# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**JNANA SANGAMA, BELAGAVI-590018**

**A MAD MINI PROJECT REPORT**
**ON**

# "SOS APPLICATION"

**A PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF**

**THE REQUIREMENT FOR THE DEGREE OF**

**BACHELOR OF ENGINEERING**
IN
**COMPUTER SCIENCE & ENGINEERING**

**SUBMITTED BY**

**ANUSHA K BHAT (1RG18CS003)**

**FAIQUA RAHMAN (1RG18CS017)**

**UNDER THE GUIDANCE OF,**

**MRS. PUSHPLATA DUBEY**

**ASSISTANT PROFESSOR, DEPT. OF CSE**
**RGIT, BENGALURU-32**

**RAJIV GANDHI INSTITUTE OF TECHNOLOGY**
**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**
Cholanagar, R.T.Nagar Post, Bengaluru-560032
2020-2021

# VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**(Affiliated To Visvesvaraya Technological University) Cholanagar, R.T.Nagar Post, Bangalore-560032**

## Department of Computer Science Engineering

## CERTIFICATE

This is to certify that the MAD Mini Project Report entitled **"SOS APPLICATION"** is a bonafide work carried out by **Ms. ANUSHA K BHAT (1RG18CS003) & Ms. FAIQUA RAHMAN (1RG18CS017)** in partial fulfillment for the award of **Bachelor of Engineering in Computer Science Engineering** under **Visvesvaraya Technological University, Belgavi,** during the year **2020-2021.** It is certified that all corrections/suggestions given for Internal Assessment have been incorporated in the report. This Mini Project report has been approved as it satisfies the academic requirements in respect of seminar work.

| | |
|---|---|
| …………………… | ………………… |
| Signature of Guide | Signature of HOD |
| **Mrs. Pushplata Dubey** | **Mrs. Arudra A** |
| Asst. Professor | Professor & HOD |
| Dept. of CSE | Dept. of CSE |
| RGIT, Bengaluru | RGIT, Bengaluru |

## External Viva

| **Name of the Examiners** | **Signature With Date** |
|---|---|
| **1** | |
| **2.** | |

## VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**JNANA SANGAMA, BELAGAVI-590018**

# RAJIV GANDHI INSTITUTE OF TECHNOLOGY

## DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

## DECLARATION

I hereby declare that the mini project work entitled **" SOS APPLICATION"** submitted to the **Visvesvaraya Technological University, Belagavi** during the academic year **2020-2021**, is record of an original work done by us under the guidance of **Mrs. Pushplata Dubey , Assistant Professor, Department of Computer Science and Engineering, Rajiv Gandhi Institute of Technology, Bengaluru** and this project work is submitted in the partial fulfillment of requirements for the award of the degree of **Bachelor of Engineering** in **Computer Science & Engineering.** The results embodied in this thesis have not been submitted to any other University or Institute for award of any degree or diploma.

**ANUSHA K BHAT**
**(1RG18CS003)**

**FAIQUA RAHMAN**
**(1RG18CS017)**

# ACKNOWLEDGEMENT

I take this opportunity to express my sincere gratitude and respect to the **Rajiv Gandhi Institute of Technology, Bengaluru** for providing me an opportunity to carry out my project work.

I express my sincere regards and thanks to **Dr. D G ANAND, Principal, RGIT, Bengaluru,** and **Mrs. ARUDRA A**, **Professor and Head, Department of Computer Science & Engineering, RGIT, Bengaluru,** for their encouragement and support throughout the Project.

With profound sense of gratitude, I acknowledge the guidance and support extended by **Mrs. Pushplata Dubey, Asst. Professor, Department of Computer Science & Engineering, RGIT, Bengaluru.** Her incessant encouragement and valuable technical support have been of immense help in realizing this project. Her guidance gave me the environment to enhance my knowledge, skills and to reach the pinnacle with sheer determination, dedication and hard work.

I also extend my thanks to the entire faculty of the Department of CSE, RGIT, Bengaluru, who have encouraged me throughout the course of Bachelor Degree.

**ANUSHA K BHAT**
**(1RG18CS003)**

**FAIQUA RAHMAN**
**(1RG18CS017)**

# ABSTRACT

The aim of the project is to develop an Android application that lets its users to send notifications in case of an emergency or a panic situation. The users can send  text messages with location on shaking the  phone 3 times    . The phone numbers, and names can be set from within the application. The text messages  sent, along with the content, also have the last known location of the user. This is very helpful in tracking the whereabouts of the person. The user can also send message on 112 directly from within the application, if the nature of the situation demands it and user’s. GPS is off.  Additionally, the user of the application may allow the app to track their location. This application  is very useful and can be used in a variety of ways. One such use of the location data is from within the Android app where the user can view a map that shows their location history over a period of time for a particular day.

# TABLE OF CONTENTS

# TABLE OF FIGURES

# CHAPTER 1

# INTRODUCTION

## 1.1 Project Overview

The project “SOS APPLICATION” is developed using java in Android Studio. This is basically an advanced emergency apps that can rescue you and/or your loved ones if you and/or they find themselves in a life-threatening emergency situation and need immediate assistance. When you need some personal assistance, you can actually turn on your phone and can call or message someone for help. But in a life-threatening emergency like attack, sexual assault, robbery, harassment, accident, fire, birth assistance, we don’t have time to open our phone, instead, we need some accessibility methods by which we can reach out for help without actually operating the phone.

## 1.2 Background of Project

SOS is an emergency handling application in Android. In an emergency, a user with this app can send message to the registered contacts by shaking the phone 3 times and can send their current location with a single button press on the app. It helps to track the user and the location of the user will sent to people stating that “It’s emergency” with the address through SMS. (The users are allowed to type the content of the message at beginning that sent to relatives at emergency times.) All you have to do select a few numbers of your relatives and friends. The app updates the location when the user starts to use and sent the present location to their registered relatives and friends. In addition, this app trigger to provide the alert sound when the user presses the “ALERT” button, the nearby people can come to know there is something happening by the alert sound. It will be very useful to everyone using Mobile and travelling more places. The app stands for the personal safety to the mobile users. When the GPS is off then when the user shakes their phone. A message will be sent within 10 seconds sayings “I am in DANGER!! I need help my GPS is turned off so the location can’t be sent. Call your nearest police station".

## 1.3 Scope Of The Project

The app helps in the emergency and provides the instant update on the current user to their parents/ friends. When there is an emergency, by using the Smartphone it will take some time like to open every app individually such as Google maps, messages app or any other apps for that matter. Sometimes opening these apps will consume time, so if we integrate all the functionalities within one app with one action that will be more useful for many users do these kinds of operation very frequently. As the app provides the automation to the user, thus it will automatically send messages to our parents/ friends, to alert by sounds with the same app. When there is some unknown situation what actually happening, the app provides the option to check out the location to google map API, also to play some alert sound, send message regarding our location within a single app to do all the operation instantly. The app will be more useful in the emergency and make to the user to feel safe with the app in their Smartphone.

## 1.4 Aim and Objective of the Project

The project aims and objectives that will be achieved after completion of this project are mentioned below:

- Gives an assured security for the user.
- User login, where the user can add contacts to the “Emergency Contact List” and can also send a SMS when in emergency.

# CHAPTER 2

# REQUIREMENT SPECIFICATIONS

## 2.1 Details of Software and Languages

### 2.1.1 Introduction to Android Studio:

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA . On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

- A flexible Gradle-based build system.
- A fast and feature-rich emulator.
- A unified environment where you can develop for all Android devices.
- Apply Changes to push code and resource changes to your running app without restarting your app.
- Code templates and GitHub integration to help you build common app features and import sample code.
- Extensive testing tools and frameworks.
- Lint tools to catch performance, usability, version compatibility, and other problems.
- C++ and NDK support.
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine.

