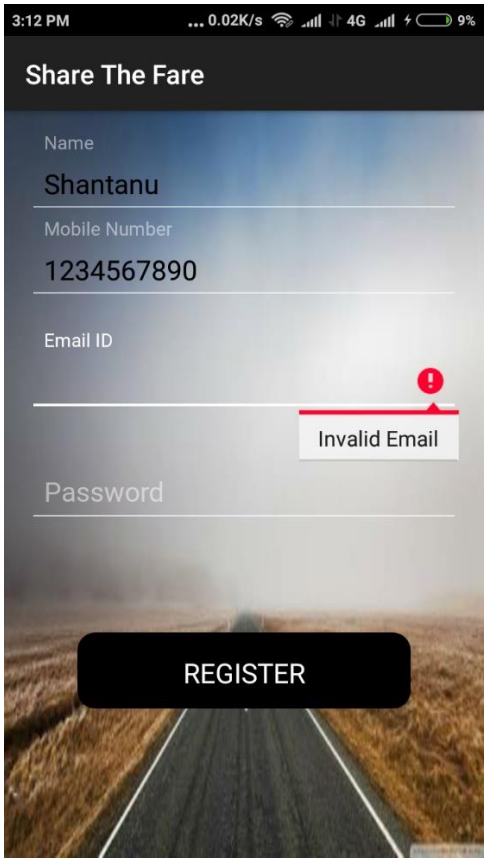

3:11 PM
0.00K/s
4G
9%
Email
Password
Remember me
LOG IN
OR
REGISTER
Forgot your Password?

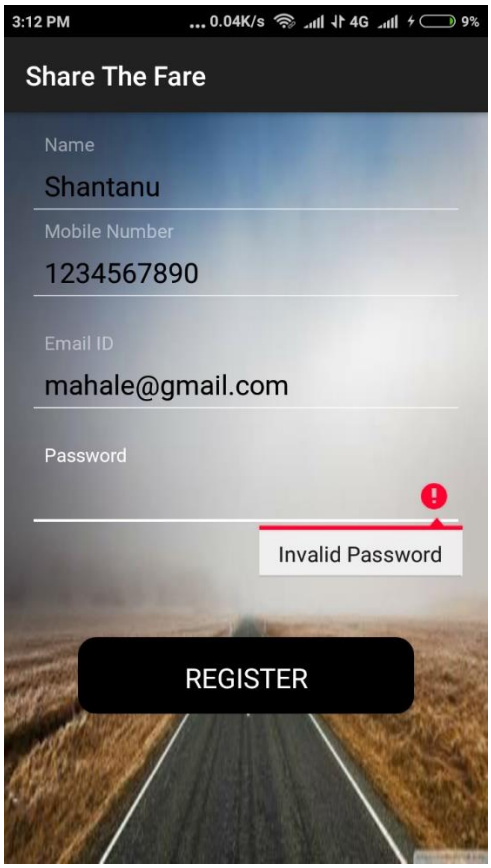

3:11 PM
1.67K/s
4G
9%
Share The Fare
Name
Invalid Name
Mobile Number
Email ID
Password
REGISTER

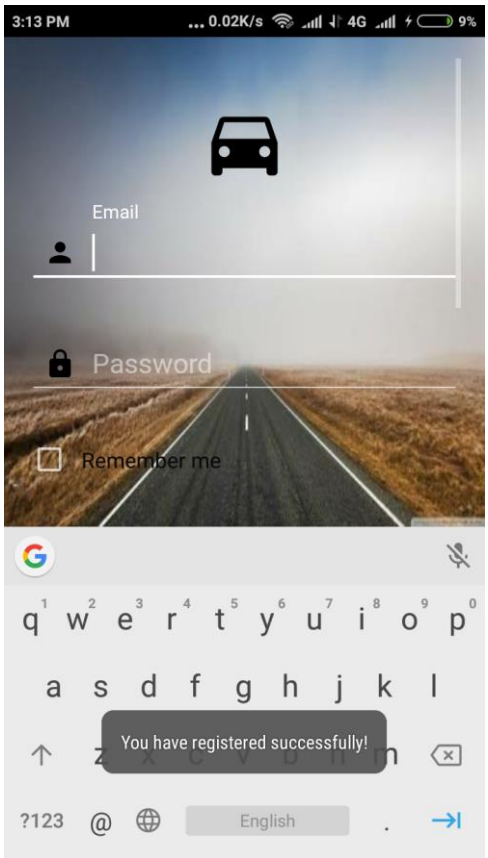

3:11 PM
0.02K/s
4G
9%
Share The Fare
Name
Shantanu
Mobile Number
Invalid Mobile Number
Email ID
Password
REGISTER

