APPLICATION FOR DIGITAL AMBULANCE SERVICES

A PROJECT REPORT

Submitted by

JACOB JABAKUMAR S (110819104012)

SHYAM M V (110819104303)

SURIYA J (110819104036)

NITHISH KUMAR M (110819104302)

JOTHI LINGAM S (110819104016)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

JAYA ENGINEERING COLLEGE, THIRUNINRAVUR

ANNA UNIVERSITY: CHENNAI 600 025

JUNE 2022

ANNA UNIVERSITY CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **" APPLICATION FOR DIGITAL AMBULANCE SERVICES '' ,** is the bonafide work of **" *JACOB JABAKUMAR S (110819104012), SHYAM M V (110819104303), SURIYA J (110819104036), NITHISH KUMAR M (110819104302), JOTHI LINGAM S (110819104016)*"** who carried out the project work under my supervision.

Prof. M. KUMARAN

Head of the Department,

Professor,

Department of CSE,

Jaya Engineering College,

Thiruninravur -602024.

J . LIN EBY CHANDRA

Supervisor &

Professor,

Department of CSE,

Jaya Engineering College,

Thiruninravur -602024.

ANNA UNIVERSITY CHENNAI 600 025

VIVA VOCE EXAMINATION

The viva-voce examination of the project work titled **"APPLICATION FOR DIGITAL AMBULANCE SERVICES "**, submitted by **"*JACOB JABAKUMAR S (110819104012),SHYAM M V (110819104303), SURIYA J (110819104036), NITHISH KUMAR M (110819104302), JOTHI LINGAM S (110819104016)*"**

Held on ————————

INTERNAL EXAMINER **EXERNAL EXAMINER**

ABSTRACT

An emergency is a situation that poses an immediate risk to health, life, property, or environment. Most emergencies require urgent intervention to prevent a worsening of the situation, although in some situations, mitigation may not be possible and agencies may only be able to offer palliative care for the aftermath.

While some emergencies are self-evident (such as a natural disaster that threatens many lives), many smaller incidents require that an observer (or affected party) decide whether it qualifies as an emergency. The precise definition of an emergency, the agencies involved and the procedures used, vary by jurisdiction, and this is usually set by the government, whose agencies (emergency services) are responsible for emergency planning and management. An ambulance is a medically equipped vehicle which transports patients to treatment facilities, such as hospitals. In some instances, out-of-hospital medical care is provided to the patient. Ambulances are used to respond to medical emergencies by emergency medical services. For this purpose, they are generally equipped with flashing warning lights and sirens. They can rapidly transport paramedics and other first responders to the scene, carry equipment for administering emergency care and transport patients to hospital or other definitive care. Most ambulances use a design based on vans or pick-up trucks. Others take the form of motorcycles, cars, buses, aircraft and boats.

CONTENTS

INTRODUCTION

1.1 Purpose

The primary focus of an Ambulance Service Team is two-fold: The first is to reach people in emergency situations as quickly as possible and administer life-saving first-aid on the spot. The second is to transport the sick or injured patient as quickly as possible to the appropriate healthcare facility for further care.

Emergency medical services (EMS), also known as ambulance

services or paramedic services, are emergency services which treat illnesses and injuries that require an urgent medical response, providing out-of-hospital treatment and transport to definitive care. They may also be known as a first aid squad, FAST squad, emergency squad, rescue squad, ambulance

squad, ambulance corps, life squad or by other initialisms such as EMAS or EMARS.

1.2 Document Conventions

The document is written on ARIAL font with font size of 12. Headings the written at BOLD. All diagrams are mentioned with TITLE.

1.3 Intended Audience and Reading Suggestions

The document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. The document should be read in the sequence that is to be written.

1.4 Project Scope

The project focus on the reducing the number of deaths caused due to emergency condition like accidents, heath problems of old people or pregnant women

Overall Description

2.1 Product Perspective

This product is a new, self-contained product based on the present emergency condition of our society. It will help users to get ambulance services in any emergency condition. People will not have to call and wait for ambulance. Using this product users can track the ambulance.

2.2 Product Features

This product provides ambulance services to the users in emergency condition. In very serious case the software automatically selects the nearest located hospital from where the ambulance will come. Or user can even select the hospital from where user want an ambulance. This product only provides the ambulance services from respective hospitals in users locality

2.3 User Classes and Characteristics

This product has 4 users.

Customers :- Use the ambulance service provides by the product.

Hospitals :- Will register and add drivers for their respective hospitals.

Drivers :- Will pick up the customers from the location with their ambulance.

Admin :- Controls all activities.

2.4 Operating Environment

This products will primarily be launched on ANDROID operating system with primary working version of Android 4.4 KitKat. Later we will work on IOS as well as WINDOWS and WEB.

2.5 Design and Implementation Constraints

The global schema, fragmentation schema, and allocation schema.

There will be 3 application (one for hospital, one for drivers, one for customers)

FireBase Database commands for above queries/applications

How the response for application 1, 2 and 3 will be generated. Assuming these are global queries. Explain how various fragments will be combined to do so.

Implement the database at least using a centralized database management system.

Requirement Analysis

3.1 External Interface Requirements

3.1.1 User Interfaces

Our software is an android app running on android OS. Any user having an android device with ANDROID version 4.4 and above can use this app.

3.1.2 Hardware Interfaces

User need a android device to use this software. Map is the most important interface in this software. All data are stored on online database.

3.1.3 Software Interfaces

The software is build on ANDROID operating system with minimum android version 4.4. The database use is FIREBASE realtime database.we have used google apis for maps and other purpose also. Its main component is GOOGLE MAPS services. Many libaries like GEOFIRE, GOOGLE MAPS, FIREBASE are used. The whole software is built on ANDROID STUDIO BUMBLEBEE 2021.1.1

3.2 Other Nonfunctional Requirements

3.2.1 Performance Requirements

Performance of a system the following must be clearly specified:

- **Response Time**

  The response time of the software depends upon the HOSPITAL response or can depend on the response of the driver of the ambulance.

- **Workload**

  Since we are using firebase database therefore we have very limited querying and indexing, No aggregation, No map reduce, Can't query or list users or stored files.

- **Platform**

  The products is android base so use should have a android smartphone with min android version 4.2.

3.2.2 Safety Requirements

There is no possible loss, damage or harm that user will have using this software. This app monitors your current location.

3.2.3 Security Requirements

The user need not to worry about his/her data. It is fully secured. User should have a gmail id and a working phone number for registration. OTP will be sent to the mobile number.

3.2.4 Software Quality Attributes

The software is very adaptable and weasy to use because of its simple design. The software is available easily on google play store. It is maintained regularly from our side. The products is very robust and is tested through various phases.

System Features

This app possess a lot of features for the users and their health requirements. This app enables the ambulance to reach at an emergency service as fast as possible and thus may save many lives. This app not only rescues the accident victims but also the pregnant woman and old people with emergency need.

4.1 System Feature 1

This app eases the process of calling ambulance for an emergency patient in any remote location or any condition and thus helping the victim to reach nearest hospital in no time.

4.2 System Feature 2

This app can used in a huge scale and thus can enable its reach to help people to get faster ambulance services to save more lives. The app is completely reliable for the victims and the trespassers and the data is completely secured.

4.3 System Feature 3

The app also track the ambulance with GPS system which helps the victim to locate the ambulance and also can assess the ambulance and its driver. Thus the victims can feel secured and lives could be saved without problems.

.

SYSTEM DESIGN

System design is to organize the software modules in such a way that are easy to develop and change. Structured design techniques helped us to deal with the size and complexity of programs. System design plays a very important role in developing a software. If any pre-existing code needs to be understood, organized and pieced together system design will help us in it.

We have used the Bottom up approach in this Project.

Bottom-up approach:

The design starts with the lowest level components and subsystems. By using these components, the next immediate higher level components and subsystems are created or composed. The process is continued till all the components and subsystems are composed into a single component, which is considered as the complete system.

To be precise we used RAD model to build this app. RAD model distributes the analysis, design, build and test phases into a series of short, iterative development cycles. It focuses on input-output source and destination of the information. It emphasizes on delivering projects in small pieces; the larger projects are divided into a series of smaller projects. The main features of RAD model are that it focuses on the reuse of templates, tools, processes, and code. RAD model distributes the analysis, design, build and test phases into a series of short, iterative development cycles.

The four phases of RAD model are-

Business Modeling:

The information flow among business functions is defined by answering questions like what data drives the business process, what data is generated, who generates it, where does the information go, who process it and so on.

Data Modeling

The data collected from business modeling is refined into a set of data objects (entities) that are needed to support the business. The attributes (character of each entity) are identified, and the relation between these data objects (entities) is defined.

Process Modeling

The information object defined in the data modeling phase are transformed to achieve the data flow necessary to implement a business function. Processing descriptions are created for adding, modifying, deleting, or retrieving a data object.

Application Generation

The actual system is built and coding is done by using automation tools to convert process and data models into actual prototypes.

Using RAD Model out of all SDLC models have been quite useful for us in following ways-

- Changing requirements can be accommodated.
- Progress can be measured.
- Iteration time can be short with use of powerful RAD tools.
- Productivity with fewer people in a short time.
- Reduced development time.
- Increases reusability of components.
- Quick initial reviews occur.
- Encourages customer feedback.
- Integration from very beginning solves a lot of integration issues.

PROJECT PLANNING

Project planning is an organized and integrated management process, in which we focused on activities required for successful completion of the project. It prevented the obstacles that arise in the project such as changes in projects or project objectives, non-availability of resources, and so on. Project planning also helps in better utilization of resources and optimal usage of the allotted time for the project.

SCOPE MANAGEMENT:

In scope management we define the scope of project; this includes all the activities, process need to be done in order to make the software deliverable. Scope management is essential because it creates boundaries of the project by clearly defining what would be done in the project and what would not be done. This made our project to contain limited and quantifiable tasks, which can be easily documented and in turn avoids cost and time overrun.

PROJECT ESTIMATION

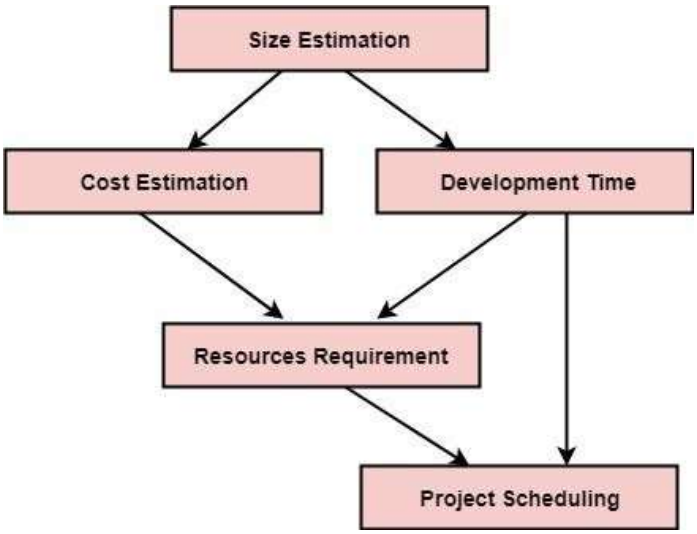
For effective management of the project accurate estimation of various measures is a must. With correct estimation we could manage and control the project more efficiently and effectively.