### 2.1.2 Introduction to Firebase:

Firebase is a product of Google which helps developers to build, manage, and grow their apps easily. It helps developers to build their apps faster and in a more secure way. No programming is required on the firebase side which makes it easy to use its features more efficiently. It provides services to android, iOS, web, and unity. It provides cloud storage. It uses NoSQL for the database for the storage of data.

cccccccccc

## 2.2 System Requirements

### 2.2.1 Hardware Interface

- Processor: Pentium or Higher. (Preferred x64 bit – Intel i3 processor 1.6GHz)
- Ram: 8GB
- Hard Disk: 500GB (At least 80GB)
- Display: Intel HD graphics (128MB)
- Keyboard, Mouse

### 2.2.2 Software Interface

- Client on Internet: Web Browser,
- Operating System (any)

### 2.2.3 Tools Used

- ❖ **Android Studio:**

# CHAPTER 3

# SYSTEM DESIGN

## 3.1 Design Approach

This project is based on the functional design approach, which helps in understanding the design of the project in a simpler way by explaining its flow, use cases, and implementation more like a modular approach. For example, there are different modules in this project which have separate functionality and, other sub functionalities/modules. This module body treats data as nodes connected to each other; however, it allows for more complex connections, such as many-to-many relationships and cycles. This schema can model the movement of attributes and entities between locations, or the workflow required to accomplish a particular task. Furthermore, this is a blueprint for how the outlook is going to be. The design is an abstract structure or outline that represents the pictorial view of the application as a whole. By defining categories of data and relationships between those categories design makes data much easier to retrieve, consume, manipulate, and interpret. The schema design organizes the data into separate entities, determines how to create relationships between organized entities, and how to apply the constraints on the data. All the modules are designed, implemented and integrated together to make a flawless working application.

## 3.2 Front End design

Design for any application should be very simple. We should have only a few clicks or navigation among the features when using the application to avoid hassle.

In this application, there is one main screen, the Home screen.

The homepage where the users can access the contacts for emergency contact list by using phone contact details.

All you have to do is select a few numbers of your relatives and friends. The app updates the location when the user starts to use and sends the present location to their registered relatives and friends.

In addition, this app trigger to provide the alert sound when the user presses the “ALERT” button, the nearby people can come to know there is something happening by the alert sound.

To make the application interactive, different controls have been used and designed using the layout file. Following are the important controls that are designed and used in this application-

cccccccccc

- Text View: The text view component belongs to the view group as a part of GUI. It displays the text or content view of any activity to the user and allows them to edit in the code section.
- Edit Text: This allows itself to be editable in the text box.
- Button: One of the important components in which the application needs. It is mainly associated with action when the user clicks it. We can represent the button using any text which holds the action class on it.
- List View: This is a key component under the view group which helps in displaying the information about anything when we click the action button. It also allows us to scroll through the screen and have a look about the information displayed. Using the list adapter, the content is pulled from the database.

Layout - A layout defines the structure for a user interface in your app, such as in an activity. All elements in the layout are built using a hierarchy of View and View Group objects.

- Constraint Layout: This is a View Group which allows you to position and size widgets in a flexible way.
- Relative Layout: This is a view group that displays child views in relative positions. The position of each view can be specified as relative to sibling elements or in positions relative to the parent Relative Layout area.
- Linear Layout: This is a view group that aligns all children in a single direction, vertically or horizontally.
- Table Layout: This used to arrange the group of views into rows and columns. Table Layout containers do not display a border line for their columns, rows or cells. A Table will have as many columns as the row with the most cells.

## 3.3: ER Diagram

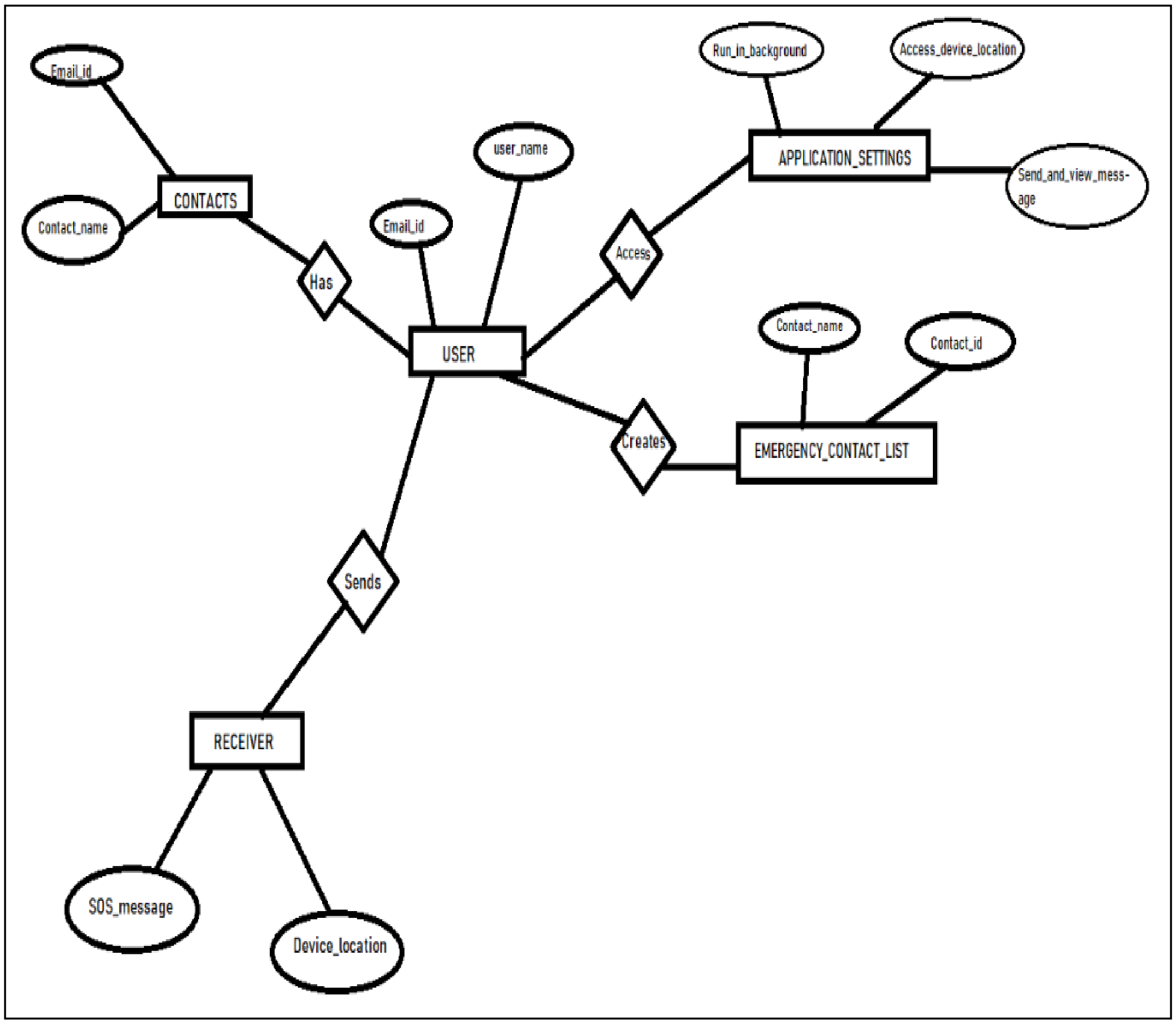


**Figure 3.2 ER diagram: SOS MOBILE APPLICATION**

# CHAPTER 4

# SYSTEM IMPLEMENTATION

## 4.1 Main Activity.XML

Source code:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/Button1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="12dp"
        android:background="#D3C4C8"
        android:backgroundTint="#ECE2E2"
        android:text="Add Emergency Contact "
        android:textColor="#DC0745"
        android:textSize="18sp"
        android:textStyle="bold" />