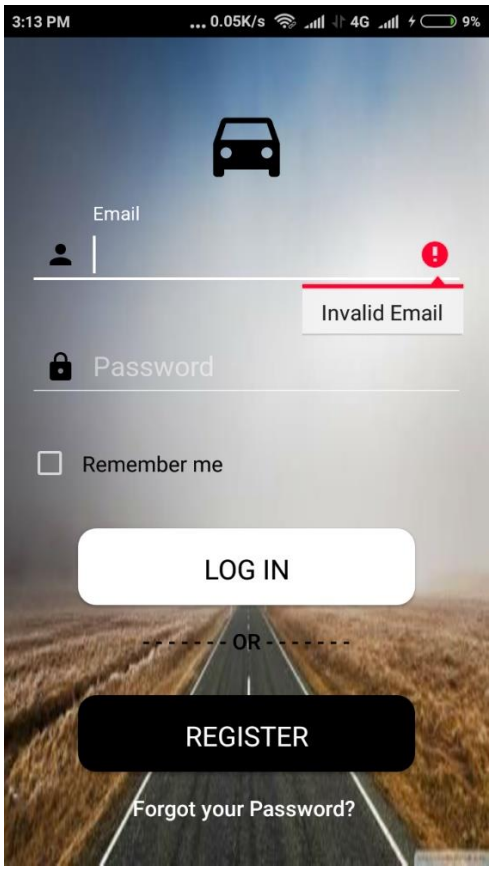

3:11 PM
0.08K/s
4G
9%
Share The Fare
Name
Shantanu
Mobile Number
123
Invalid Mobile Number
Email ID
Password
REGISTER

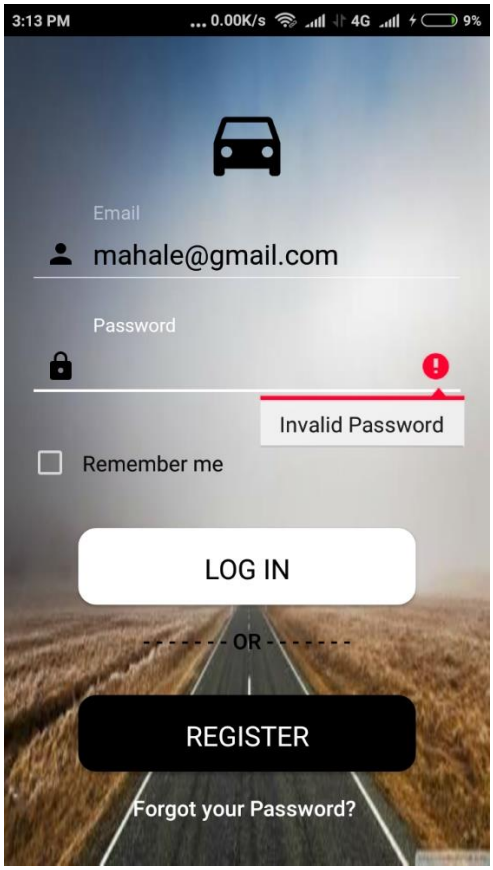

3:12 PM
0.05K/s
4G
9%
Share The Fare
Name
Shantanu
Mobile Number
123-55 89
Invalid Mobile Number
Email ID
Password
REGISTER