Software Size Estimation- Software size is estimated either in terms of KLOC (Kilo Line of Code) or by calculating number of function points in software.

Effort Estimation- We estimated efforts in terms of personnel requirement and man-hour required to produce the software.

Time Estimation- Once size and effort is estimated time estimation become very easy task. Software tasks were divided into smaller tasks, activities or events by Work Breakthrough Structure (WBS).

Cost Estimation- We haven’t spend money for this project, but for industrial purposes cost estimation is very essential. It mainly depends on hardware, size of software, quality of software, additional tools etc.



Project Scheduling

Project Scheduling in a project refers to roadmap of all activities to be done with specified order and within time slot allotted to each activity. We defined various tasks, and project milestones and arrange them keeping various factors in mind. We looked for tasks lie in critical path in the schedule, which were necessary to complete in specific manner (because of task interdependency) and strictly within the time allocated.

Resource management

All elements used to develop a software product maybe assumed as a resource for the project. The resources were available in limited quantity and stayed in the group as a pool of assets. The shortage of resources hampers the development of project and it can lag behind the schedule. Allocating extra resources increases development cost in the end.

Resource management includes creating proper project team and allocate specific task to each member and also determine the resources required at a certain stage and its availability.

Code

Ambulances.java-

```
package com.jack2002.miniproject2;

import java.util.Map;

public class Ambulances {
private String inputdname,inputdphone,inputdemail,inputdpassword,inputaid,inputadesc;

public Ambulances() {
}

        public Ambulances(String inputdname, String inputdphone, String inputdemail, String
inputdpassword, String inputaid, String inputadesc) {
this.inputdname = inputdname; this.inputdphone = inputdphone; this.inputdemail = inputdemail;
this.inputdpassword = inputdpassword;this.inputaid = inputaid;



this.inputadesc = inputadesc;
}

public String getInputdname() {return inputdname;
}

public void setInputdname(String inputdname) {this.inputdname = inputdname;
}

public String getInputdphone() {return inputdphone;
}

public void setInputdphone(String inputdphone) {this.inputdphone = inputdphone;
}

public String getInputdemail() {return inputdemail;
}

public void setInputdemail(String inputdemail) {this.inputdemail = inputdemail;
}
```

```
public String getInputdpassword() {return inputdpassword;
   }

public void setInputdpassword(String inputdpassword) {this.inputdpassword = inputdpassword;
   }

public String getInputaid() {return inputaid;
   }


   this.inputaid = inputaid;
   }

public String getInputadesc() {return inputadesc;
   }

public void setInputadesc(String inputadesc) {this.inputadesc = inputadesc;
   }
   }
```

DriverLoginActivity.java-

```
   package com.jack2002.miniproject2;

   import androidx.annotation.NonNull;
   import androidx.appcompat.app.AppCompatActivity;import androidx.core.app.ActivityCompat;
   import androidx.core.content.ContextCompat;

   import  android.Manifest; import android.app.AlertDialog;
   import android.app.ProgressDialog; import android.content.DialogInterface;import android.content.Intent;
   import android.content.IntentSender;
   import android.content.pm.PackageManager;import android.os.Build;
   import android.os.Bundle; import android.view.View; import android.widget.Button;import android.widget.Toast;

   import com.google.android.gms.common.api.ApiException;
   import com.google.android.gms.common.api.ResolvableApiException;import
   com.google.android.gms.location.LocationRequest;
   import com.google.android.gms.location.LocationServices;
```

```
import com.google.android.gms.location.LocationSettingsRequest; import
com.google.android.gms.location.LocationSettingsResponse; import
com.google.android.gms.location.LocationSettingsStatusCodes;import
com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.android.material.textfield.TextInputLayout;import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

public class DriverLoginActivity extends AppCompatActivity {

TextInputLayout inputemail,inputpassword;Button btnlogin;

private ProgressDialog progressDialog;

FirebaseAuth mfirebaseauth;

private static final int Request_User_Location_Code=99;

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

GPSLocationPermission();
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
checkUserLocationPermission();
}


inputemail=findViewById(R.id.inputemail); inputpassword=findViewById(R.id.inputpassword);
btnlogin=findViewById(R.id.btnlogin);

mfirebaseauth = FirebaseAuth.getInstance();

btnlogin.setOnClickListener(new View.OnClickListener() {@Override
public void onClick(View view) {
```

```
String email=inputemail.getEditText().getText().toString().trim();
String password=inputpassword.getEditText().getText().toString().trim(); if(email.isEmpty())
{
inputemail.setError("Please enter email id");inputemail.requestFocus();
}
else if(password.isEmpty())
{
inputpassword.setError("Please enter password");inputpassword.requestFocus();
}
else if(email.isEmpty() && password.isEmpty())
{
Toast.makeText(DriverLoginActivity.this, "Login failed ..... please fill up the details",
Toast.LENGTH_LONG).show();
}
else if(!(email.isEmpty() && password.isEmpty()))
{
progressDialog=new ProgressDialog(DriverLoginActivity.this);progressDialog.setMessage("Logging in....please
wait..."); progressDialog.show();


mfirebaseauth.signInWithEmailAndPassword(email,password).addOnCompleteListener(DriverLoginActi
vity.this, new OnCompleteListener<AuthResult>() {
@Override
public void onComplete(@NonNull Task<AuthResult> task) {if(!task.isSuccessful())
{
progressDialog.dismiss();
Toast.makeText(DriverLoginActivity.this, "Login Unsuccessful.....please try
again later", Toast.LENGTH_LONG).show();
}
else
{
progressDialog.dismiss();
startActivity(new Intent(DriverLoginActivity.this,DriverMapActivity.class));
}
```

```
                                                                                    }
                                                                                });
        }
        else
        {
        progressDialog.dismiss();
        Toast.makeText(DriverLoginActivity.this, "Error Occured ..... please try again later",
        Toast.LENGTH_LONG).show();
        }

                                                                                    }
                                                                                });
        }


        // RunTime permission for Google Location Servicesprivate void GPSLocationPermission()
        {
        LocationRequest mLocationRequest = LocationRequest.create()
        .setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY)
        .setInterval(10 * 1000)
        .setFastestInterval(1 * 1000);

        LocationSettingsRequest.Builder settingsBuilder = new LocationSettingsRequest.Builder()
                .addLocationRequest(mLocationRequest);settingsBuilder.setAlwaysShow(true);

        Task<LocationSettingsResponse> result = LocationServices.getSettingsClient(this)
        .checkLocationSettings(settingsBuilder.build());

    result.addOnCompleteListener(new OnCompleteListener<LocationSettingsResponse>() {@Override
    public void onComplete(@NonNull Task<LocationSettingsResponse> task) {try {
LocationSettingsResponse response = task.getResult(ApiException.class);
    } catch (ApiException ex) { switch (ex.getStatusCode()) {
    case LocationSettingsStatusCodes.RESOLUTION_REQUIRED:try {
```

```
                    ResolvableApiException resolvableApiException =(ResolvableApiException) ex;
                        resolvableApiException
                                .startResolutionForResult(DriverLoginActivity.this,Request_User_Locat

ion_Code                        } catch (IntentSender.SendIntentException e) {
);
                        }
                        break;
                        case LocationSettingsStatusCodes.SETTINGS_CHANGE_UNAVAILABLE:

                        break;

                    }
                }
                    }
                });


    }

    public boolean checkUserLocationPermission()
    {
    GPSLocationPermission();
                if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
    != PackageManager.PERMISSION_GRANTED &&ContextCompat.checkSelfPermission(this,
    Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {

    // Permission is not granted
    // Should we show an explanation?

if (ActivityCompat.shouldShowRequestPermissionRationale(DriverLoginActivity.this,
    Manifest.permission.ACCESS_FINE_LOCATION)) {
    // Show an explanation to the user *asynchronously* -- don't block
    // this thread waiting for the user's response! After the user
    // sees the explanation, try again to request the permission.

    new AlertDialog.Builder(this)
                        .setTitle("Required Location Permission").setMessage("You have to give thispermission
    to access this feature")
    .setPositiveButton("OK", new DialogInterface.OnClickListener() {
```

```
    @Override
public void onClick(DialogInterface dialog, int which) { ActivityCompat.requestPermissions(DriverLoginActivity.this,
    new String[]{Manifest.permission.ACCESS_FINE_LOCATION},Request_User_Location_Code);
    }
    })
.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {@Override
public void onClick(DialogInterface dialog, int which) {dialog.dismiss();
    }
    })


    } else {
    .create()
    .show();

    // No explanation needed; request the permission ActivityCompat.requestPermissions(DriverLoginActivity.this,
    new String[]{Manifest.permission.ACCESS_FINE_LOCATION},Request_User_Location_Code);
    }
    return false;
    }
    else
    {
    return true;
    }
    }
    }
```

HospitalLoginActivity.java-

```
    package com.jack2002.miniproject2;

    import androidx.annotation.NonNull;
```

```
import androidx.appcompat.app.AppCompatActivity;import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import  android.Manifest; import android.app.AlertDialog;
import android.app.ProgressDialog; import android.content.DialogInterface;import android.content.Intent;
import android.content.IntentSender;
import android.content.pm.PackageManager;import android.os.Build;
import android.os.Bundle; import android.view.View; import android.widget.Button; import
android.widget.TextView;import android.widget.Toast;

import com.google.android.gms.common.api.ApiException;
import com.google.android.gms.common.api.ResolvableApiException;import
com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.location.LocationSettingsRequest; import
com.google.android.gms.location.LocationSettingsResponse; import
com.google.android.gms.location.LocationSettingsStatusCodes;import
com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.android.material.textfield.TextInputLayout;import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

public class HospitalLoginActivity extends AppCompatActivity {

TextInputLayout inputemail,inputpassword;Button btnlogin;
TextView tvregister,tvforgotpassword,tvdriverlogin;

private ProgressDialog progressDialog;

FirebaseAuth mfirebaseauth;
```

```
    private long backPressedTime;

    private static final int Request_User_Location_Code=99;

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

    GPSLocationPermission();
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
    checkUserLocationPermission();
    }

    inputemail=findViewById(R.id.inputemail); inputpassword=findViewById(R.id.inputpassword);
    btnlogin=findViewById(R.id.btnlogin); tvregister=findViewById(R.id.tvregister);
    tvforgotpassword=findViewById(R.id.tvforgotpassword);tvdriverlogin=findViewById(R.id.tvdriverlogin);

    mfirebaseauth = FirebaseAuth.getInstance();

btnlogin.setOnClickListener(new View.OnClickListener() {@Override
    public void onClick(View v) {
    String email=inputemail.getEditText().getText().toString().trim();
    String password=inputpassword.getEditText().getText().toString().trim(); if(email.isEmpty())
    {
    inputemail.setError("Please enter email id");inputemail.requestFocus();
    }
    else if(password.isEmpty())
    {
    inputpassword.setError("Please enter password");inputpassword.requestFocus();
    }
    else if(email.isEmpty() && password.isEmpty())
```

```
    {
    Toast.makeText(HospitalLoginActivity.this, "Login failed ..... please fill up the details",
    Toast.LENGTH_LONG).show();
    }
    else if(!(email.isEmpty() && password.isEmpty()))
    {
    progressDialog=new ProgressDialog(HospitalLoginActivity.this); progressDialog.setTitle("Hospital Login...");
    progressDialog.setMessage("Processing your request...please wait...");progressDialog.show();


    mfirebaseauth.signInWithEmailAndPassword(email,password).addOnCompleteListener(HospitalLoginAc
    tivity.this, new OnCompleteListener<AuthResult>() {
    @Override
public void onComplete(@NonNull Task<AuthResult> task) {if(!task.isSuccessful())
    {
    progressDialog.dismiss();
    Toast.makeText(HospitalLoginActivity.this, "Login Unsuccessful .....please try
    again later", Toast.LENGTH_LONG).show();
    }
    else
    {
    progressDialog.dismiss();
    startActivity(new Intent(HospitalLoginActivity.this,AddAmbulances.class));finish();
    }
                                                                                    }
                                                                                });
    }
    else
    {
    Toast.makeText(HospitalLoginActivity.this, "Error Occured.....please try again later",
    Toast.LENGTH_LONG).show();
    }
                                                                                    }
                                                                                });

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