    <ListView
        android:id="@+id/ListView"
        android:layout_width="match_parent"
        android:layout_height="356dp">


    </ListView>


    <ImageButton
        android:id="@+id/button2"
        android:layout_width="match_parent"
        android:layout_height="158dp"
        android:outlineAmbientShadowColor="#D1C1C7"
        android:src="@drawable/ic_baseline_add_alert_24"
```

cccccccccc

```
android:text="press here " />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="46dp"
        android:text="PRESS THE BELL!!!"
        android:textAlignment="center"
        android:textColor="#DA0D53"
        android:textSize="30sp"
        android:textStyle="bold" />

</LinearLayout>
```

## 4.2 Activity Main.java

Source code:

```
package com.faiqua.sos;


import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;


import android.Manifest;
import android.annotation.SuppressLint;
import android.app.Activity;
import android.app.ActivityManager;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.media.MediaPlayer;
```

cccccccccc

```
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.os.PowerManager;
import android.provider.ContactsContract;
import android.provider.Settings;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ImageButton;
import android.widget.ListView;
import android.widget.Toast;

import com.faiqua.sos.Contacts.ContactModel;
import com.faiqua.sos.Contacts.CustomAdapter;
import com.faiqua.sos.Contacts.DbHelper;
import com.faiqua.sos.ShakeServices.ReactivateService;
import com.faiqua.sos.ShakeServices.SensorService;

import java.util.List;

public class MainActivity extends AppCompatActivity {

    private static final int IGNORE_BATTERY_OPTIMIZATION_REQUEST = 1002;
    private static final int PICK_CONTACT = 1;

    //create instances of various classes to be used
    Button button1;
    ImageButton button2;
    ListView listView;
    DbHelper db;
    List<ContactModel> list;
```

cccccccccc

```
 CustomAdapter customAdapter;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.activity_main);
       button2 = findViewById(R.id.button2);
       final MediaPlayer mediaPlayer = MediaPlayer.create(this,R.raw.ring);
       button2.setOnClickListener(new View.OnClickListener() {
           @Override
           public void onClick(View v) {
               mediaPlayer.start();

           }
       });

       //check for runtime permissions
       if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
           if(ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION)==
PackageManager.PERMISSION_DENIED) {
               requestPermissions(new
String[]{Manifest.permission.ACCESS_FINE_LOCATION,
Manifest.permission.SEND_SMS,Manifest.permission.READ_CONTACTS}, 100);
           }
       }

       //this is a special permission required only by devices using
       //Android Q and above. The Access Background Permission is responsible
       //for populating the dialog with "ALLOW ALL THE TIME" option
       if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.Q) {
           requestPermissions(new
String[]{Manifest.permission.ACCESS_BACKGROUND_LOCATION}, 100);
ccccccccccc
```

```
    }

    //check for BatteryOptimization,
    PowerManager pm = (PowerManager) getSystemService(POWER_SERVICE);

if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
        if (pm != null && !pm.isIgnoringBatteryOptimizations(getPackageName())) {
            askIgnoreOptimization();
        }
    }

    //start the service
    SensorService sensorService = new SensorService();
    Intent intent = new Intent(this, sensorService.getClass());
    if (!isMyServiceRunning(sensorService.getClass())) {
        startService(intent);
    }



    button1 = findViewById(R.id.Button1);
    listView=(ListView)findViewById(R.id.ListView);
    db=new DbHelper(this);
    list=db.getAllContacts();
    customAdapter=new CustomAdapter(this,list);
    listView.setAdapter(customAdapter);

    button1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            // calling of getContacts()
            if(db.count()!=5) {
                Intent intent = new Intent(Intent.ACTION_PICK,
```

cccccccccc

```
ContactsContract.Contacts.CONTENT_URI);
                startActivityForResult(intent, PICK_CONTACT);
            }else{
                Toast.makeText(MainActivity.this, "Can't Add more than 5 Contacts",
Toast.LENGTH_SHORT).show();
            }
        }
    });
}

//method to check if the service is running
private boolean isMyServiceRunning(Class<?> serviceClass) {
    ActivityManager manager = (ActivityManager)
getSystemService(Context.ACTIVITY_SERVICE);
    for (ActivityManager.RunningServiceInfo service :
manager.getRunningServices(Integer.MAX_VALUE)) {
        if (serviceClass.getName().equals(service.service.getClassName())) {
            Log.i ("Service status", "Running");
            return true;
        }
    }
    Log.i ("Service status", "Not running");
    return false;
}

@Override
protected void onDestroy() {
    Intent broadcastIntent = new Intent();
    broadcastIntent.setAction("restartservice");
    broadcastIntent.setClass(this, ReactivateService.class);
    this.sendBroadcast(broadcastIntent);
    super.onDestroy();
}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[]
permissions, @NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    if (requestCode==100){
        if(grantResults[0]==PackageManager.PERMISSION_DENIED){
            Toast.makeText(this, "Permissions Denied!\n Can't use the App!",
Toast.LENGTH_SHORT).show();
        }
    }
}

@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);

    //get the contact from the PhoneBook of device
    switch (requestCode) {
        case (PICK_CONTACT):
                if (resultCode == Activity.RESULT_OK) {

                    Uri contactData = data.getData();
                    Cursor c = managedQuery(contactData, null, null, null, null);
                    if (c.moveToFirst()) {

                    String id =
c.getString(c.getColumnIndexOrThrow(ContactsContract.Contacts._ID));
                    String hasPhone =
c.getString(c.getColumnIndex(ContactsContract.Contacts.HAS_PHONE_NUMBER));
                    String phone = null;
cccccccccc
```

```
                        try {
                            if (hasPhone.equalsIgnoreCase("1")) {
                                Cursor phones =
getContentResolver().query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI,
null,ContactsContract.CommonDataKinds.Phone.CONTACT_ID + " = " + id,null, null);
                                phones.moveToFirst();
                                phone = phones.getString(phones.getColumnIndex("data1"));
                             }
                            String name =
c.getString(c.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME));
                            db.addcontact(new ContactModel(0,name,phone));
                            list=db.getAllContacts();
                            customAdapter.refresh(list);
                        }
                        catch (Exception ex)
                        {
                        }
                        }
                    }
                    break;
        }
    }

    //this method prompts the user to remove any battery optimisation constraints from the
App
    private void askIgnoreOptimization() {

        if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.M) {
            @SuppressLint("BatteryLife") Intent intent = new
Intent(Settings.ACTION_REQUEST_IGNORE_BATTERY_OPTIMIZATIONS);
            intent.setData(Uri.parse("package:" + getPackageName()));
            startActivityForResult(intent, IGNORE_BATTERY_OPTIMIZATION_REQUEST);
cccccccccc
```

```
        }

    }

    }
```

## 4.3 ShakeDetector.java

Source code:

```
package com.faiqua.sos.ShakeServices;

import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;

public class ShakeDetector implements SensorEventListener {

    /*
     * The gForce that is necessary to register as shake.
     * Must be greater than 1G (one earth gravity unit).
     * You can install "G-Force", by Blake La Pierre
     * from the Google Play Store and run it to see how
     *  many G's it takes to register a shake
     */
    private static final float SHAKE_THRESHOLD_GRAVITY = 2.7F;
    private static final int SHAKE_SLOP_TIME_MS = 500;
    private static final int SHAKE_COUNT_RESET_TIME_MS = 3000;

    private OnShakeListener mListener;
```

cccccccccc

```
private long mShakeTimestamp;
 private int mShakeCount;

 public void setOnShakeListener(OnShakeListener listener) {
     this.mListener = listener;
 }

 public interface OnShakeListener {
     public void onShake(int count);
 }

 @Override
 public void onAccuracyChanged(Sensor sensor, int accuracy) {
     // ignore
 }