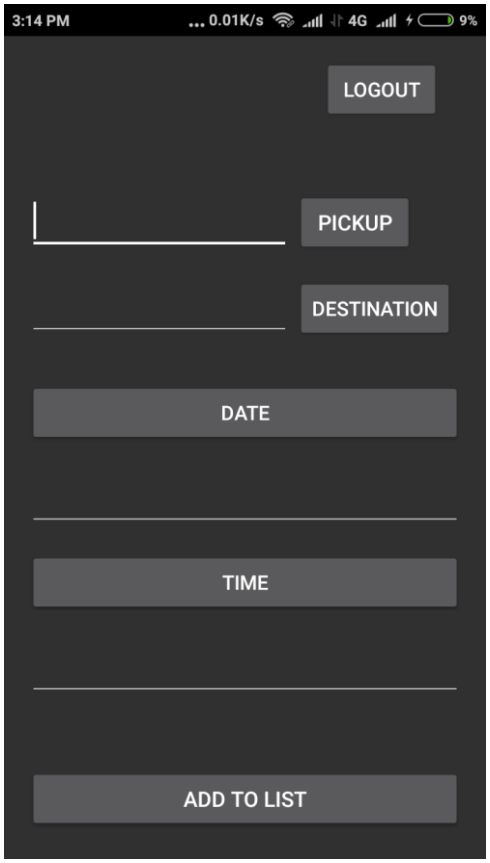

3:12 PM
0.02K/s
4G
9%
Share The Fare
Name
Shantanu
Mobile Number
1234567890
Email ID
Invalid Email
Password
REGISTER


3:12 PM
0.04K/s
4G
9%
Share The Fare
Name
Shantanu
Mobile Number
1234567890
Email ID
mahale@gmail.com
Password
Invalid Password
REGISTER

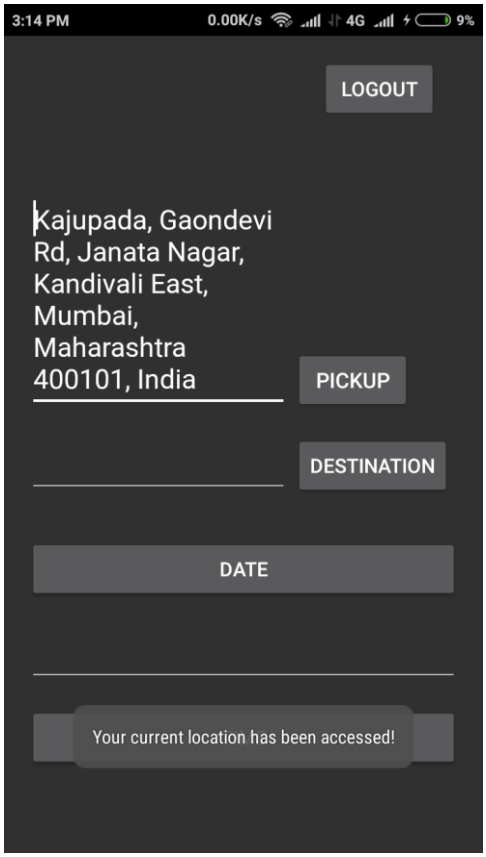

3:13 PM
0.02K/s
4G
9%
Email
Password
Remember me
You have registered successfully!
?123
English

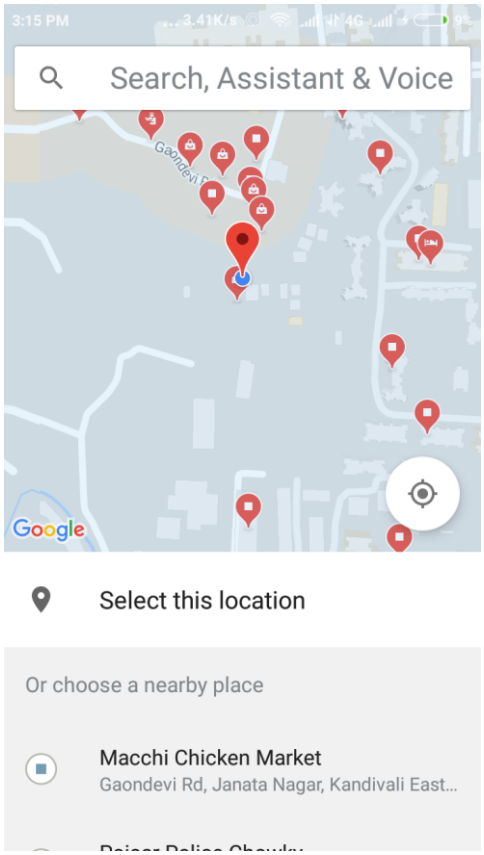

3:13 PM
0.05K/s
4G
9%
Email
Invalid Email
Password
Remember me
LOG IN
OR
REGISTER
Forgot your Password?