```
    @Override
    public void onClick(View v) {
    Intent i=new Intent(HospitalLoginActivity.this, HospitalRegisterActivity.class);startActivity(i);
                                                                                                    }
                                                                                                });

tvforgotpassword.setOnClickListener(new View.OnClickListener() {@Override
    public void onClick(View v) {
    Intent intent=new Intent(HospitalLoginActivity.this,Forgot_password_Activity.class);startActivity(intent);
                                                                                                    }
                                                                                                });

tvdriverlogin.setOnClickListener(new View.OnClickListener() {@Override
    public void onClick(View view) {
    Intent i=new Intent(HospitalLoginActivity.this, DriverLoginActivity.class);startActivity(i);
                                                                                                    }
                                                                                                });

    }
    // RunTime permission for Google Location Servicesprivate void GPSLocationPermission()
    {
    LocationRequest mLocationRequest = LocationRequest.create()
    .setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY)
    .setInterval(10 * 1000)
    .setFastestInterval(1 * 1000);

                    LocationSettingsRequest.Builder settingsBuilder = newLocationSettingsRequest.Builder()
    .addLocationRequest(mLocationRequest);
```

```
        settingsBuilder.setAlwaysShow(true);

        Task<LocationSettingsResponse> result = LocationServices.getSettingsClient(this)
        .checkLocationSettings(settingsBuilder.build());

                                result.addOnCompleteListener(new OnCompleteListener<LocationSettingsResponse>()
        {
        @Override
    public void onComplete(@NonNull Task<LocationSettingsResponse> task) {try {
LocationSettingsResponse response = task.getResult(ApiException.class);
    } catch (ApiException ex) { switch (ex.getStatusCode()) {
    case  LocationSettingsStatusCodes.RESOLUTION_REQUIRED: try {
ResolvableApiException resolvableApiException =(ResolvableApiException) ex;
        resolvableApiException
                                                        .startResolutionForResult(HospitalLoginActivity.this,Request_User_
        Location_Code);
                                    } catch (IntentSender.SendIntentException e) {

                                    }
                                    break;
                                    case     LocationSettingsStatusCodes.SETTINGS_CHANGE_UNAVAILABLE:

                                    break;
                                }
                                }
                                }
                            });


        }

        public boolean checkUserLocationPermission()
        {
        GPSLocationPermission();
                        if (ContextCompat.checkSelfPermission(this,
        Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED &&
```

```
        ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION) !=
        PackageManager.PERMISSION_GRANTED) {

        // Permission is not granted
        // Should we show an explanation?

if (ActivityCompat.shouldShowRequestPermissionRationale(HospitalLoginActivity.this,
        Manifest.permission.ACCESS_FINE_LOCATION)) {
        // Show an explanation to the user *asynchronously* -- don't block
        // this thread waiting for the user's response! After the user
        // sees the explanation, try again to request the permission.

        new AlertDialog.Builder(this)
                                        .setTitle("Required Location Permission").setMessage("You have to give this
        permission to access this feature")
    .setPositiveButton("OK", new DialogInterface.OnClickListener() {@Override
    public void onClick(DialogInterface dialog, int which) {
        ActivityCompat.requestPermissions(HospitalLoginActivity.this,
        new String[]{Manifest.permission.ACCESS_FINE_LOCATION},Request_User_Location_Code);
        }
        })
    .setNegativeButton("Cancel", new DialogInterface.OnClickListener() {@Override
    public void onClick(DialogInterface dialog, int which) {dialog.dismiss();
        }
        })
                                        .create()
                                        .show();
                                } else {

        // No explanation needed; request the permission
        ActivityCompat.requestPermissions(HospitalLoginActivity.this,
        new String[]{Manifest.permission.ACCESS_FINE_LOCATION},Request_User_Location_Code);
        }
        return false;
        }
        else
```

```
    {
    return true;
    }
    }

    @Override
protected void onStart() {super.onStart();

    }

    @Override
    public void onBackPressed() {

    if(backPressedTime + 2000 > System.currentTimeMillis())
    {
    Intent a = new Intent(Intent.ACTION_MAIN); a.addCategory(Intent.CATEGORY_HOME);
    a.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);startActivity(a);
    }
    else
    {
                    Toast.makeText(this, "Press back again to exit", Toast.LENGTH_SHORT).show();
    }
    backPressedTime=System.currentTimeMillis();
    }
    }
```

HospitalProfileActivity.java-

```
    package com.jack2002.miniproject2;
    import androidx.annotation.NonNull;
    import androidx.appcompat.app.AppCompatActivity;

    import android.content.Intent;import android.os.Bundle;
```

```
import android.text.method.ScrollingMovementMethod;import android.widget.TextView;
import android.widget.Toast;

import com.google.firebase.auth.FirebaseAuth; import com.google.firebase.database.DataSnapshot;import
com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;import
com.google.firebase.database.FirebaseDatabase; import com.google.firebase.database.ValueEventListener;

public class HospitalProfileActivity extends AppCompatActivity {

private TextView textname,textemail,textpin,textaddress;private FirebaseAuth firebaseAuth;
private DatabaseReference databaseReference;

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

textname=findViewById(R.id.textname); textemail=findViewById(R.id.textemail);
textpin=findViewById(R.id.textpin); textaddress=findViewById(R.id.textaddress);

textemail.setMovementMethod(new ScrollingMovementMethod()); textaddress.setMovementMethod(new
ScrollingMovementMethod());

firebaseAuth=FirebaseAuth.getInstance();databaseReference=
FirebaseDatabase.getInstance().getReference().child("ADMIN").child("HOSPITALS").child(firebaseAuth.
getCurrentUser().getUid());

databaseReference.addValueEventListener(new ValueEventListener() {@Override
public void onDataChange(@NonNull DataSnapshot dataSnapshot) {Hospital
h=dataSnapshot.getValue(Hospital.class); textname.setText(h.getName());



textemail.setText(h.getEmail()); textaddress.setText(h.getAddress());textpin.setText(h.getPin());

}

@Override
public void onCancelled(@NonNull DatabaseError databaseError) {
Toast.makeText(HospitalProfileActivity.this,databaseError.getCode(),
Toast.LENGTH_SHORT).show();
```

```
                                                                                    }
                                                                                });
}
@Override
public void onBackPressed() {
startActivity(new Intent(HospitalProfileActivity.this,AddAmbulances.class));finish();
}
}
```

HospitalRegisterActivity.java-

```
package com.jack2002.miniproject2;

import androidx.annotation.NonNull; import androidx.annotation.RequiresApi;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.app.ProgressDialog;
import android.content.DialogInterface;import android.content.Intent;
import android.content.IntentSender;
import android.content.pm.PackageManager;import android.location.Address;
import android.location.Geocoder;import android.location.Location; import android.os.Build;
import android.os.Bundle; import android.util.Log; import android.view.View; import android.view.Window;
import android.view.WindowManager;import android.widget.Button;
import android.widget.Toast;

import com.google.android.gms.common.api.ApiException;
import com.google.android.gms.common.api.ResolvableApiException; import
com.google.android.gms.location.FusedLocationProviderClient; import
com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.location.LocationSettingsRequest; import
com.google.android.gms.location.LocationSettingsResponse; import
com.google.android.gms.location.LocationSettingsStatusCodes;import
com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.OnSuccessListener;import com.google.android.gms.tasks.Task;
```

```
import com.google.android.material.textfield.TextInputEditText;
import com.google.android.material.textfield.TextInputLayout; import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseAuthException; import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError; import
com.google.firebase.database.DatabaseReference;import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import java.io.IOException;import java.util.List;
import java.util.Locale;

public class HospitalRegisterActivity extends AppCompatActivity {

private TextInputLayout inputname, inputemail, inputpassword;private TextInputEditText inputaddress,inputpin;
private Button btnsubmit; Hospital hospital; DatabaseReference d,hcount;FirebaseAuth mfirebaseauth;
private ProgressDialog progressDialog;Geocoder geocoder;
List<Address> addresses;String loc="";
Double lat,lon;

private static final int MY_PERMISSIONS_REQUEST_READ_LOCATION=99;private
FusedLocationProviderClient fusedLocationProviderClient;


@RequiresApi(api = Build.VERSION_CODES.M)@Override
protected void onCreate(Bundle savedInstanceState) {super.onCreate(savedInstanceState);
setContentView(R.layout.activity_hospital_register);

if (Build.VERSION.SDK_INT >= 21) {
Window window = this.getWindow();

window.addFlags(WindowManager.LayoutParams.FLAG_DRAWS_SYSTEM_BAR_BACKGROUNDS);
window.clearFlags(WindowManager.LayoutParams.FLAG_TRANSLUCENT_STATUS);
window.setStatusBarColor(this.getResources().getColor(R.color.colorPrimary));
}

inputname = findViewById(R.id.inputname); inputemail = findViewById(R.id.inputemail); inputpassword =
findViewById(R.id.inputpassword);inputpin =  findViewById(R.id.inputpin); inputaddress =
findViewById(R.id.address); btnsubmit = findViewById(R.id.btnsubmit); geocoder=new
Geocoder(this,Locale.getDefault());
```