 @Override
 public void onSensorChanged(SensorEvent event) {

     if (mListener != null) {
         float x = event.values[0];
         float y = event.values[1];
         float z = event.values[2];

         float gX = x / SensorManager.GRAVITY_EARTH;
         float gY = y / SensorManager.GRAVITY_EARTH;
         float gZ = z / SensorManager.GRAVITY_EARTH;

         // gForce will be close to 1 when there is no movement.
         Float f = new Float(gX * gX + gY * gY + gZ * gZ);
         Double d = Math.sqrt(f.doubleValue());
         float gForce = d.floatValue();
```

```
            if (gForce > SHAKE_THRESHOLD_GRAVITY) {
                final long now = System.currentTimeMillis();
                // ignore shake events too close to each other (500ms)
                if (mShakeTimestamp + SHAKE_SLOP_TIME_MS > now) {
                    return;
                }

                // reset the shake count after 3 seconds of no shakes
                if (mShakeTimestamp + SHAKE_COUNT_RESET_TIME_MS < now) {
                    mShakeCount = 0;
                }

                mShakeTimestamp = now;
                mShakeCount++;

                mListener.onShake(mShakeCount);
            }
        }
    }
}
```

## 4.4 SensorService.java

source code:

```
package com.faiqua.sos.ShakeServices;

import android.annotation.SuppressLint;
import android.app.Notification;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.app.Service;
import android.content.Context;
import android.content.Intent;

cccccccccc
```

```
import android.hardware.Sensor;
import android.hardware.SensorManager;
import android.location.Location;
import android.os.Build;
import android.os.IBinder;
import android.os.VibrationEffect;
import android.os.Vibrator;
import android.telephony.SmsManager;
import android.util.Log;

import androidx.annotation.NonNull;
import androidx.annotation.RequiresApi;
import androidx.core.app.NotificationCompat;

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.tasks.CancellationToken;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.android.gms.tasks.OnTokenCanceledListener;
import com.faiqua.sos.Contacts.ContactModel;
import com.faiqua.sos.Contacts.DbHelper;
import com.faiqua.sos.R;

import java.util.List;

public class SensorService extends Service {

    private SensorManager mSensorManager;
    private Sensor mAccelerometer;
    private ShakeDetector mShakeDetector;

    public SensorService(){
    }

    @Override
    public IBinder onBind(Intent intent) {
        // TODO: Return the communication channel to the service.
        throw new UnsupportedOperationException("Not yet implemented");
    }

    @Override
    public int onStartCommand(Intent intent, int flags, int startId) {
        super.onStartCommand(intent, flags, startId);

        return START_STICKY;
    }
```

cccccccccc

```
 @Override
   public void onCreate() {
       super.onCreate();

       //start the foreground service
       if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O)
           startMyOwnForeground();
       else
           startForeground(1, new Notification());

       // ShakeDetector initialization
       mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
       mAccelerometer = mSensorManager
               .getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
       mShakeDetector = new ShakeDetector();
       mShakeDetector.setOnShakeListener(new ShakeDetector.OnShakeListener() {

           @SuppressLint("MissingPermission")
           @Override
           public void onShake(int count) {
               //check if the user has shaked the phone for 3 time in a row
               if(count==3) {
                 //vibrate the phone
                   vibrate();

                   //create FusedLocationProviderClient to get the user location
                   FusedLocationProviderClient fusedLocationClient =
LocationServices.getFusedLocationProviderClient(getApplicationContext());
                   //use the PRIORITY_BALANCED_POWER_ACCURACY so that the service
doesn't use unnecessary power via GPS
                   //it will only use GPS at this very moment

fusedLocationClient.getCurrentLocation(LocationRequest.PRIORITY_BALANCED_POWE
R_ACCURACY, new CancellationToken() {
                       @Override
                       public boolean isCancellationRequested() {
                           return false;
                       }
                       @NonNull
                       @Override
                       public CancellationToken onCanceledRequested(@NonNull
OnTokenCanceledListener onTokenCanceledListener) {
                           return null;
                       }
                   }).addOnSuccessListener(new OnSuccessListener<Location>() {
                       @Override
                       public void onSuccess(Location location) {
                           //check if location is null
```

cccccccccc

```
                    //for both the cases we will create different messages
                    if(location!=null){

                      //get the SMSManager
                      SmsManager smsManager = SmsManager.getDefault();
                       //get the list of all the contacts in Database
                       DbHelper db=new DbHelper(SensorService.this);
                       List<ContactModel> list=db.getAllContacts();
                       //send SMS to each contact
                       for(ContactModel c: list){
                          String message = "Hey, "+c.getName()+" I am in DANGER, i need
help. Please urgently reach me out. Here are my coordinates.\n
"+"http://maps.google.com/?q=" + location.getLatitude() + "," + location.getLongitude();
                          smsManager.sendTextMessage(c.getPhoneNo(), null, message, null,
null);
                       }
                    }else{
                       String message= " I am in DANGER, i need help. Please urgently reach
me out.\n"+"GPS was turned off.Couldn't find location. Call your nearest Police Station.";
                       SmsManager smsManager = SmsManager.getDefault();
                       DbHelper db=new DbHelper(SensorService.this);
                       List<ContactModel> list=db.getAllContacts();
                       for(ContactModel c: list){
                          smsManager.sendTextMessage(c.getPhoneNo(), null, message, null,
null);
                       }
                    }
                 }
              }).addOnFailureListener(new OnFailureListener() {
                 @Override
                 public void onFailure(@NonNull Exception e) {
                    Log.d("Check: ","OnFailure");
                    String message= "I am in DANGER, i need help. Please urgently reach me
out.\n"+"GPS was turned off.Couldn't find location. Call your nearest Police Station.";
                    SmsManager smsManager = SmsManager.getDefault();
                    DbHelper db=new DbHelper(SensorService.this);
                    List<ContactModel> list=db.getAllContacts();
                    for(ContactModel c: list){
                       smsManager.sendTextMessage(c.getPhoneNo(), null, message, null,
null);
                    } }
              });

           }
        }
     });

     //register the listener
```

```
        mSensorManager.registerListener(mShakeDetector, mAccelerometer,
SensorManager.SENSOR_DELAY_UI);
    }

    //method to vibrate the phone
    public void vibrate(){
        final Vibrator vibrator = (Vibrator)
getSystemService(Context.VIBRATOR_SERVICE);
        VibrationEffect vibEff;
        //Android Q and above have some predefined vibrating patterns
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.Q) {

vibEff=VibrationEffect.createPredefined(VibrationEffect.EFFECT_DOUBLE_CLICK);
            vibrator.cancel();
            vibrator.vibrate(vibEff);
        }else{
            vibrator.vibrate(500);
        }



    }

    //For Build versions higher than Android Oreo, we launch
    // a foreground service in a different way. This is due to the newly
    // implemented strict notification rules, which require us to identify
    // our own notification channel in order to view them correctly.
    @RequiresApi(Build.VERSION_CODES.O)
    private void startMyOwnForeground()
    {
        String NOTIFICATION_CHANNEL_ID = "example.permanence";
        String channelName = "Background Service";
        NotificationChannel chan = new
NotificationChannel(NOTIFICATION_CHANNEL_ID, channelName,
NotificationManager.IMPORTANCE_MIN);

        NotificationManager manager = (NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);
        assert manager != null;
        manager.createNotificationChannel(chan);