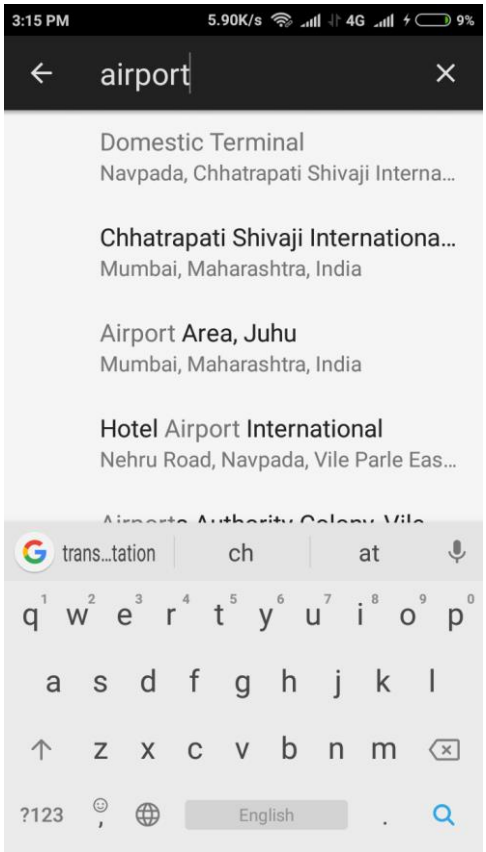

3:13 PM
0.00K/s
4G
9%
Email
mahale@gmail.com
Password
Invalid Password
Remember me
LOG IN
OR
REGISTER
Forgot your Password?