```
    mfirebaseauth = FirebaseAuth.getInstance();
    d = FirebaseDatabase.getInstance().getReference();hospital=new Hospital();

    fusedLocationProviderClient= LocationServices.getFusedLocationProviderClient(this);

    fetchLocation();

btnsubmit.setOnClickListener(new View.OnClickListener() {@Override
    public void onClick(View v) {
if (inputname.getEditText().getText().toString().trim().length() == 0)inputname.setError("Hospital Name is
    required");
else if (inputemail.getEditText().getText().toString().trim().length() == 0)inputemail.setError("Hospital Email is
    required");
else if (inputpassword.getEditText().getText().toString().trim().length() == 0)inputpassword.setError("Password is
    required");
else if (inputaddress.getText().toString().trim().length() == 0)inputaddress.setError("Hospital Address is
    required");
else if (inputpin.getText().toString().trim().length() == 0) inputpin.setError("Pin Code of your area is required");
    else {
    CreateUserAndSaveData();
    }
                                                                                                    }
                                                                                                });
    }



    private void CreateUserAndSaveData() {

    progressDialog=new ProgressDialog(this); progressDialog.setTitle("Hospital Registration...");
    progressDialog.setMessage("Processing your request...please wait...");progressDialog.show();


    mfirebaseauth.createUserWithEmailAndPassword(inputemail.getEditText().getText().toString(),inputpas
```

```
sword.getEditText().getText().toString()).addOnCompleteListener(this, new OnCompleteListener<AuthResult>()
{
@Override
public void onComplete(@NonNull Task<AuthResult> task) {if (task.isSuccessful())
{
progressDialog.dismiss();saveData();
                Toast.makeText(HospitalRegisterActivity.this, "Hospital Successfully Registered...",
Toast.LENGTH_SHORT).show();
}
else
{
progressDialog.dismiss();
FirebaseAuthException e = (FirebaseAuthException)task.getException();Log.i("ERROR",e.getMessage());
Toast.makeText(HospitalRegisterActivity.this, "Hospital Registration Failed .....Please try
again later", Toast.LENGTH_SHORT).show();
}
                                                                                                    }
                                                                                                  });
}

private void saveData()
{
hospital.setName(inputname.getEditText().getText().toString().trim());
hospital.setEmail(inputemail.getEditText().getText().toString().trim());
hospital.setPassword(inputpassword.getEditText().getText().toString().trim());
hospital.setAddress(inputaddress.getText().toString().trim());
hospital.setPin(inputpin.getText().toString().trim());

hospital.setLatitude(lat); hospital.setLongitude(lon);

    d=FirebaseDatabase.getInstance().getReference().child("ADMIN").child("HOSPITALS").child(mfirebaseaut
h.getCurrentUser().getUid());

d.setValue(hospital);
```

```
        FirebaseAuth.getInstance().signOut();
        startActivity(new Intent(HospitalRegisterActivity.this, HospitalLoginActivity.class));finish();

        }
        // Runtime permission for the location on the appprivate void fetchLocation()
        {
        GPSLocationPermission();
if (ContextCompat.checkSelfPermission(HospitalRegisterActivity.this,
        Manifest.permission.ACCESS_FINE_LOCATION)
        != PackageManager.PERMISSION_GRANTED) {

        // Permission is not granted
        // Should we show an explanation?

        if (ActivityCompat.shouldShowRequestPermissionRationale(HospitalRegisterActivity.this
        ,
        Manifest.permission.ACCESS_FINE_LOCATION)) {
        // Show an explanation to the user *asynchronously* -- don't block
        // this thread waiting for the user's response! After the user
        // sees the explanation, try again to request the permission.

        new AlertDialog.Builder(this)
                        .setTitle("Required Location Permission").setMessage("You have to give this permissionto
        access this feature")
    .setPositiveButton("OK", new DialogInterface.OnClickListener() {@Override
    public void onClick(DialogInterface dialog, int which) {
        ActivityCompat.requestPermissions(HospitalRegisterActivity.this,
        new String[]{Manifest.permission.ACCESS_FINE_LOCATION},
        MY_PERMISSIONS_REQUEST_READ_LOCATION);
        }
```

```
    })
.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {@Override
public void onClick(DialogInterface dialog, int which) {dialog.dismiss();
    }
    })
                        } else {
                        .create()
                        .show();
    // No explanation needed; request the permission
    ActivityCompat.requestPermissions(HospitalRegisterActivity.this,
    new String[]{Manifest.permission.ACCESS_FINE_LOCATION},
    MY_PERMISSIONS_REQUEST_READ_LOCATION);

    // MY_PERMISSIONS_REQUEST_READ_CONTACTS is an
    // app-defined int constant. The callback method gets the
    // result of the request.
    }
    } else {
    // Permission has already been granted fusedLocationProviderClient.getLastLocation()
.addOnSuccessListener(this, new OnSuccessListener<Location>() {@Override
    public void onSuccess(Location location) {
    // Got last known location. In some rare situations this can be null.if (location != null) {
    // Logic to handle location objectlat = location.getLatitude();
    lon = location.getLongitude();try {
    addresses=geocoder.getFromLocation(lat,lon,1); String address=addresses.get(0).getAddressLine(0);String
    Area=addresses.get(0).getLocality();
    String City=addresses.get(0).getAdminArea(); String post=addresses.get(0).getPostalCode();
    String fulladdress=address+" , "+Area+" , "+City+" , "+post;inputaddress.setText(fulladdress);
    inputpin.setText(post);
    } catch (IOException e) {
```

```
    e.printStackTrace();
    }

    }
    }
    });
    }
    }

    @Override
        public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNullint[]
    grantResults) {
    if (requestCode == MY_PERMISSIONS_REQUEST_READ_LOCATION) {
if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {if
    (ActivityCompat.checkSelfPermission(this,
    Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED &&
    ActivityCompat.checkSelfPermission(this, 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;
    }
    fusedLocationProviderClient.getLastLocation()
.addOnSuccessListener(this, new OnSuccessListener<Location>() {@Override
    public void onSuccess(Location location) {
    // Got last known location. In some rare situations this can be null.if (location != null) {
    // Logic to handle location object Double lat = location.getLatitude(); Double lon = location.getLongitude();

    //loc="Latitude :-" + lat + "\n" + "Longitude :-" + lon;

    //tvlat.setText(String.valueOf(lat));
    //tvlon.setText(String.valueOf(lon));
    }
    }
    });
    }
    else
```

```
    {
    new AlertDialog.Builder(this)
                    .setTitle("Required Location Permission").setMessage("You have to give this permissionto
    access this feature")
  .setPositiveButton("OK", new DialogInterface.OnClickListener() {@Override
  public void onClick(DialogInterface dialog, int which) {
    ActivityCompat.requestPermissions(HospitalRegisterActivity.this,
    new String[]{Manifest.permission.ACCESS_FINE_LOCATION},
    MY_PERMISSIONS_REQUEST_READ_LOCATION);
    }
    })
  .setNegativeButton("Cancel", new DialogInterface.OnClickListener() {@Override
  public void onClick(DialogInterface dialog, int which) {dialog.dismiss();
    }
    })
    .create()
    .show();
    }
    }
    }
    // RunTime permission for Google Location Servicesprivate void GPSLocationPermission()
    {
    LocationRequest mLocationRequest = LocationRequest.create()
    .setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY)
    .setInterval(10 * 1000)
    .setFastestInterval(1 * 1000);

    LocationSettingsRequest.Builder settingsBuilder = new LocationSettingsRequest.Builder()
          .addLocationRequest(mLocationRequest);settingsBuilder.setAlwaysShow(true);


    Task<LocationSettingsResponse> result = LocationServices.getSettingsClient(this)
    .checkLocationSettings(settingsBuilder.build());

  result.addOnCompleteListener(new OnCompleteListener<LocationSettingsResponse>() {@Override
  public void onComplete(@NonNull Task<LocationSettingsResponse> task) {try {
LocationSettingsResponse response = task.getResult(ApiException.class);
  } catch (ApiException ex) { switch (ex.getStatusCode()) {
  case LocationSettingsStatusCodes.RESOLUTION_REQUIRED:try {
ResolvableApiException resolvableApiException =(ResolvableApiException) ex;
    resolvableApiException
```

```
startResolutionForResult(HospitalRegisterActivity.this,MY_PERMISSIONS_R EQUEST_READ_LOCATION);
} catch (IntentSender.SendIntentException e) {

}
break;
case LocationSettingsStatusCodes.SETTINGS_CHANGE_UNAVAILABLE:

break;
}
}
}
});

}
}
```

RegisterAmbulance.java-

```
package com.jack2002.miniproject2;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.app.ProgressDialog;import android.content.Intent; 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.tasks.OnCompleteListener;import com.google.android.gms.tasks.Task;
import com.google.android.material.textfield.TextInputLayout;import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseAuthException; import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError; import
com.google.firebase.database.DatabaseReference;import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;

public class RegisterAmbulance extends AppCompatActivity {

private TextInputLayout inputdname,inputdphone,inputdemail,inputdpassword,inputaid,inputadesc; private
Button btnsubmit;

FirebaseAuth firebaseAuth; DatabaseReference ref,hospital_ref;

Ambulances ambulances;
```

```
    private ProgressDialog progressDialog;private String hospitalID,h_key;

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

    inputdname=findViewById(R.id.inputdname); inputdphone=findViewById(R.id.inputdphn);
    inputdemail=findViewById(R.id.inputdemail); inputdpassword=findViewById(R.id.inputdpassword);
    inputaid=findViewById(R.id.inputaid); inputadesc=findViewById(R.id.inputadesc);

    btnsubmit=findViewById(R.id.btnsubmit);

    firebaseAuth=FirebaseAuth.getInstance(); ambulances=new Ambulances();
    hospitalID=firebaseAuth.getCurrentUser().getUid();

btnsubmit.setOnClickListener(new View.OnClickListener() {@Override
    public void onClick(View v) {
    String dname=inputdname.getEditText().getText().toString().trim(); String
    dphone=inputdphone.getEditText().getText().toString().trim(); String
    demail=inputdemail.getEditText().getText().toString().trim();
    String dpassword=inputdpassword.getEditText().getText().toString().trim(); String
    aid=inputaid.getEditText().getText().toString().trim();
    String adesc=inputadesc.getEditText().getText().toString().trim();

if (dname.length() == 0) inputdname.setError("Driver Name is required");
else if (dphone.length() == 0) inputdphone.setError("Driver phone no. is required");
else if (demail.length() == 0) inputdphone.setError("Driver Email-ID is required");
else if (dpassword.length() == 0) inputdphone.setError("Driver password is required");
```