        NotificationCompat.Builder notificationBuilder = new NotificationCompat.Builder(this,
NOTIFICATION_CHANNEL_ID);
        Notification notification = notificationBuilder.setOngoing(true)
                .setContentTitle("You are protected.")
                .setContentText("We are there for you")

                //this is important, otherwise the notification will show the way
cccccccccc
```

```
   //you want i.e. it will show some default notification
            .setSmallIcon(R.drawable.ic_launcher_foreground)

            .setPriority(NotificationManager.IMPORTANCE_MIN)
            .setCategory(Notification.CATEGORY_SERVICE)
            .build();
      startForeground(2, notification);
   }


   @Override
   public void onDestroy() {

      //create an Intent to call the Broadcast receiver
      Intent broadcastIntent = new Intent();
      broadcastIntent.setAction("restartservice");
      broadcastIntent.setClass(this, ReactivateService.class);
      this.sendBroadcast(broadcastIntent);
      super.onDestroy();
   }

}
```

## 4.5 CustomAdapter.java

Source code:

```
package com.faiqua.sos.Contacts;


import android.content.Context;
import android.content.DialogInterface;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;


import androidx.annotation.NonNull;
cccccccccc
```

```
import com.google.android.material.dialog.MaterialAlertDialogBuilder;
import com.faiqua.sos.R;

import java.util.List;

public class CustomAdapter extends ArrayAdapter<ContactModel> {

    Context context;
    List<ContactModel> contacts;
    public CustomAdapter(@NonNull Context context, List<ContactModel> contacts) {
        super(context, 0, contacts);
        this.context=context;
        this.contacts=contacts;
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {

        //create a database helper object to handle the database manipulations
        DbHelper db=new DbHelper(context);

        // Get the data item for this position
        ContactModel c = getItem(position);
        // Check if an existing view is being reused, otherwise inflate the view
        if (convertView == null) {
            convertView = LayoutInflater.from(getContext()).inflate(R.layout.item_user, parent,
false);
        }

        LinearLayout linearLayout=(LinearLayout)convertView.findViewById(R.id.linear);

        // Lookup view for data population
```

cccccccccc

```
    TextView tvName = (TextView) convertView.findViewById(R.id.tvName);
    TextView tvPhone = (TextView) convertView.findViewById(R.id.tvPhone);
    // Populate the data into the template view using the data object
    tvName.setText(c.getName());
    tvPhone.setText(c.getPhoneNo());

    linearLayout.setOnLongClickListener(new View.OnLongClickListener() {
      @Override
      public boolean onLongClick(View view) {

        //generate an MaterialAlertDialog Box
        new MaterialAlertDialogBuilder(context)
            .setTitle("Remove Contact")
            .setMessage("Are you sure want to remove this contact?")
            .setPositiveButton("YES", new DialogInterface.OnClickListener() {
              @Override
              public void onClick(DialogInterface dialogInterface, int i) {
                //delete the specified contact from the database
                db.deleteContact(c);
                //remove the item from the list
                contacts.remove(c);
                //notify the listview that dataset has been changed
                notifyDataSetChanged();
                Toast.makeText(context, "Contact removed!",
Toast.LENGTH_SHORT).show();
              }
            })
            .setNegativeButton("NO", new DialogInterface.OnClickListener() {
              @Override
              public void onClick(DialogInterface dialogInterface, int i) {

              }
            })
cccccccccc
```

```
                    .show();
                return false;
            }
        });
        // Return the completed view to render on screen
        return convertView;
    }

    //this method will update the ListView
    public void refresh(List<ContactModel> list){
        contacts.clear();
        contacts.addAll(list);
        notifyDataSetChanged();
    }
}
```

## 4.6 DbHelper.java

Source code:

```
package com.faiqua.sos.Contacts;

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;
import java.util.List;

public class DbHelper extends SQLiteOpenHelper {
    // Database Version
    private static final int DATABASE_VERSION = 1;

    // Database Name
    private static final String DATABASE_NAME = "contactdata";
```

cccccccccc

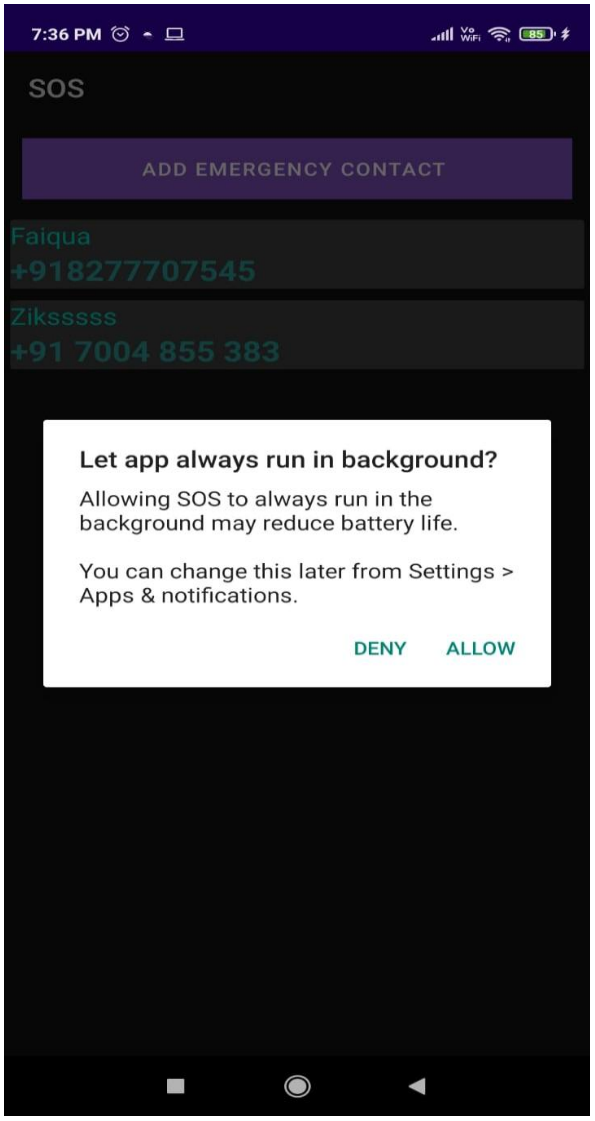
```
// Country table name
   private static final String TABLE_NAME= "contacts";

   // Country Table Columns names
   private static final String KEY_ID = "id";
   private static final String NAME = "Name";
   private static final String PHONENO = "PhoneNo";


   public DbHelper(Context context){
       super(context, DATABASE_NAME, null, DATABASE_VERSION);
   }

   @Override
   public void onCreate(SQLiteDatabase db) {

       //create the table for the first time
       String CREATE_COUNTRY_TABLE = "CREATE TABLE " + TABLE_NAME + "("
               + KEY_ID + " INTEGER PRIMARY KEY," + NAME + " TEXT,"
               + PHONENO + " TEXT" + ")";
       db.execSQL(CREATE_COUNTRY_TABLE);
   }

   @Override
   public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {

   }

   //method to add the contact
   public void addcontact(ContactModel contact){
       SQLiteDatabase db=this.getWritableDatabase();
       ContentValues c=new ContentValues();
       c.put(NAME,contact.getName());
       c.put(PHONENO,contact.getPhoneNo());
       db.insert(TABLE_NAME,null,c);
       db.close();
   }

   //method to retrieve all the contacts in List
   public List<ContactModel> getAllContacts(){
       List<ContactModel> list=new ArrayList<>();
       String query="SELECT * FROM "+TABLE_NAME;
       SQLiteDatabase db=this.getReadableDatabase();
       Cursor c=db.rawQuery(query,null);
       if(c.moveToFirst()) {
           do {
               list.add(new ContactModel(c.getInt(0),c.getString(1),c.getString(2)));