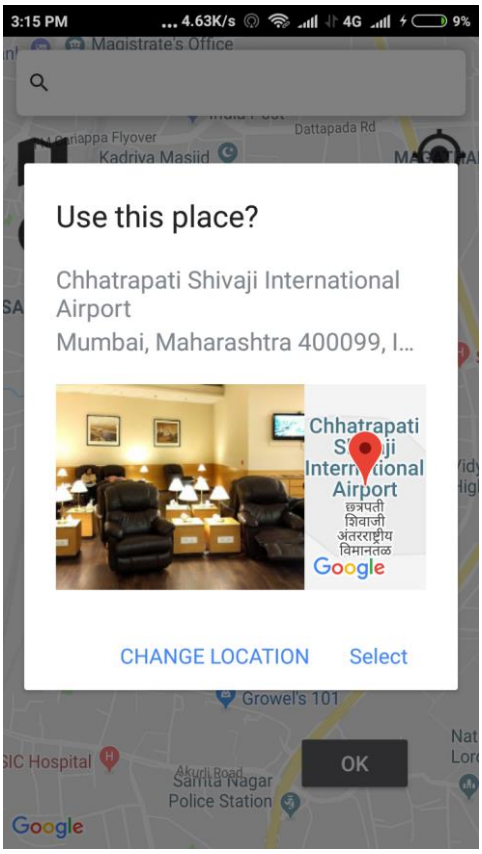

3:14 PM
0.01K/s
4G
9%
LOGOUT
PICKUP
DESTINATION
DATE
TIME
ADD TO LIST

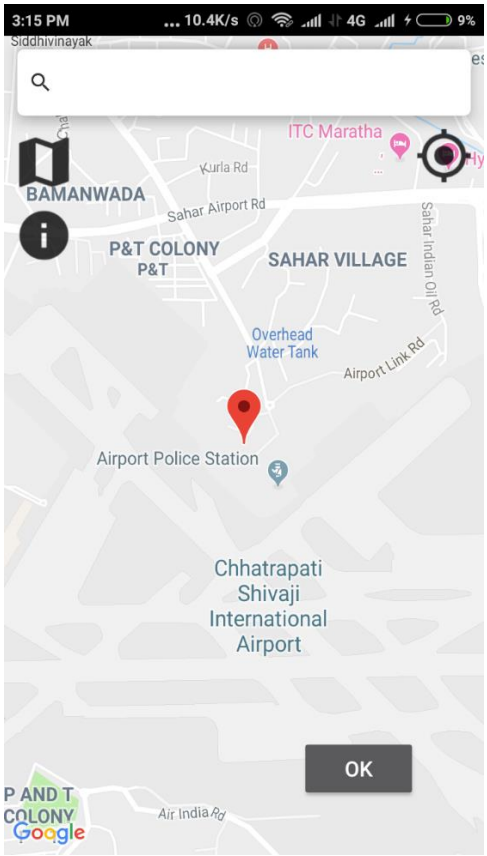

3:14 PM
2.23K/s
4G
9%
OK
Gaondevi Mandir
FCI Rd
Office of the Deputy Chief Engineer...
90 Feet Rd
Poisar Indra Chowk
St. Lawrence High School
Bihari Tekdi Rd
Cambridge School
Google

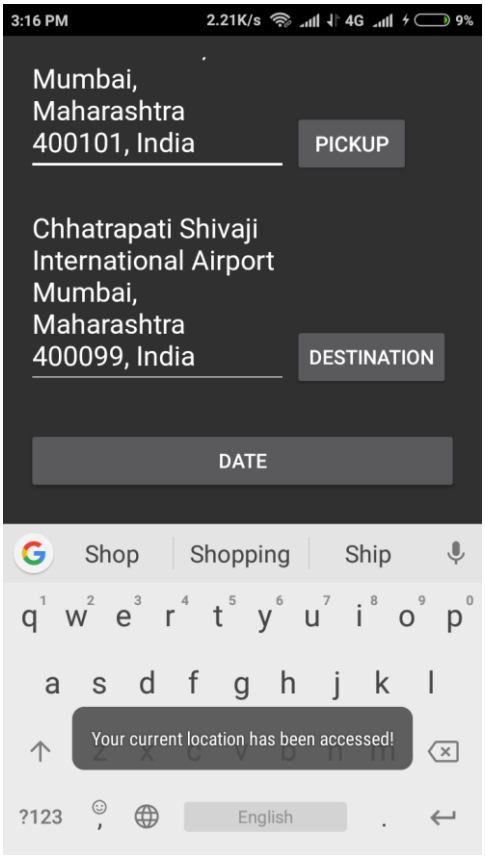

3:14 PM
0.00K/s
4G
9%
LOGOUT
Kajupada, Gaondevi Rd, Janata Nagar, Kandivali East, Mumbai, Maharashtra 400101, India
PICKUP
DESTINATION
DATE
Your current location has been accessed!


3:15 PM
Search, Assistant & Voice
Google
Select this location
Or choose a nearby place
Macchi Chicken Market
Gaondevi Rd, Janata Nagar, Kandivali East...