```
else if (aid.length() == 0) inputaid.setError("Ambulance no. is required");
else if (adesc.length() == 0) inputadesc.setError("Ambulance Description is required");
    else
    {
    progressDialog=new ProgressDialog(RegisterAmbulance.this); progressDialog.setTitle("Ambulance
    Registration..."); progressDialog.setMessage("Adding driver details...please wait...");progressDialog.show();
    firebaseAuth.createUserWithEmailAndPassword(demail,
    dpassword).addOnCompleteListener(RegisterAmbulance.this, new OnCompleteListener<AuthResult>()
    {
    @Override
public void onComplete(@NonNull Task<AuthResult> task) {if (task.isSuccessful()) {
    SaveDriverDetails(dname,dphone,demail,dpassword,aid,adesc); Toast.makeText(RegisterAmbulance.this,
    "Driver Successfully Registered...",
    Toast.LENGTH_SHORT).show();
    } else {
    progressDialog.dismiss();
    FirebaseAuthException e = (FirebaseAuthException) task.getException();Log.i("ERROR", e.getMessage());
    Toast.makeText(RegisterAmbulance.this, "Driver Registration Failed .....Please try
    again later", Toast.LENGTH_SHORT).show();
    }
                                                                                    }
                                                                                });
    }
                                                                                    }
                                                                                });
    }

        private void SaveDriverDetails(String dname, String dphone,String demail, String dpassword, Stringaid,
    String adesc)
    {
    ambulances.setInputdname(dname); ambulances.setInputdphone(dphone);
    ambulances.setInputdemail(demail); ambulances.setInputdpassword(dpassword);
```

```
ambulances.setInputaid(aid); ambulances.setInputadesc(adesc);

if(firebaseAuth.getCurrentUser() != null)
{
progressDialog.dismiss();ref=
FirebaseDatabase.getInstance().getReference().child("ADMIN").child("HOSPITALS").child(hospitalID).c
hild("AMBULANCES").child(firebaseAuth.getCurrentUser().getUid());

ref.setValue(ambulances);


h_key=FirebaseDatabase.getInstance().getReference().child("ADMIN").child("HOSPITALS").child(hospit
alID).getKey();


        hospital_ref=FirebaseDatabase.getInstance().getReference().child("DRIVER-
HOSPITAL").child(firebaseAuth.getCurrentUser().getUid());
hospital_ref.setValue(h_key);

Toast.makeText(this, "Driver added Successfully...", Toast.LENGTH_SHORT).show();Intent intent=new
Intent(RegisterAmbulance.this,AddAmbulances.class); startActivity(intent);
finish();
}
else
{
progressDialog.dismiss();
Toast.makeText(this, "Hospital not signed in...please sign in..", Toast.LENGTH_SHORT).show();
startActivity(new Intent(RegisterAmbulance.this, HospitalLoginActivity.class));
finish();
}
}

@Override
public void onBackPressed() {
startActivity(new Intent(RegisterAmbulance.this,AddAmbulances.class));finish();
}
```

}

Drivermapactivity-

```
package com.jack2002.miniproject2;

import androidx.annotation.NonNull; import androidx.annotation.Nullable; import
androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat; import androidx.fragment.app.FragmentActivity;

import  android.Manifest; import android.app.AlertDialog;import android.app.Application;
import android.content.DialogInterface;import android.content.Intent;
import android.content.IntentSender;
import android.content.pm.PackageManager;import android.location.Location;
import android.os.Build; import android.os.Bundle; import android.os.Handler; import android.util.Log; import
android.view.View; import android.widget.Button;import android.widget.Toast;

import com.firebase.geofire.GeoFire; import com.firebase.geofire.GeoLocation;
import com.google.android.gms.common.ConnectionResult; import
com.google.android.gms.common.api.ApiException; import
com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.common.api.ResolvableApiException;import
com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.location.LocationSettingsRequest; import
com.google.android.gms.location.LocationSettingsResponse;
```

```
import com.google.android.gms.location.LocationSettingsStatusCodes;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.BitmapDescriptorFactory;import
com.google.android.gms.maps.model.CameraPosition;
import com.google.android.gms.maps.model.LatLng;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 com.google.firebase.auth.FirebaseAuth; import com.google.firebase.auth.FirebaseUser; import
com.google.firebase.database.DataSnapshot;import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;import
com.google.firebase.database.FirebaseDatabase; import com.google.firebase.database.ValueEventListener;

import java.sql.Driver;import java.util.List; import java.util.Timer;
import java.util.TimerTask;

import javax.net.ssl.SNIHostName;

public class DriverMapActivity extends FragmentActivity implements OnMapReadyCallback,
GoogleApiClient.ConnectionCallbacks, GoogleApiClient.OnConnectionFailedListener,
com.google.android.gms.location.LocationListener {

private GoogleMap mMap;
private GoogleApiClient googleApiClient; private LocationRequest locationRequest;private Location
lastLocation;
private Marker currentlocationmarker,PickUpMarker;

private static final int Request_User_Location_Code=99;

private Button btnlogout,btnsettings;
```

```
    private FirebaseAuth mAuth; private FirebaseUser currentUser;
    private String driverID,customerID = "";
    private DatabaseReference AssignedCustomerRef,AssignedCustomerPickupRef;private boolean
    currentLogoutDriverStatus = false;

    private ValueEventListener AssignedCustomerPickupRefListener;

    @Override
protected void onCreate(Bundle savedInstanceState) {super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_driver_map);
    // Obtain the SupportMapFragment and get notified when the map is ready to be used. SupportMapFragment
    mapFragment = (SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.map);mapFragment.getMapAsync(this);

    GPSLocationPermission();
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
    checkUserLocationPermission();
    }

    btnlogout = findViewById(R.id.btnlogout); btnsettings = findViewById(R.id.btnsettings);

    mAuth = FirebaseAuth.getInstance(); currentUser = mAuth.getCurrentUser(); driverID =
    mAuth.getCurrentUser().getUid();

btnlogout.setOnClickListener(new View.OnClickListener() {@Override
    public void onClick(View view) {

    currentLogoutDriverStatus = true;DisconnectTheDriver();

    mAuth.signOut();LogOutDriver();
    }
```

```
    });
    GetAssignedCustomerRequest();

btnsettings.setOnClickListener(new View.OnClickListener() {@Override
public void onClick(View view) { Toast.makeText(DriverMapActivity.this, "SETTINGS ACTIVITY",
    Toast.LENGTH_SHORT).show();
    }
    });

    //refresh(5000);

    }

private void refresh(int miliseconds){ final Handler handler=new Handler();
final Runnable runnable=new Runnable() {@Override
    public void run() {


                DatabaseReference rootRef = FirebaseDatabase.getInstance().getReference().child("CUSTOMERS
    REQUESTS").child(driverID);

rootRef.addListenerForSingleValueEvent(new ValueEventListener() {@Override
public void onDataChange(DataSnapshot snapshot) {if (snapshot.exists()) {
    // run some code
    Toast.makeText(DriverMapActivity.this, "Refreshing", Toast.LENGTH_SHORT).show();
    }
    else {
 Toast.makeText(DriverMapActivity.this, "ELSE Refreshing",Toast.LENGTH_SHORT).show();
    }
    }

    @Override
    public void onCancelled(@NonNull DatabaseError databaseError) {
```

```
    }
    });

    refresh(5000);
    }
    };
    handler.postDelayed(runnable,miliseconds);
    }

    private void GetAssignedCustomerRequest() {

        AssignedCustomerRef=FirebaseDatabase.getInstance().getReference().child("DRIVER-
    CUSTOMER").child(driverID).child("CustomerRideID");
AssignedCustomerRef.addValueEventListener(new ValueEventListener() {@Override
public void onDataChange(@NonNull DataSnapshot dataSnapshot) {if(dataSnapshot.exists()){
    customerID=dataSnapshot.getValue().toString();

    GetAssignedCustomerPickupLocation() ;
    }
    else
    {
    customerID="";

    if(PickUpMarker != null)
    {
    PickUpMarker.remove();
    }

                DatabaseReference driveravailableref =
    FirebaseDatabase.getInstance().getReference().child("DRIVERS AVAILABLE");
                DatabaseReference driverworkingref =
    FirebaseDatabase.getInstance().getReference().child("DRIVERS WORKING");

                moveFirebaseRecord(driverworkingref.child(driverID), driveravailableref.child(driverID));

    }
    }
```

```
    @Override
    public void onCancelled(@NonNull DatabaseError databaseError) {

    }l
    });
    }

public void moveFirebaseRecord(DatabaseReference fromPath, final DatabaseReference toPath) {
    fromPath.addListenerForSingleValueEvent(new ValueEventListener() {
    @Override
public void onDataChange(DataSnapshot dataSnapshot) { toPath.setValue(dataSnapshot.getValue(), new
    DatabaseReference.CompletionListener() {
    @Override
                public void onComplete(DatabaseError databaseError, DatabaseReferencedatabaseReference) {
    if (databaseError != null) {
    Log.i("AVAILABLE TO WORKING :", "COPY FAILED");
    } else {
    Log.i("AVAILABLE TO WORKING :", "COPY SUCCESS");

    }
    }
    });
    }

    @Override
public void onCancelled(DatabaseError databaseError) {Log.i("A to W (onCancelled) :", "COPY FAILED");

    }
    });
    }

    private void GetAssignedCustomerPickupLocation() {

    AssignedCustomerPickupRef=FirebaseDatabase.getInstance().getReference().child("CUSTOMERS
    REQUESTS").child(customerID).child("l");
        AssignedCustomerPickupRefListener = AssignedCustomerPickupRef.addValueEventListener(new
    ValueEventListener() {
```

```
    @Override
public void onDataChange(@NonNull DataSnapshot dataSnapshot) {if(dataSnapshot.exists()){
    List<Object> customerLocationMap=( List<Object>) dataSnapshot.getValue();

    double LocationLat= 0;
    double LocationLong= 0;

if(customerLocationMap.get(0) != null){ LocationLat=Double.parseDouble(customerLocationMap.get(0).toString());

    }
if(customerLocationMap.get(1) != null){
    LocationLong=Double.parseDouble(customerLocationMap.get(1).toString());

    }
    LatLng driverLatLng = new LatLng(LocationLat, LocationLong);if (PickUpMarker != null)
    {
    PickUpMarker.remove();
    }
                PickUpMarker = mMap.addMarker(new MarkerOptions().position(driverLatLng).title("Customer
    Pickup Location"));

    mMap.animateCamera(CameraUpdateFactory.newLatLngZoom(PickUpMarker.getPosition(),14));
    }
    }