           } while (c.moveToNext());
       }
       return list;
   }

   //get the count of data, this will allow user to not add more that five contacts in database
   public int count(){
       int count=0;
       String query="SELECT COUNT(*) FROM "+TABLE_NAME;
       SQLiteDatabase db = this.getReadableDatabase();
       Cursor c=db.rawQuery(query,null);
       if(c.getCount()>0){
           c.moveToFirst();
           count=c.getInt(0);
       }
       c.close();
       return count;
   }

   // Deleting single country
   public void deleteContact(ContactModel contact) {
       SQLiteDatabase db = this.getWritableDatabase();
       int i=db.delete(TABLE_NAME,KEY_ID + " = ?",
               new String[] { String.valueOf(contact.getId()) });

       db.close();
```

# CHAPTER 5

# APPLICATION’S OUTPUT (FRONT END )

## 5.1 Application settings

**A**pp asks for the permission to run in the background.



Let app always run in background?

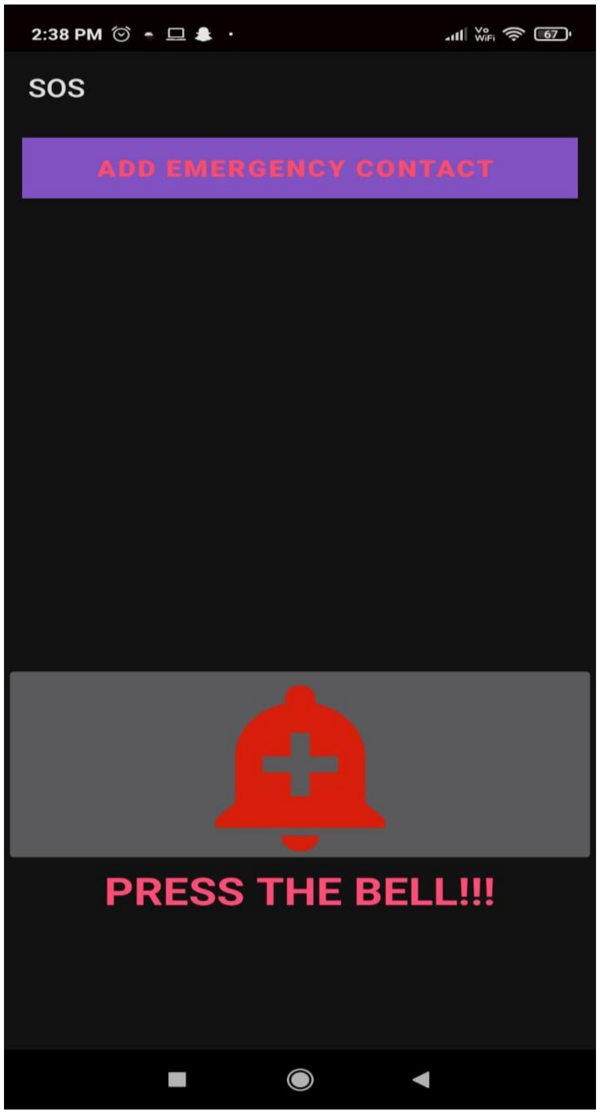
Allowing SOS to always run in the background may reduce battery life.

You can change this later from Settings > Apps & notifications.

**Snapshot 4.1: application setting**

## 5.2 Home Page

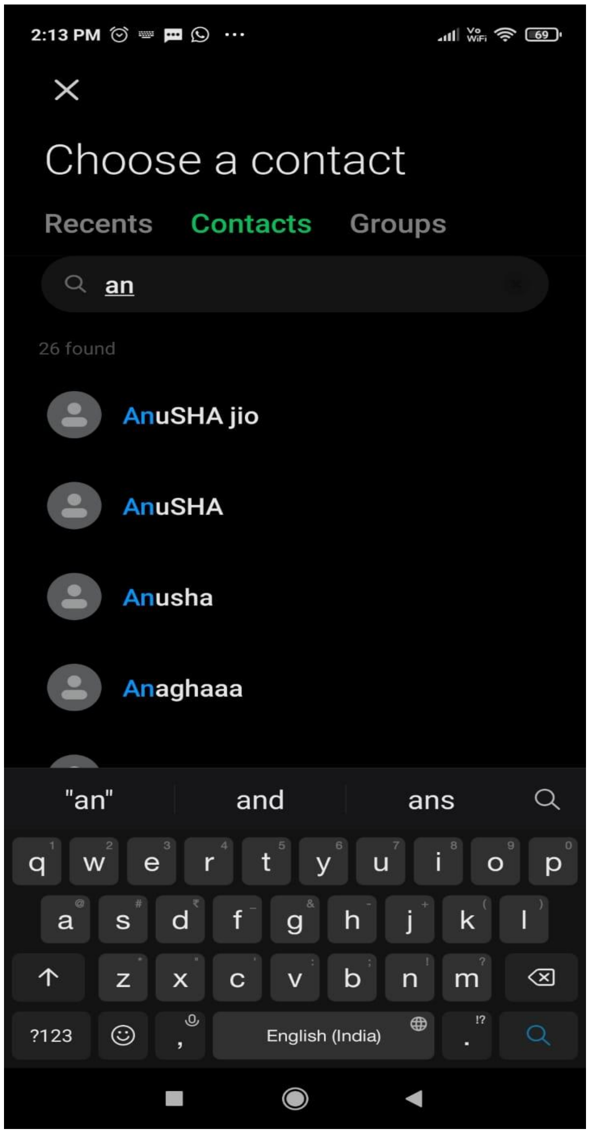
It displays the emergency contact button and alert button.



**Snapshot 4.2: Home screen**

## 5.3 Accessing Contact

After clicking the emergency button it takes us to the phone contact list from where we can select and add it to the emergency contact list.

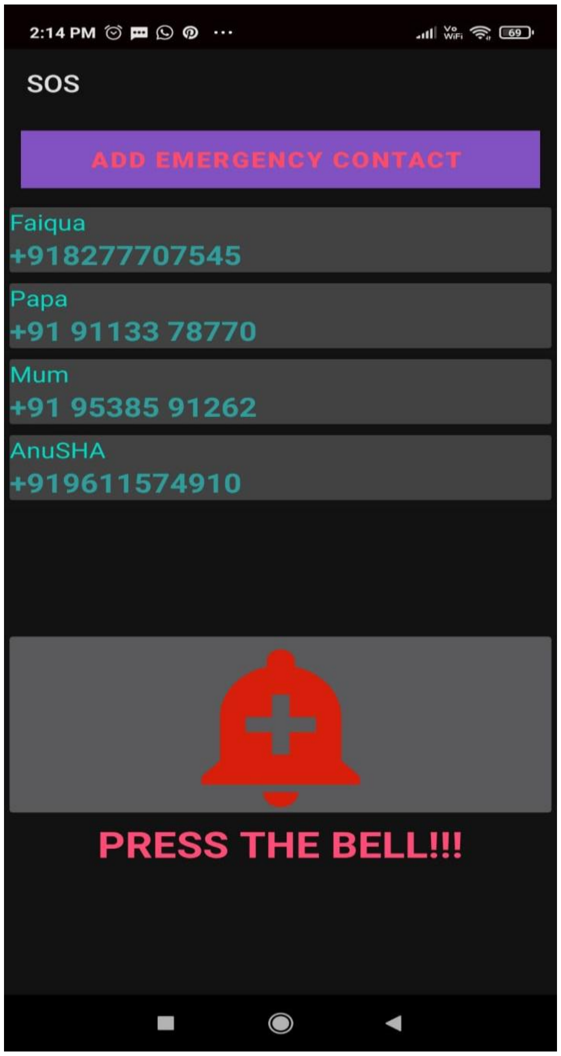


**Snapshot4.3: Accessing contact**

## 5.4 Contact Added

The emergency contact list is updated according to our priorities.

We can press the ALERT button so that we can seek attention from our surrounding.



Faiqua
+918277707545

Papa
+91 91133 78770

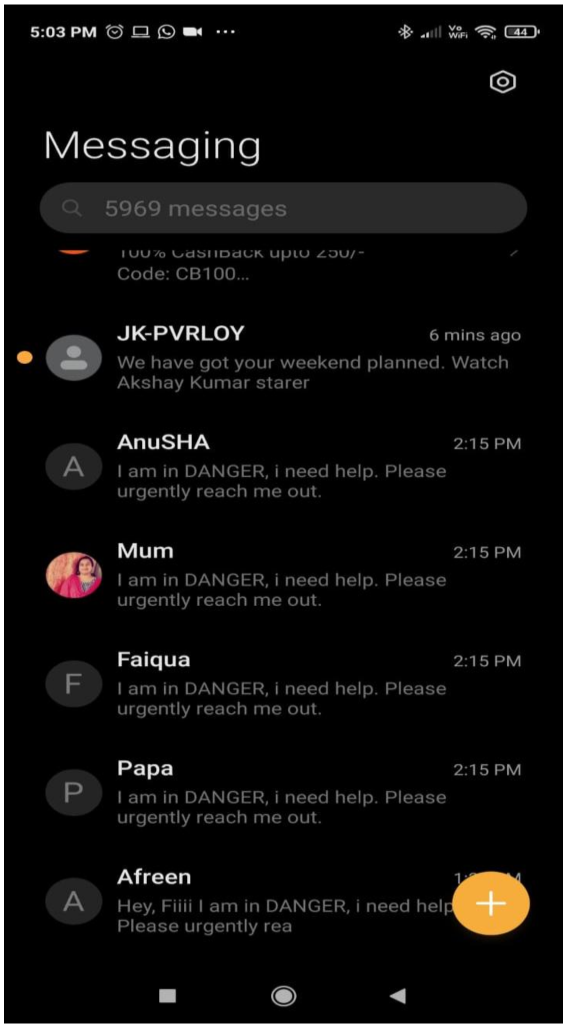
Mum
+91 95385 91262

AnuSHA
+919611574910

**Snapshot 5.4.1: Contact Added**

## 5.5 Sending of Message

When we shake the phone 3 consecutive times SOS message is sent to our emergency contact list


5:03 PM

Messaging

5969 messages