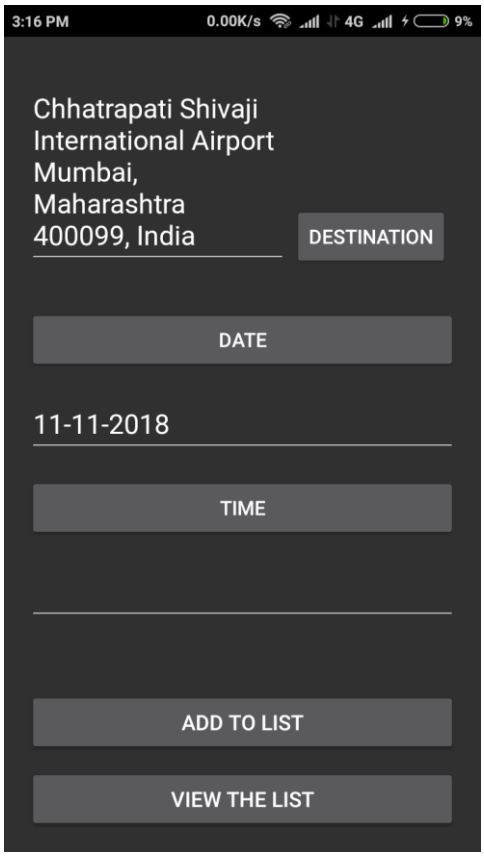

3:15 PM
5.90K/s
4G
9%
airport
Domestic Terminal
Navpada, Chhatrapati Shivaji Interna...
Chhatrapati Shivaji Internationa...
Mumbai, Maharashtra, India
Airport Area, Juhu
Mumbai, Maharashtra, India
Hotel Airport International
Nehru Road, Navpada, Vile Parle Eas...
trans...tation
ch
at
English

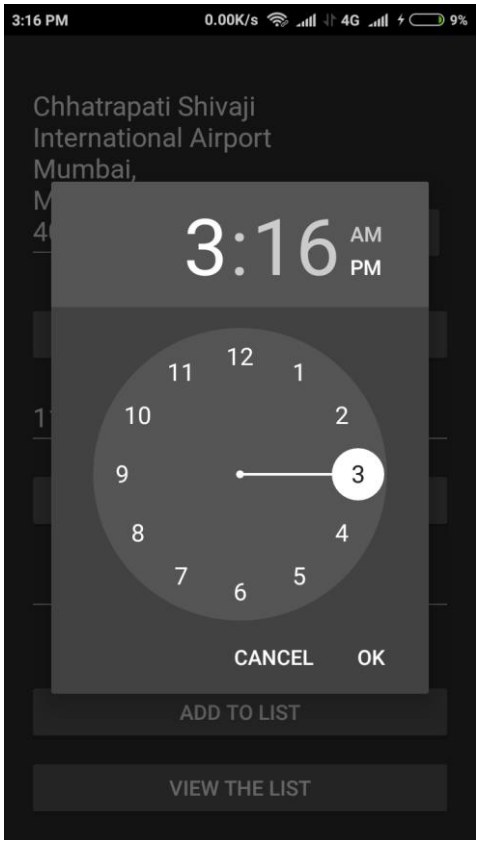

3:15 PM
4.63K/s
4G
9%
Use this place?
Chhatrapati Shivaji International Airport
Mumbai, Maharashtra 400099, I...
CHANGE LOCATION
Select
OK
Google

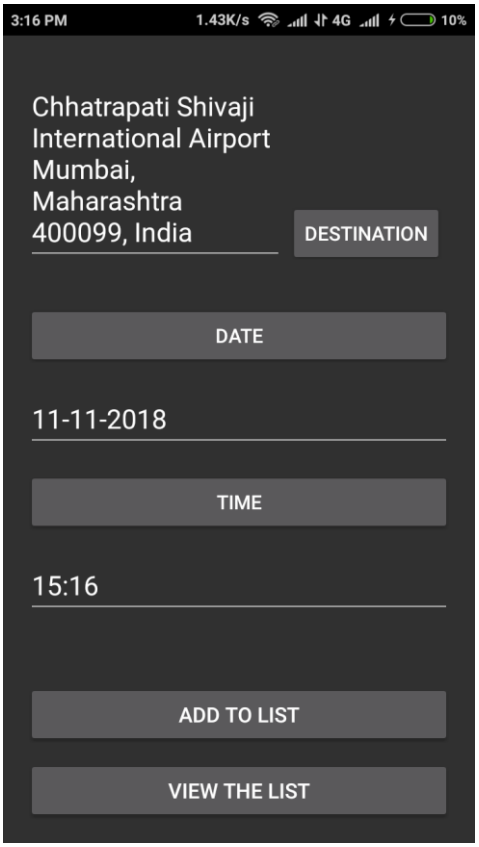

3:15 PM
10.4K/s
4G
9%
Siddhivinayak
ITC Maratha
Kurla Rd
BAMANWADA
Sahar Airport Rd
P&T COLONY
P&T
SAHAR VILLAGE
Sahar Indian Oil Rd
Overhead
Water Tank
Airport Link Rd
Airport Police Station
Chhatrapati
Shivaji
International
Airport
OK
P AND T
COLONY
Air India Rd
Google