    @Override
    public void onCancelled(@NonNull DatabaseError databaseError) {

    }
    });
    }


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

```
    if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)==
PackageManager.PERMISSION_GRANTED)
{
buildGoogleApiCLient();

mMap.setMyLocationEnabled(true);
}
}

protected synchronized void buildGoogleApiCLient()
{
googleApiClient=new GoogleApiClient.Builder(this)
.addConnectionCallbacks(this)
.addOnConnectionFailedListener(this)
.addApi(LocationServices.API)
.build();

googleApiClient.connect();
}

@Override
public void onConnected(@Nullable Bundle bundle) {

locationRequest=new LocationRequest();locationRequest.setInterval(1000);
locationRequest.setFastestInterval(1000);
locationRequest.setPriority(LocationRequest.PRIORITY_BALANCED_POWER_ACCURACY);

    if (ContextCompat.checkSelfPermission(this,Manifest.permission.ACCESS_FINE_LOCATION)==Package
Manager.PERMISSION_GRANTED)
{
LocationServices.FusedLocationApi.requestLocationUpdates(googleApiClient,locationRequest,
this);
}

}

@Override
public void onConnectionSuspended(int i) {
```

```
    }

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

    }

    @Override
public void onLocationChanged(Location location) {if(getApplicationContext()!=null) {
    lastLocation = location;

    LatLng latLng = new LatLng(location.getLatitude(), location.getLongitude());

    mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
    mMap.animateCamera(CameraUpdateFactory.zoomBy(14f));

    mMap.animateCamera(CameraUpdateFactory.newCameraPosition(getCameraPositionWithBearing(latL ng)));

if (googleApiClient != null) { LocationServices.FusedLocationApi.removeLocationUpdates(googleApiClient,
    (com.google.android.gms.location.LocationListener) this);
    }

    String driverId = FirebaseAuth.getInstance().getCurrentUser().getUid();

            DatabaseReference DriverAvailibilityRef =
    FirebaseDatabase.getInstance().getReference().child("DRIVERS AVAILABLE");
    GeoFire geoFireAvailability = new GeoFire(DriverAvailibilityRef);

            DatabaseReference DriverWorkingRef =
    FirebaseDatabase.getInstance().getReference().child("DRIVERS WORKING");
    GeoFire geoFireWorking = new GeoFire(DriverWorkingRef);

switch (customerID) {case "":
    geoFireWorking.removeLocation(driverId); geoFireAvailability.setLocation(driverId, new
    GeoLocation(location.getLatitude(),
```

```
    location.getLongitude()));
        break;default:
    geoFireAvailability.removeLocation(driverId); geoFireWorking.setLocation(driverId, new
    GeoLocation(location.getLatitude(),
    location.getLongitude()));
    break;
    }
    }
    }

    @NonNull
private CameraPosition getCameraPositionWithBearing(LatLng latLng) {return new
    CameraPosition.Builder().target(latLng).zoom(15).build();
    }

    @Override
protected void onStop() {super.onStop();

    if (!currentLogoutDriverStatus)
    {
    DisconnectTheDriver();
    }
    }

    private void DisconnectTheDriver() {

    String driverId = FirebaseAuth.getInstance().getCurrentUser().getUid();DatabaseReference
    DriverAvailibilityRef =
    FirebaseDatabase.getInstance().getReference().child("DRIVERS AVAILABLE");

    GeoFire geoFire = new GeoFire(DriverAvailibilityRef);geoFire.removeLocation(driverId);

    }

    private void LogOutDriver() {
    Intent welcomeIntent=new Intent(DriverMapActivity.this,DriverLoginActivity.class);
    welcomeIntent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK |
```

```
        Intent.FLAG_ACTIVITY_CLEAR_TASK);
        startActivity(welcomeIntent);finish();
        }
        // RunTime permission for Google Location Servicesprivate void GPSLocationPermission()
        {
        LocationRequest mLocationRequest = LocationRequest.create()
        .setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY)
        .setInterval(1000)
        .setFastestInterval(1000);

        LocationSettingsRequest.Builder settingsBuilder = new LocationSettingsRequest.Builder()
                .addLocationRequest(mLocationRequest);settingsBuilder.setAlwaysShow(true);

        Task<LocationSettingsResponse> result = LocationServices.getSettingsClient(this)
        .checkLocationSettings(settingsBuilder.build());

    result.addOnCompleteListener(new OnCompleteListener<LocationSettingsResponse>() {@Override
    public void onComplete(@NonNull Task<LocationSettingsResponse> task) {try {
LocationSettingsResponse response = task.getResult(ApiException.class);
    } catch (ApiException ex) { switch (ex.getStatusCode()) {
    case LocationSettingsStatusCodes.RESOLUTION_REQUIRED:try {
ResolvableApiException resolvableApiException =(ResolvableApiException) ex;
        resolvableApiException
        .startResolutionForResult(DriverMapActivity.this,Request_User_Location_Code);

        }
        }
        }
        });
        } catch (IntentSender.SendIntentException e) {

        }
        break;
        case LocationSettingsStatusCodes.SETTINGS_CHANGE_UNAVAILABLE:

        break;

        }

        public boolean checkUserLocationPermission()
        {
                if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION) !=
        PackageManager.PERMISSION_GRANTED && ContextCompat.checkSelfPermission(this,
        Manifest.permission.ACCESS_COARSE_LOCATION)  !=  PackageManager.PERMISSION_GRANTED)
```

```
        {

        // Permission is not granted
        // Should we show an explanation?

if (ActivityCompat.shouldShowRequestPermissionRationale(DriverMapActivity.this,
        Manifest.permission.ACCESS_FINE_LOCATION)) {
        // Show an explanation to the user *asynchronously* -- don't block
        // this thread waiting for the user's response! After the user
        // sees the explanation, try again to request the permission.

        new AlertDialog.Builder(this)
                        .setTitle("Required Location Permission").setMessage("You have to give this permissionto
        access this feature")
    .setPositiveButton("OK", new DialogInterface.OnClickListener() {@Override
    public void onClick(DialogInterface dialog, int which) { ActivityCompat.requestPermissions(DriverMapActivity.this,
        new String[]{Manifest.permission.ACCESS_FINE_LOCATION},Request_User_Location_Code);



        }
        })
    .setNegativeButton("Cancel", new DialogInterface.OnClickListener() {@Override
    public void onClick(DialogInterface dialog, int which) {dialog.dismiss();
        }
        })
                        .create()
                        .show();
        } else {

        // No explanation needed; request the permission ActivityCompat.requestPermissions(DriverMapActivity.this,
        new String[]{Manifest.permission.ACCESS_FINE_LOCATION},Request_User_Location_Code);
        }
        return false;
        }
        else
        {
        return true;
        }
        }
        }
```

CategoryActivity.java-

```
packagecom.jack2002.miniproject2;

import androidx.annotation.NonNull;import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;import androidx.fragment.app.Fragment;

import  android.Manifest; import android.app.AlertDialog;
import android.content.DialogInterface;import android.content.Intent;
import android.content.IntentSender;
import android.content.pm.PackageManager;import android.location.Location;
import android.os.Build; import android.os.Bundle; import android.view.Menu;
import android.view.MenuItem;import android.view.Window;
import android.view.WindowManager;import android.widget.Toast;

import com.google.android.gms.common.ConnectionResult;
```

```
import com.google.android.gms.common.api.ApiException; import
com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.common.api.ResolvableApiException;import
com.google.android.gms.location.LocationListener;
import com.google.android.gms.location.LocationRequest; import
com.google.android.gms.location.LocationServices;
import com.google.android.gms.location.LocationSettingsRequest; import
com.google.android.gms.location.LocationSettingsResponse; import
com.google.android.gms.location.LocationSettingsStatusCodes;import
com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.android.material.bottomnavigation.BottomNavigationView; import
com.google.firebase.auth.FirebaseAuth;

public class CategoryActivity extends AppCompatActivity implementsGoogleApiClient.ConnectionCallbacks,
GoogleApiClient.OnConnectionFailedListener,LocationListener {

private long backPressedTime;

Fragment selectedFragment=null;
private static final int Request_User_Location_Code=99;

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

if (Build.VERSION.SDK_INT>=21)
{
Window window=this.getWindow();

window.addFlags(WindowManager.LayoutParams.FLAG_DRAWS_SYSTEM_BAR_BACKGROUNDS);
window.clearFlags(WindowManager.LayoutParams.FLAG_TRANSLUCENT_STATUS);
window.setStatusBarColor(this.getResources().getColor(R.color.colorPrimary));
}

if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M)
{
```

```
        checkUserLocationPermission();
        }
        GPSLocationPermission();

        BottomNavigationView bottomnavigation = findViewById(R.id.bottom_navigation);
        bottomnavigation.setOnNavigationItemSelectedListener(navlistener);

        Fragment selectedFragment = new HomeFragment();
        getSupportFragmentManager().beginTransaction().replace(R.id.fragment_container,
        selectedFragment).commit();
        }

        // RunTime permission for Google Location Servicesprivate void GPSLocationPermission()
        {
        LocationRequest mLocationRequest = LocationRequest.create()
        .setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY)
        .setInterval(10 * 1000)
        .setFastestInterval(1 * 1000);

        LocationSettingsRequest.Builder settingsBuilder = new LocationSettingsRequest.Builder()
                .addLocationRequest(mLocationRequest);settingsBuilder.setAlwaysShow(true);

        Task<LocationSettingsResponse> result = LocationServices.getSettingsClient(this)
        .checkLocationSettings(settingsBuilder.build());

    result.addOnCompleteListener(new OnCompleteListener<LocationSettingsResponse>() {@Override
    public void onComplete(@NonNull Task<LocationSettingsResponse> task) {try {
LocationSettingsResponse response = task.getResult(ApiException.class);
    } catch (ApiException ex) { switch (ex.getStatusCode()) {
    case LocationSettingsStatusCodes.RESOLUTION_REQUIRED:try {
ResolvableApiException resolvableApiException =(ResolvableApiException) ex;
```

```
        resolvableApiException
        .startResolutionForResult(CategoryActivity.this,Request_User_Location_Code);
        } catch (IntentSender.SendIntentException e) {

        }
        break;
        case LocationSettingsStatusCodes.SETTINGS_CHANGE_UNAVAILABLE:

        break;
        }
        }
        }
        });
        }

        public boolean checkUserLocationPermission()
        {
              if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION) !=
        PackageManager.PERMISSION_GRANTED && ContextCompat.checkSelfPermission(this,
        Manifest.permission.ACCESS_COARSE_LOCATION)  !=  PackageManager.PERMISSION_GRANTED)
        {

        // Permission is not granted
        // Should we show an explanation?

if (ActivityCompat.shouldShowRequestPermissionRationale(CategoryActivity.this,
        Manifest.permission.ACCESS_FINE_LOCATION)) {
        // Show an explanation to the user *asynchronously* -- don't block
        // this thread waiting for the user's response! After the user
        // sees the explanation, try again to request the permission.

        new AlertDialog.Builder(this)
                        .setTitle("Required Location Permission").setMessage("You have to give this permissionto
        access this feature")
    .setPositiveButton("OK", new DialogInterface.OnClickListener() {@Override
    public void onClick(DialogInterface dialog, int which) { ActivityCompat.requestPermissions(CategoryActivity.this,
                                new String[]{Manifest.permission.ACCESS_FINE_LOCATION},
```

```
    Request_User_Location_Code);
    }
    })
  .setNegativeButton("Cancel", new DialogInterface.OnClickListener() {@Override
  public void onClick(DialogInterface dialog, int which) {dialog.dismiss();
    }
    })
                        .create()
                        .show();
    } else {

    // No explanation needed; request the permission ActivityCompat.requestPermissions(CategoryActivity.this,
    new String[]{Manifest.permission.ACCESS_FINE_LOCATION},Request_User_Location_Code);
    }
    return false;
    }
    else
    {
    return true;
    }
    }

private BottomNavigationView.OnNavigationItemSelectedListener navlistener =new
    BottomNavigationView.OnNavigationItemSelectedListener() {
    @Override
    public boolean onNavigationItemSelected(@NonNull MenuItem menuItem) {

  switch (menuItem.getItemId()) {case R.id.nav_home:
    selectedFragment = new HomeFragment();break;

    case R.id.nav_about:
    selectedFragment = new AboutFragment();break;

    case R.id.nav_profile:
```

```
    selectedFragment = new ProfileFragment();break;
    }
                    getSupportFragmentManager().beginTransaction().replace(R.id.fragment_container,
    selectedFragment).commit();

    return true;
    }
    };

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {

    getMenuInflater().inflate(R.menu.menu, menu);return true;

    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {

    int id = item.getItemId();

    switch (id) {
case R.id.logout: FirebaseAuth.getInstance().signOut();
    Intent intent = new Intent(CategoryActivity.this, Login.class);
    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK);
    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP); startActivity(intent);
    break;
    }
    return true;
    }

    @Override
    public void onBackPressed() {

if (backPressedTime + 2000 > System.currentTimeMillis()) {Intent a = new Intent(Intent.ACTION_MAIN);
```

```
a.addCategory(Intent.CATEGORY_HOME); a.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);startActivity(a);
} else {
Toast.makeText(this, "Press back again to exit", Toast.LENGTH_SHORT).show();
}
backPressedTime = System.currentTimeMillis();
}

@Override
public void onConnected(@Nullable Bundle bundle) {

}

@Override
public void onConnectionSuspended(int i) {

}

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

}

@Override
public void onLocationChanged(Location location) {

}
}
```