100% CashBack upto 250/-
Code: CB100...

JK-PVRLOY 6 mins ago
We have got your weekend planned. Watch Akshay Kumar starer

AnuSHA 2:15 PM
I am in DANGER, i need help. Please urgently reach me out.

Mum 2:15 PM
I am in DANGER, i need help. Please urgently reach me out.

Faiqua 2:15 PM
I am in DANGER, i need help. Please urgently reach me out.

Papa 2:15 PM
I am in DANGER, i need help. Please urgently reach me out.

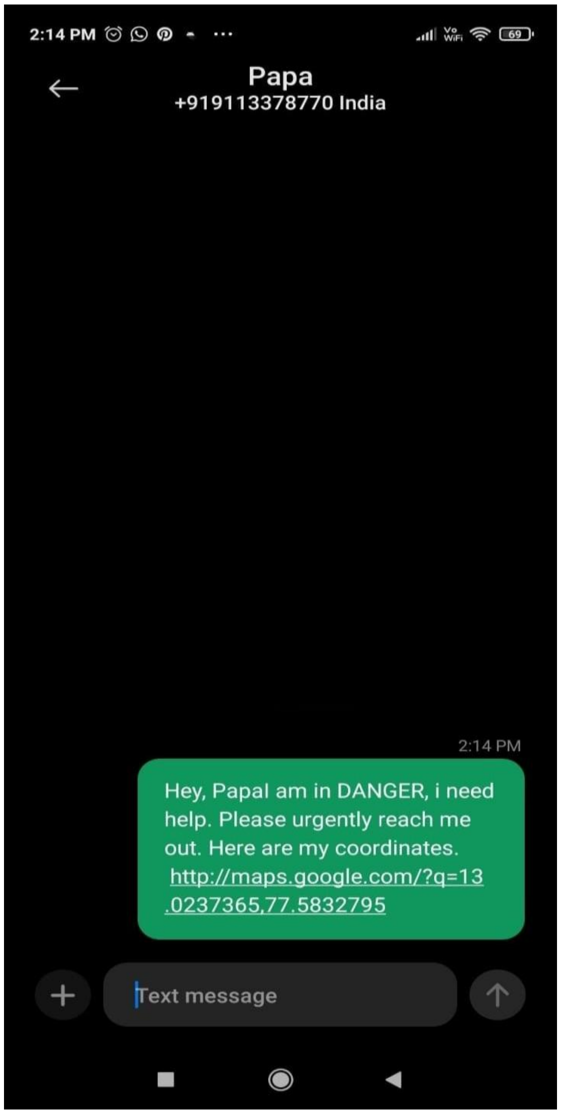
Afreen
Hey, Fiiii I am in DANGER, i need help
Please urgently rea

**Snapshot 5.5 Message sent**

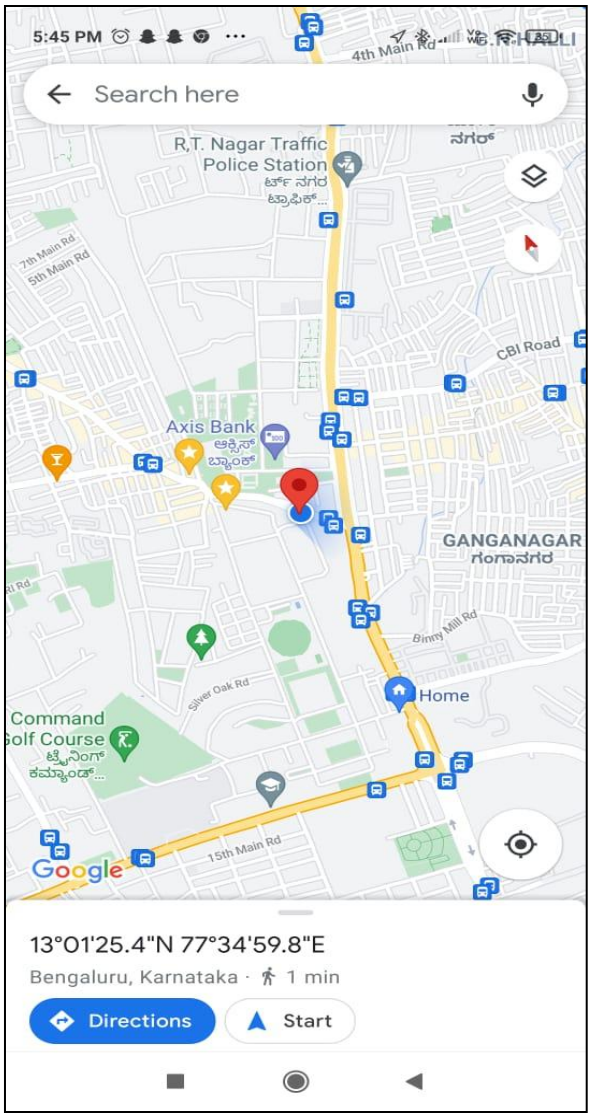
cccccccccc

### 5.5.1 Message with GPS ON

A message is sent to the contact with our current location and they can access it through google maps



Hey, PapaI am in DANGER, i need help. Please urgently reach me out. Here are my coordinates. http://maps.google.com/?q=13.0237365,77.5832795

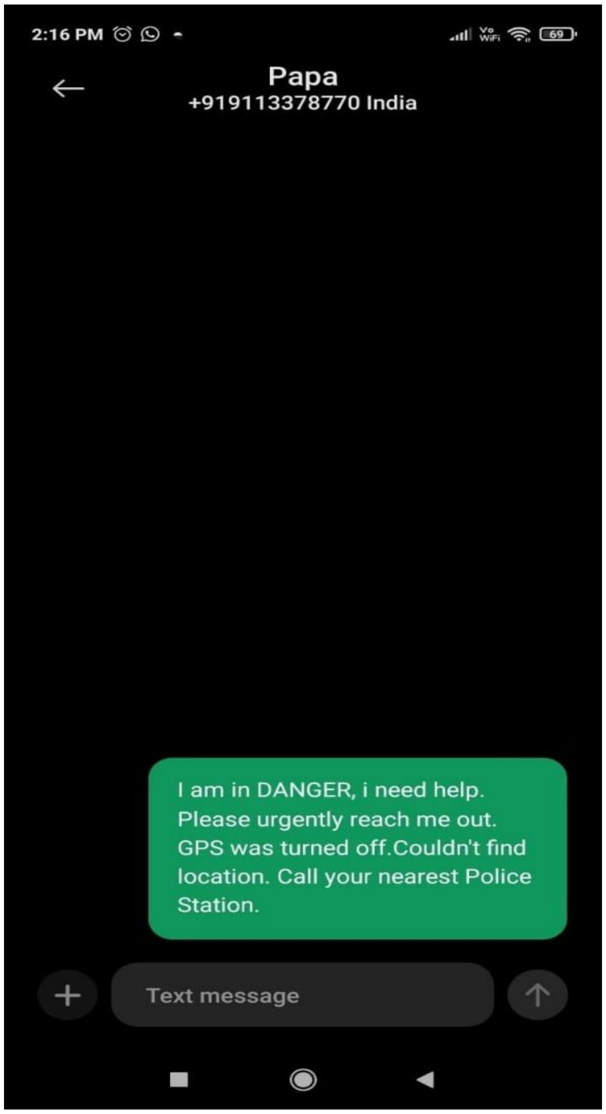


13°01'25.4"N 77°34'59.8"E
Bengaluru, Karnataka · 1 min

**Snapshot 5.5.1 message with GPS ON**

### 5.5.2 Message with GPS OFF

A message is sent to the contacts added without the location.

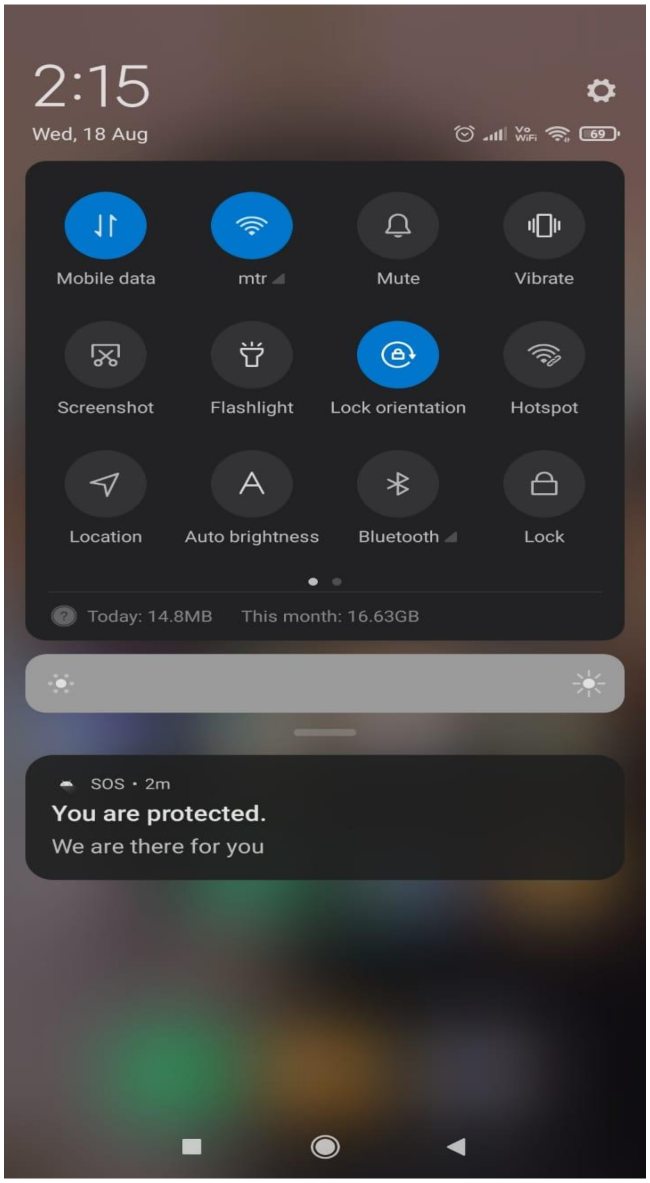


I am in DANGER, i need help. Please urgently reach me out. GPS was turned off.Couldn't find location. Call your nearest Police Station.

**Snapshot 5.5.2 message with GPS OFF**

## 5.6 App Running in the Background

A pop-up notification showing that the application is running in the background.



SOS • 2m
You are protected.
We are there for you

**Snapshot 5.6 pop-up Notification**

cccccccccc

# Chapter 6

# CONCLUSION

SOS is an essential app to have on a Smartphone. It is a personal security app that lets you send notifications to certain people via text messages in case of emergencies. The app also keeps a track of your current location so that you always known the address of where you are.

SOS app was my first attempt at an Android application. It gave me very good exposure to the Android platform and mobile development in general. The app enabled us to understand the basic of Android development and learning about Google Maps API for Android and performance testing the app.