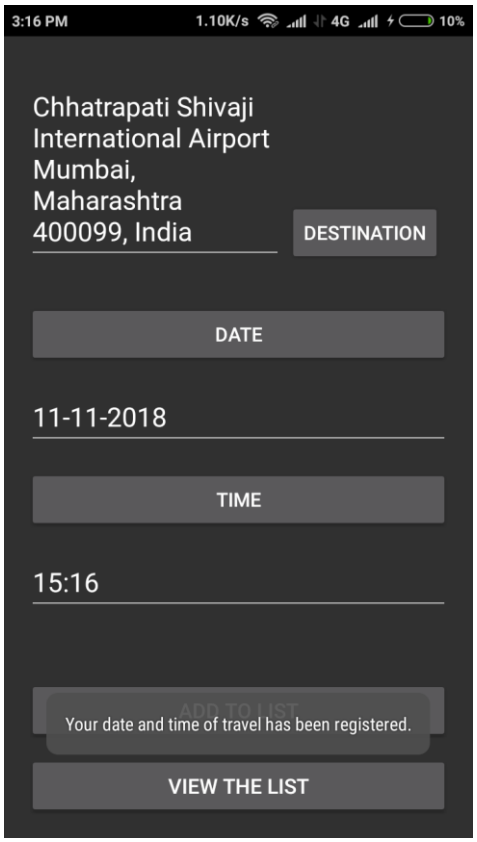

3:16 PM
2.21K/s
4G
9%
Mumbai,
Maharashtra
400101, India
PICKUP
Chhatrapati Shivaji
International Airport
Mumbai,
Maharashtra
400099, India
DESTINATION
DATE
Shop
Shopping
Ship
Your current location has been accessed!
?123
English

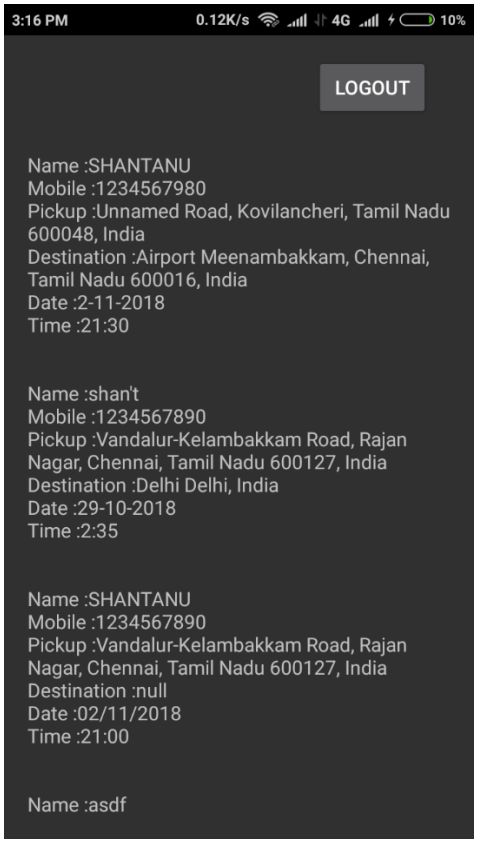

3:16 PM
0.02K/s
4G
9%
2018
Sun, Nov 11
November 2018
S M T W T F S
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30
CANCEL
OK
VIEW THE LIST