Login.java-

```
package com.jack2002.miniproject2;

import android.app.ProgressDialog;import android.content.Intent; import android.os.Bundle;
import android.view.View; import android.widget.Button; import android.widget.TextView;
```

```
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import com.google.android.gms.tasks.OnCompleteListener;import com.google.android.gms.tasks.Task;
import com.google.android.material.textfield.TextInputLayout;import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

public class Login extends AppCompatActivity {

    TextInputLayout inputemail,inputpassword;Button btnlogin;
    TextView tvregister,tvforgotpassword;FirebaseAuth mfirebaseauth;

    private long backPressedTime;
    private ProgressDialog progressDialog;

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

    inputemail=findViewById(R.id.inputemail); inputpassword=findViewById(R.id.inputpassword);
    btnlogin=findViewById(R.id.btnlogin); tvregister=findViewById(R.id.tvregister);
    tvforgotpassword=findViewById(R.id.tvforgotpassword);

    mfirebaseauth = FirebaseAuth.getInstance();

btnlogin.setOnClickListener(new View.OnClickListener() {@Override
    public void onClick(View v) {
    String email=inputemail.getEditText().getText().toString().trim();
    String password=inputpassword.getEditText().getText().toString().trim();
```

```
if(email.isEmpty())
{
inputemail.setError("Please enter email id");inputemail.requestFocus();
}
else if(password.isEmpty())
{
inputpassword.setError("Please enter password");inputpassword.requestFocus();
}
else if(email.isEmpty() && password.isEmpty())
{
        Toast.makeText(Login.this, "Login failed....please fill up the details",
Toast.LENGTH_LONG).show();
}
else if(!(email.isEmpty() && password.isEmpty()))
{
progressDialog=new ProgressDialog(Login.this); progressDialog.setMessage("Logging in....please wait...");
progressDialog.show();


mfirebaseauth.signInWithEmailAndPassword(email,password).addOnCompleteListener(Login.this, new
OnCompleteListener<AuthResult>() {
@Override
public void onComplete(@NonNull Task<AuthResult> task) {if(!task.isSuccessful())
{
progressDialog.dismiss();
            Toast.makeText(Login.this, "Login Unsuccessful....please try again later",
Toast.LENGTH_LONG).show();
}
else
{
progressDialog.dismiss();
startActivity(new Intent(Login.this,CategoryActivity.class));
}
}
});
}
else
{
progressDialog.dismiss();
        Toast.makeText(Login.this,"Error Occured....please try again later",
Toast.LENGTH_LONG).show();
}
}});
```

```
tvregister.setOnClickListener(new View.OnClickListener() {@Override
    public void onClick(View v) {
    Intent i=new Intent(Login.this, OTPverification.class);startActivity(i);
    }
    });
tvforgotpassword.setOnClickListener(new View.OnClickListener() {@Override
    public void onClick(View v) {
    Intent intent=new Intent(Login.this,Forgot_password_Activity.class);startActivity(intent);
    }
    });
    }
    @Override
    public void onBackPressed() {

    if(backPressedTime + 2000 > System.currentTimeMillis())
    {
    Intent a = new Intent(Intent.ACTION_MAIN); a.addCategory(Intent.CATEGORY_HOME);
    a.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);startActivity(a);
    }
    else
    {
    Toast.makeText(this, "Press back again to exit", Toast.LENGTH_SHORT).show();
    }
    backPressedTime=System.currentTimeMillis();
    }
    }
```

MapsActivity.java-

```
package com.jack2002.miniproject2;

import androidx.fragment.app.FragmentActivity;import android.os.Bundle;

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;

public class MapsActivity extends FragmentActivity implements OnMapReadyCallback {private GoogleMap

mMap;

@Override
protected void onCreate(Bundle savedInstanceState) {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(this);
}

/**
 * Manipulates the map once available.
 * This callback is triggered when the map is ready to be used.

 * This is where we can add markers or lines, add listeners or move the camera. In this case,
 * we just add a marker near Sydney, Australia.
 * If Google Play services is not installed on the device, the user will be prompted to install
 * it inside the SupportMapFragment. This method will only be triggered once the user has
 * installed Google Play services and returned to the app.
  */
  @Override
public void onMapReady(GoogleMap googleMap) {mMap = googleMap;

// Add a marker in Sydney and move the cameraLatLng sydney = new LatLng(-34, 151);
mMap.addMarker(new MarkerOptions().position(sydney).title("Marker in Sydney"));
mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
}
} } }
```

Register.java-

```
package com.jack2002.miniproject2;

import androidx.annotation.NonNull;

import androidx.appcompat.app.AppCompatActivity;
import android.app.ProgressDialog;import android.content.Intent; import

android.os.Build;

import android.os.Bundle; import android.util.Log; import

android.view.View;

import android.view.Window;

import android.view.WindowManager;import android.widget.Button;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;

import com.google.android.gms.tasks.Task;

import com.google.android.material.textfield.TextInputLayout;

import com.google.firebase.auth.AuthResult;


import com.google.firebase.auth.FirebaseAuth;

import com.google.firebase.auth.FirebaseAuthException; import

com.google.firebase.database.DatabaseReference;import

com.google.firebase.database.FirebaseDatabase;


public class Register extends AppCompatActivity {
    private TextInputLayout

    inputname,inputemail,inputpassword,inputage,inputaddress,inputpin;private

    Button btnsubmit;

    DatabaseReference ref; FirebaseAuth mfirebaseauth;Member member;
```

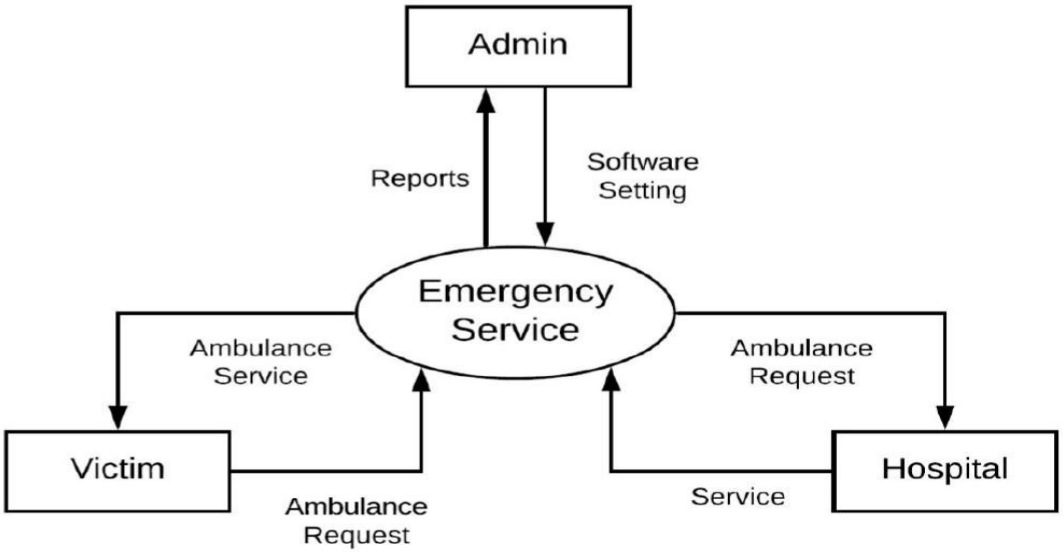
```
String phonenumber;
private ProgressDialog progressDialog;@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState); setContentView(R.layout.activity_register);

if (Build.VERSION.SDK_INT>=21)
{
Window window=this.getWindow();


window.addFlags(WindowManager.LayoutParams.FLAG_DRAWS_SYSTEM_BAR_BA
CKGROUND   S);
window.clearFlags(WindowManager.LayoutParams.FLAG_TRANSLUCENT_STATUS);
window.setStatusBarColor(this.getResources().getColor(R.color.colorPrimary));
}

inputname=findViewById(R.id.inputname);
inputemail=findViewById(R.id.inputemail);
inputpassword=findViewById(R.id.inputpassword);
inputage=findViewById(R.id.inputage);
inputaddress=findViewById(R.id.inputaddress);
inputpin=findViewById(R.id.inputpin); btnsubmit=findViewById(R.id.btnsubmit);

mfirebaseauth = FirebaseAuth.getInstance();member=new Member();
phonenumber = getIntent().getStringExtra("phonenumber");
//              ref= FirebaseDatabase.getInstance().getReference().child("USERS");



btnsubmit.setOnClickListener(new View.OnClickListener() {@Override
public void onClick(View v) {

if(inputname.getEditText().getText().toString().trim().length()==0)
inputname.setError("Name is required");
else if (inputemail.getEditText().getText().toString().trim().length()==0)
```

```
inputemail.setError("Email is required");
        else if (inputpassword.getEditText().getText().toString().trim().length()==0)
inputpassword.setError("Password is required");
        else if(inputage.getEditText().getText().toString().trim().length()==0)
inputage.setError("Age is required");
else if (inputaddress.getEditText().getText().toString().trim().length()==0)
inputaddress.setError("Address is required");
else if (inputpin.getEditText().getText().toString().trim().length()==0)
inputpin.setError("Pic Code of your area is required");
else {
progressDialog=new ProgressDialog(Register.this);
progressDialog.setMessage("Logging in....please wait...");progressDialog.show();

}
                                                                                    });
CreateUserAndSaveData();
}

}

@Override
public void onBackPressed() {


finish();
startActivity(new Intent(Register.this, Login.class));
}
```