## 6.1 Future Scope

The current work on the SOS app has a lot of essential features that would be used in case of an emergency situation like sending text messages within the app. An app for such a purpose has a lot of scope for enhancement. In the future the app may include features like –

- A home screen widget that can be used as a triggering point to send panic notifications. A user would then not have to open the app to “ALERT” button.
- Initiating a call to a number set from within the application when the user shakes the phone.
- The app can also listen to incoming messages from the set contacts. If these messages have a pre-defined text like “UPDATE LOCATION” the app can reply with a text message containing the current location or for some other text like “AUDIO”/” VIDEO”/” PHOTO” in which case the app can record a short audio/video or take a picture and send it to the person. This is very helpful as you may have already pressed the “ALERT” button and may be in some trouble where you cannot reply. This way the person can track you constantly and also understand something about the nature of the emergency from the audio clip.
- Setting up a password to stop the application.

cccccccccc

# BIBLIOGRAPHY

**Book Reference**:

1. Google Developer Training, "Android Developer Fundamentals Course – Concept Reference", Google Developer Training Team, 2017.

2. Erik Hellman, "Android Programming – Pushing the Limits", 1st Edition, Wiley India Pvt Ltd, 2014.

3. Dawn Griffiths and David Griffiths, "Head First Android Development", 1st Edition, O'Reilly SPD Publishers, 2015.

4. J F DiMarzio, "Beginning Android Programming with Android Studio", 4th Edition, Wiley India Pvt Ltd, 2016. ISBN-13: 978-8126565580

5. Anubhav Pradhan, Anil V Deshpande, "Composing Mobile Apps" using Android, Wiley 2014, ISBN: 978-81-265-4660-2

**Online References**:

- www.stackoverflow.com
- www.youtube.com
- https://developer.android.com/studio
- www.google.com