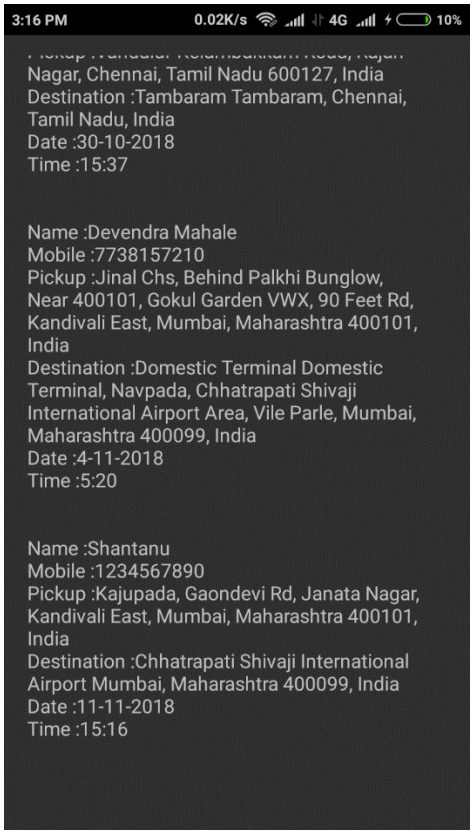

3:16 PM
0.00K/s
4G
9%
Chhatrapati Shivaji
International Airport
Mumbai,
Maharashtra
400099, India
DESTINATION
DATE
11-11-2018
TIME
ADD TO LIST
VIEW THE LIST


3:16 PM
0.00K/s
4G
9%
Chhatrapati Shivaji
International Airport
Mumbai,
3:16
AM
PM
12
1
2
3
4
5
6
7
8
9
10
11
CANCEL
OK
ADD TO LIST
VIEW THE LIST


3:16 PM
1.43K/s
4G
10%
Chhatrapati Shivaji
International Airport
Mumbai,
Maharashtra
400099, India
DESTINATION
DATE
11-11-2018
TIME
15:16
ADD TO LIST
VIEW THE LIST


3:16 PM
1.10K/s
4G
10%
Chhatrapati Shivaji
International Airport
Mumbai,
Maharashtra
400099, India
DESTINATION
DATE
11-11-2018
TIME
15:16
Your date and time of travel has been registered.
VIEW THE LIST


3:16 PM
0.12K/s
4G
10%
LOGOUT
Name :SHANTANU
Mobile :1234567980
Pickup :Unnamed Road, Kovilancheri, Tamil Nadu 600048, India
Destination :Airport Meenambakkam, Chennai, Tamil Nadu 600016, India
Date :2-11-2018
Time :21:30
Name :shan't
Mobile :1234567890
Pickup :Vandalur-Kelambakkam Road, Rajan Nagar, Chennai, Tamil Nadu 600127, India
Destination :Delhi Delhi, India
Date :29-10-2018
Time :2:35
Name :SHANTANU
Mobile :1234567890
Pickup :Vandalur-Kelambakkam Road, Rajan Nagar, Chennai, Tamil Nadu 600127, India
Destination :null
Date :02/11/2018
Time :21:00
Name :asdf



1. The app doesn't accept empty fields.
2. The app doesn't accept invalid:
    - Name
    - Mobile number (should be 10 digits without spaces or other characters)
    - Email (should be of the format zz@zz.zz)
    - Password (should not be empty)
3. The app doesn't allow invalid user login:
    - Email (only the one that is registered)
    - Password (should match with that in database)
4. The app does not give any runtime errors (i.e. the app will not crash. But it can throw exception).

CHAPTER 4

CONCLUSION

The application in its core idea, can remove the inconveniences that exist in the existing systems. The basic idea is to cab-pool. Cab-pooling not only helps in saving the customers' money but it is also environmentally friendly (cab-pooling is a type of car-pooling). But such a product demands cooperation and coordination between the fellow passengers, who are cab-pooling together.

The app constructed based on the presented idea is just a prototype. For real life implementation, there has to be a lot of things in place, like the grouping of the passengers should be in a fixed number and algorithmic.

Further, if the application is developed to the extent of booking cabs/taxis, it has to be more sophisticated and more secure than it is now.

Many facilities like changing of email, password, forgot email or forgot password, remembering the user who is logged in, OTP services, automatic GPS enabling has to be still included. What the user wants, is totally dependent on the functionality of the app.

Live – location tracking, traffic congestions, and safety issues must also be taken into consideration.

As of now, the model successfully represents that 'Share the Fare' provides efficient method of carpooling.

REEFRENCES

Resources used:

1. Android Studio Software
2. Real Device – Redmi phone
3. DB Browser for SQLite software