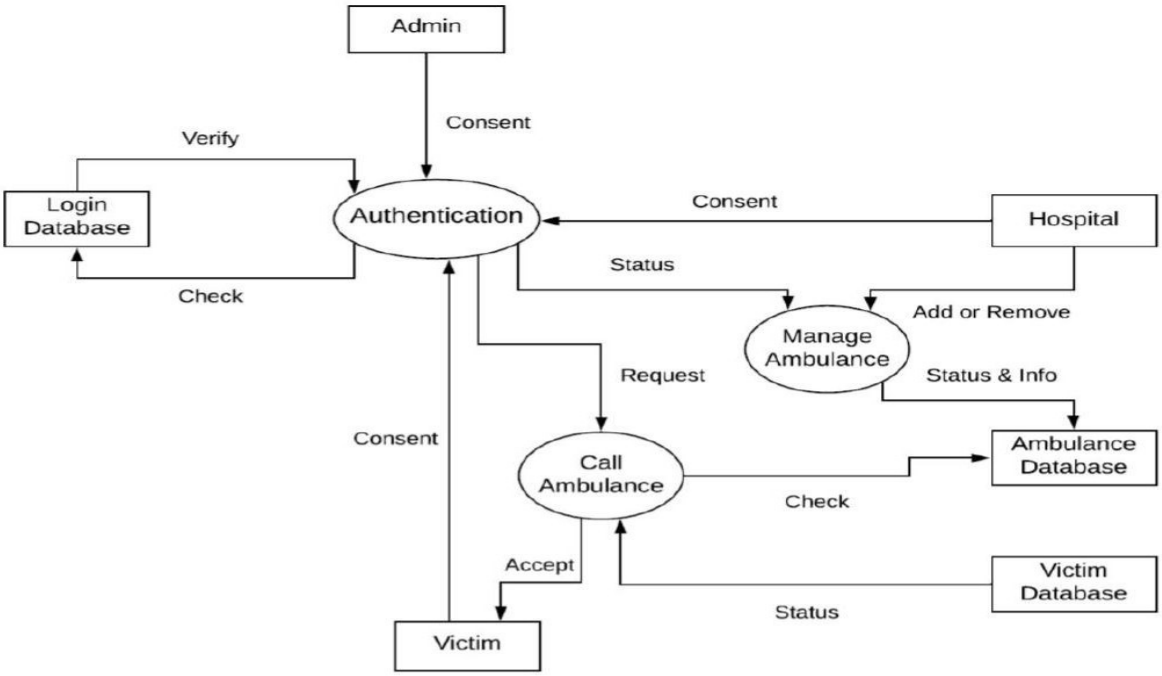
```
private void CreateUserAndSaveData() {

mfirebaseauth.createUserWithEmailAndPassword(inputemail.getEditText().getText().t
oString(),input
password.getEditText().getText().toString()).addOnCompleteListener(this,  new
OnCompleteListener<AuthResult>() {
@Override
public void onComplete(@NonNull Task<AuthResult> task) {if (task.isSuccessful())
{
progressDialog.dismiss();saveData();


Toast.makeText(Register.this,  "User  Successfully  Registered...",
Toast.LENGTH_SHORT).show();
}
else
{
progressDialog.dismiss();
FirebaseAuthException e = (FirebaseAuthException)task.getException();
Log.i("ERROR",e.getMessage());
        Toast.makeText(Register.this, "User Registration Failed....Please try again
later",Toast.LENGTH_SHORT).show();
}}});}

private void saveData()
{
phonenumber = getIntent().getStringExtra("phonenumber");


member.setName(inputname.getEditText().getText().toString().trim());
member.setEmail(inputemail.getEditText().getText().toString().trim());
member.setPassword(inputpassword.getEditText().getText().toString().trim());
```

```
member.setAge(inputage.getEditText().getText().toString().trim());
member.setAddress(inputaddress.getEditText().getText().toString().trim());
member.setPin(inputpin.getEditText().getText().toString().trim());
member.setContactnumber(phonenumber);

    ref=FirebaseDatabase.getInstance().getReference().child("USERS").child(mfirebas
eauth.getCurrentUser ().getUid());
ref.setValue(member);
FirebaseAuth.getInstance().signOut(); startActivity(new Intent(Register.this, Login.class));
}
```

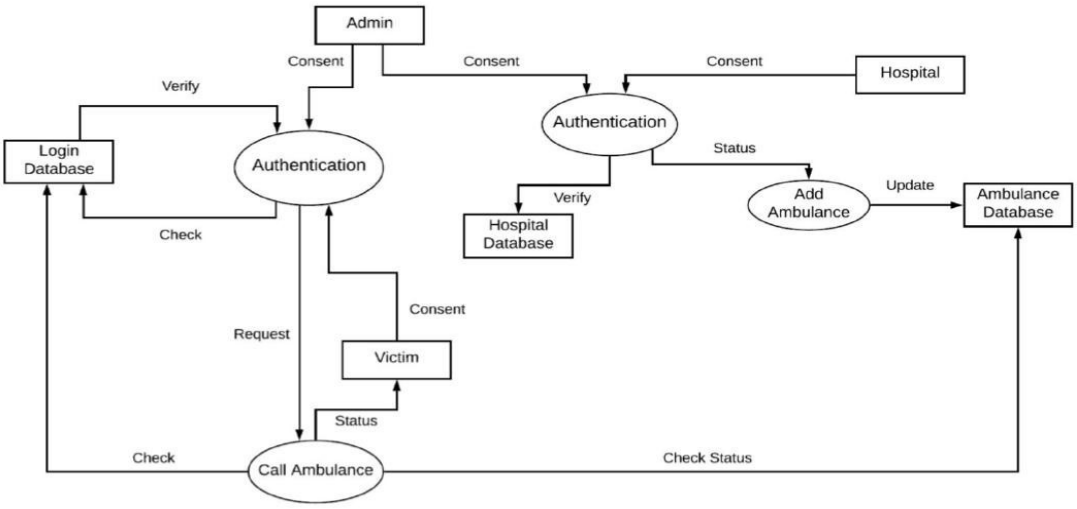
DATA FLOW DIAGRAM



Level 0



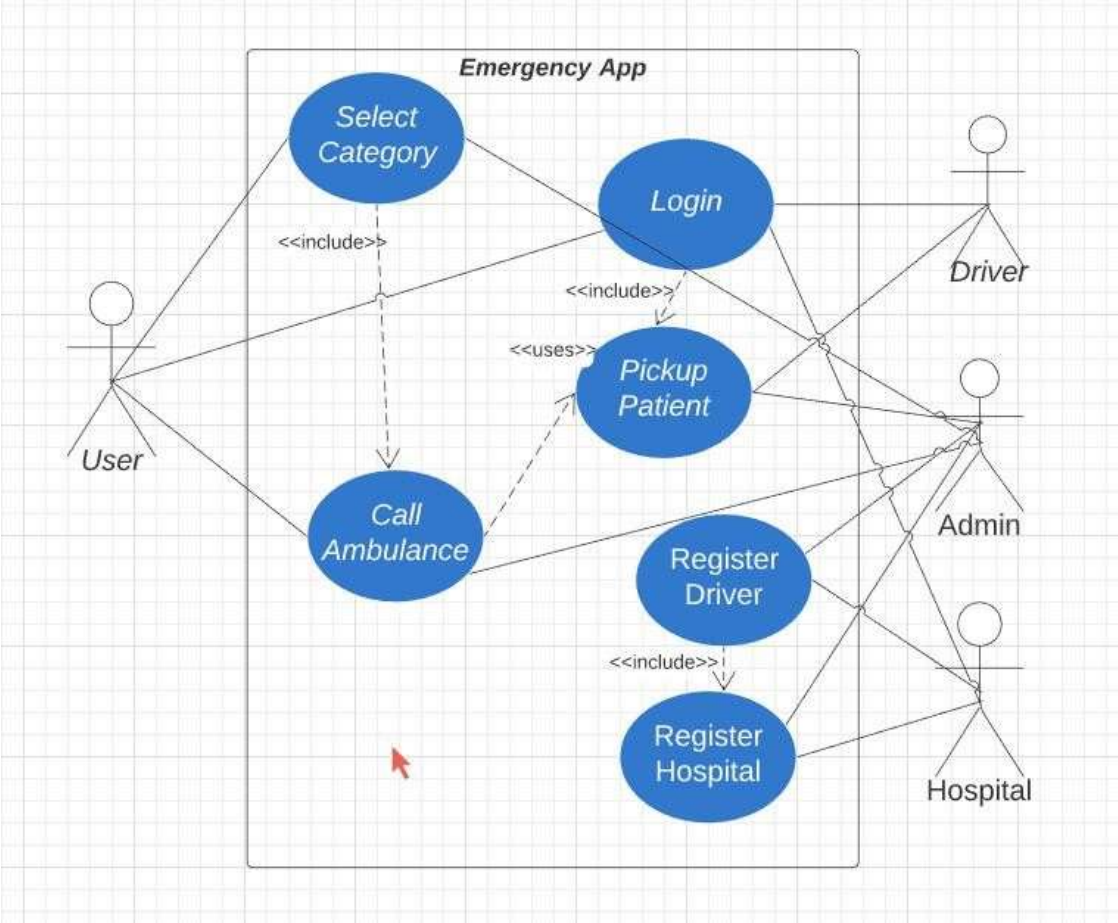
Level 1



Level 2

Level 2

USE CASE DIAGRAM



CONCLUSION

Our App will provide emegency ambulance service for the accident victims. All Hospitals will use this app. When an accident occurs the request for ambulance service can be made by the victim or by any tracepasser when the victim is in serious condition. The request will be sent to the nearest hospital andthen the ambulance will be sent from there to the location of the accident as soon as possible.

REFERENCE

1. Basic definition on Arduino: https://en.wikipedia.org/wiki/Arduino

2. Definition on Wiring: https://en.wikipedia.org/wiki/Wiring_(development_platform)

3. Definition on Heart Rate Monitor:
https://en.wikipedia.org/wiki/Heart_rate_monitor

4. M. R. Yuce, S. W. P. Ng, N. L. Myo, J. Y. Khan, and W. Liu, "Wireless body sensor network using medical implant band," J. Med. Syst., vol. 31, pp. 467–474, Dec. 2007.

5. B. Gyselinckx, J. Penders, and R. Vullers, "Potential and challenges of body area networks for cardiac monitoring," J. Electrocardiol., vol. 40, pp. S165–S168, Nov. 2007.

6. U. Varshney, "Pervasive Healthcare and Wireless Health Monitoring" Springer, Mobile NetwAppl (2007) 12:113–127 DOI 10.1007/s11036-007-0017-1.

7. MirelaPrgomet, Andrew Georgiou, and Johanna Westbrook, "The Impact of Mobile Handheld Technology on Hospital Physicians' Work Practices and Patient Care", Journal of the American Medical Informatics Association, Volume 16, no. 6 (November/December, 2009), pp. 792- 801.