**Project Report**

**Alarm System**

*using Proximity Sensor*

**Prepared by:**

Akash Rajendra Ventekar

Sneha Dipankar Roy

**Prepared for:**

Dr. Feng Zhu

*CS 670 Computer Networks*

**Overview:**

**Project Description:**

A security alarm is a system designed to detect intrusion – unauthorized entry – into a building or area. Security alarms are used in residential, commercial, industrial, and military properties for protection against burglary (theft) or property damage, as well as personal protection against intruders. Car alarms likewise protect vehicles and their contents. Prisons also use security systems for control of inmates.

We will be using the proximity sensor of a smartphone to detect the presence of any person close to the device. The smartphone will be strategically placed at a point around which we intend to detect intrusion. The proximity sensor readings will be sent to another device. This device will process the information received and take appropriate actions. The action would be an alarm in the form of a ringtone.

**Sensor:**

Proximity Sensor - A proximity sensor is a [sensor](https://en.wikipedia.org/wiki/Sensor) able to detect the presence of nearby objects without any physical contact.

**Platform:**

Java

**Operating System:**

Android

**Tools to be used:**

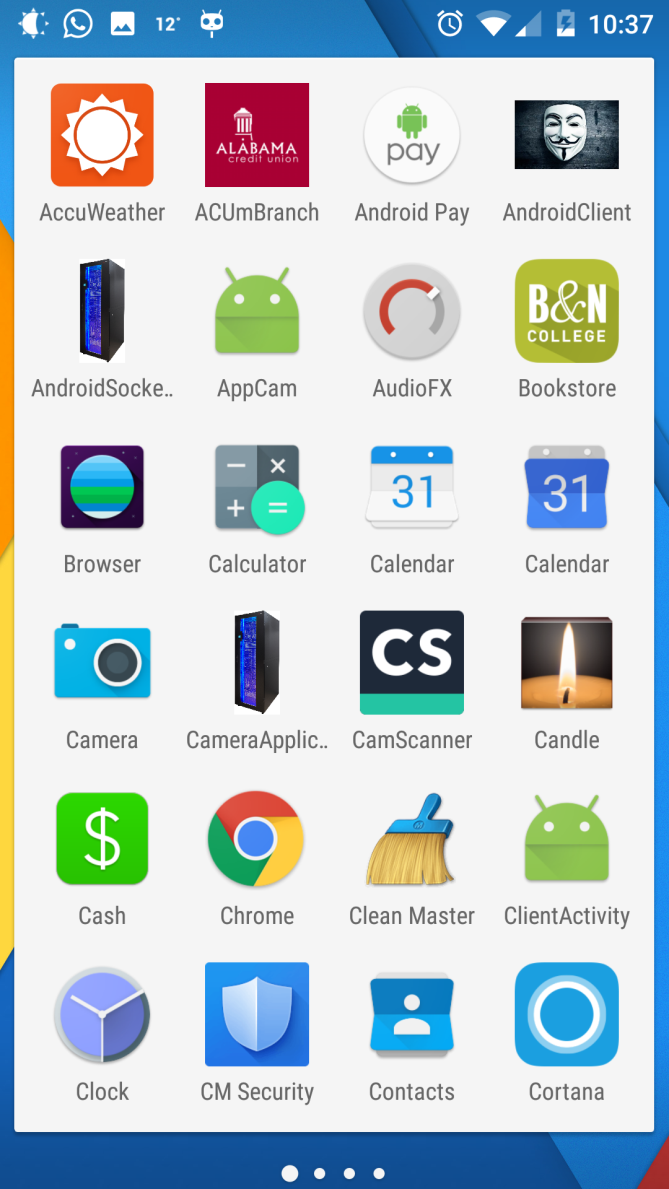
Android Studio

**Screenshots:**

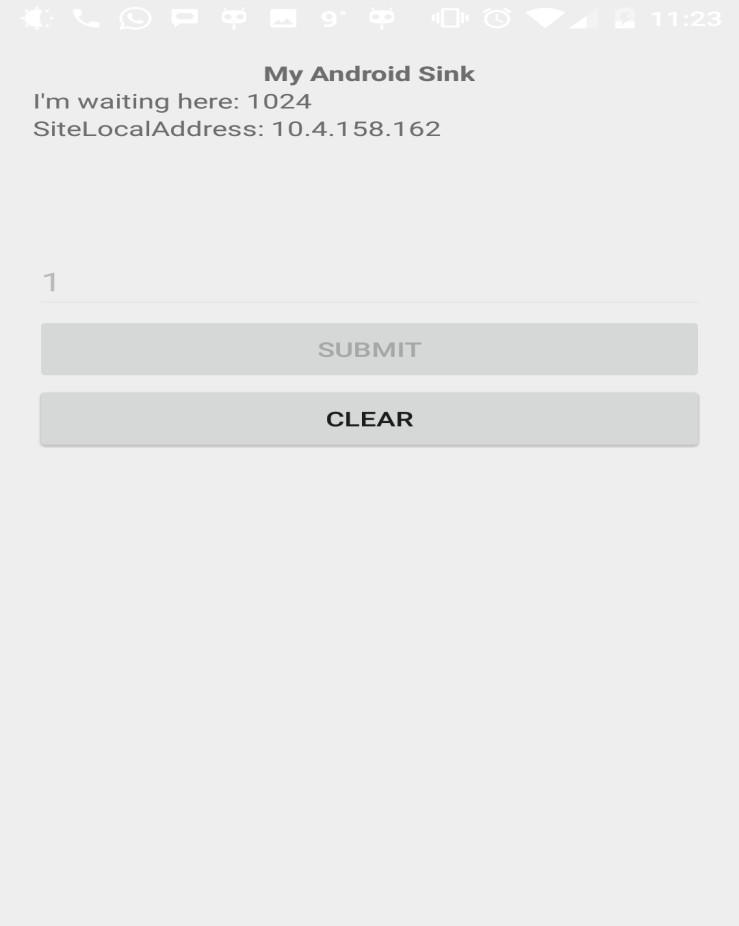
**Application Icon:**

**** ****

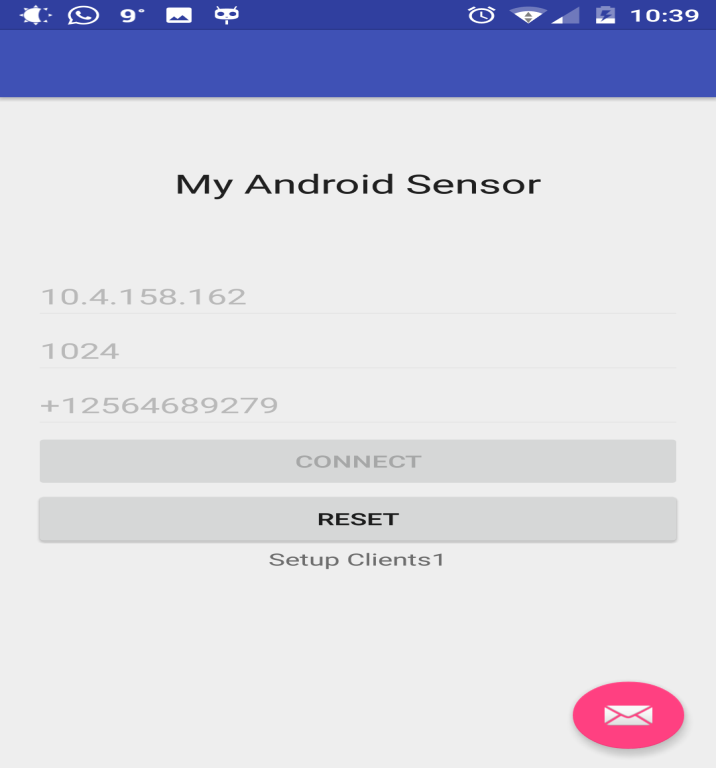
**Home Screen:**

****

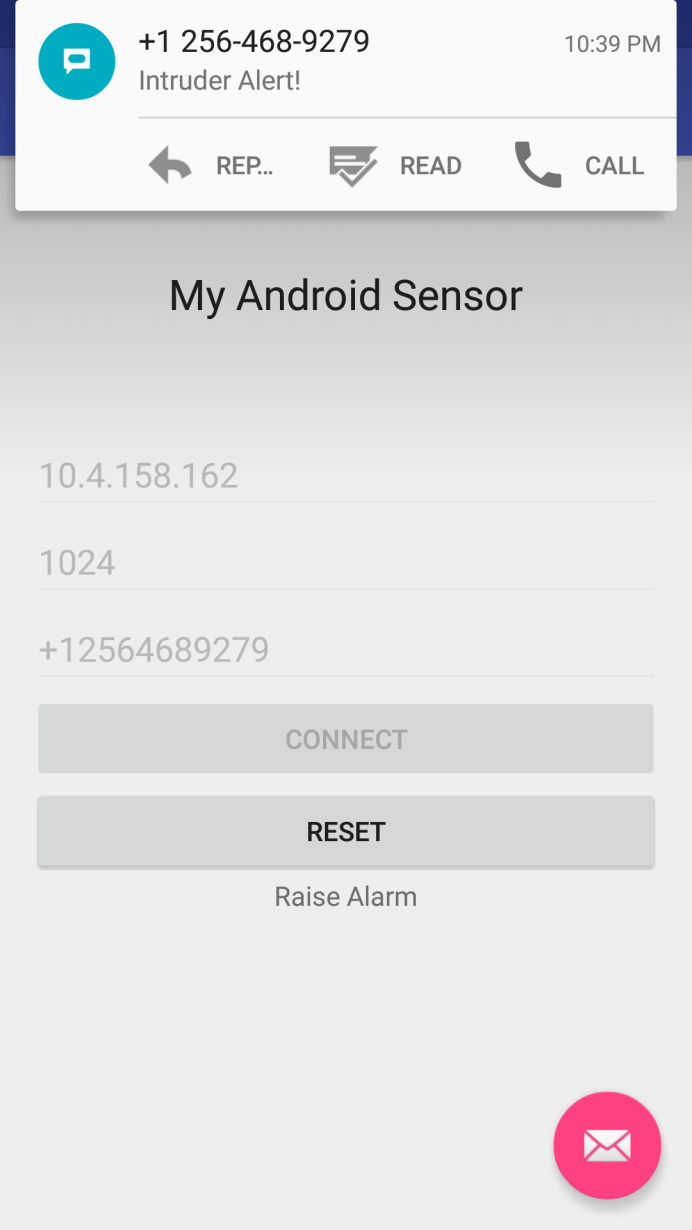
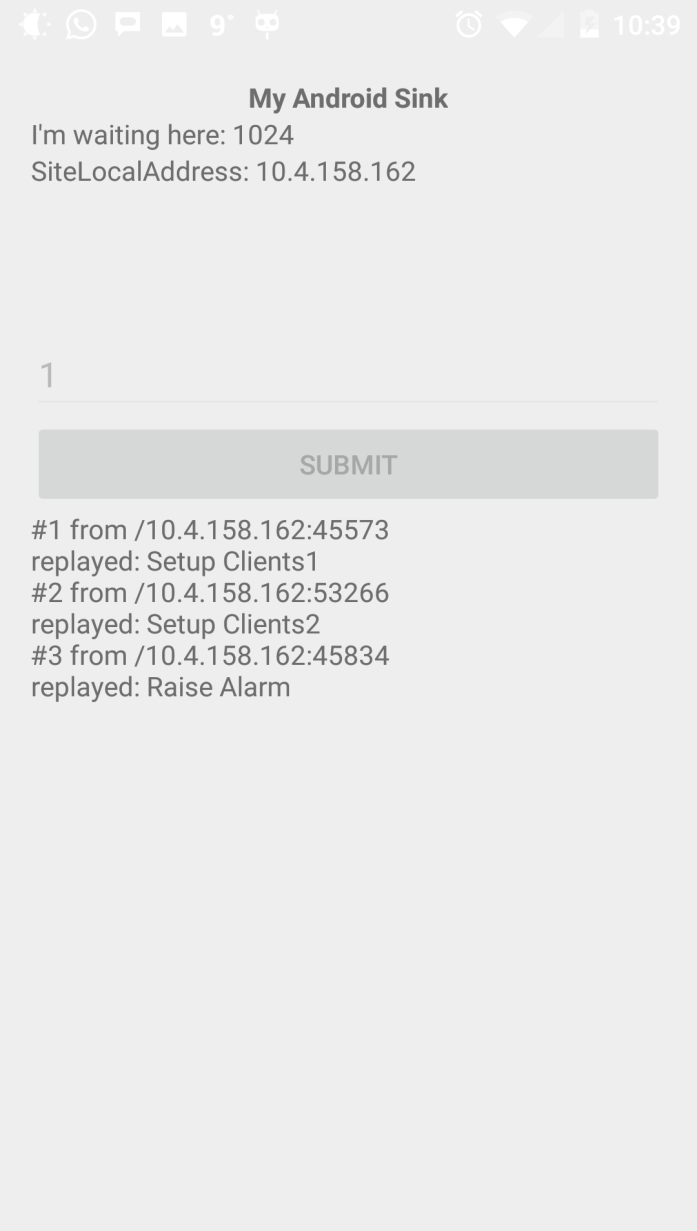
**Android Server Layout:**

****

**Android Client Layout:**

****

**After Detection:**

**** ****

**Source code for Android Client:**

**AndroidManifest.xml:** Manifest file to provide the required permissions

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.example.akash.androidclient">  
 *<!-- Provide permissions for socket programming and sending notifications-->* <uses-permission android:name="android.permission.INTERNET"/>  
 <uses-permission android:name="android.permission.SEND\_SMS"/>  
 <application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme">  
 <activity  
 android:name=".AndroidClient"  
 android:label="@string/app\_name"  
 android:theme="@style/AppTheme.NoActionBar">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 </application>  
  
</manifest>

**activity\_android\_client.xml:** Layout of the screen

<?xml version="1.0" encoding="utf-8"?>  
<android.support.design.widget.CoordinatorLayout 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:fitsSystemWindows="true"  
 tools:context="com.example.akash.androidclient.AndroidClient">  
  
 <android.support.design.widget.AppBarLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:theme="@style/AppTheme.AppBarOverlay">  
  
 <android.support.v7.widget.Toolbar  
 android:id="@+id/toolbar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="?attr/actionBarSize"  
 android:background="?attr/colorPrimary"  
 app:popupTheme="@style/AppTheme.PopupOverlay" />  
  
 </android.support.design.widget.AppBarLayout>  
  
 <include layout="@layout/content\_android\_client" />  
  
 <android.support.design.widget.FloatingActionButton  
 android:id="@+id/fab"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="bottom|end"  
 android:layout\_margin="@dimen/fab\_margin"  
 android:src="@android:drawable/ic\_dialog\_email" />  
  
</android.support.design.widget.CoordinatorLayout>

**content\_android\_client.xml:** Layout of the screen

<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:paddingBottom="@dimen/activity\_vertical\_margin"  
 android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 android:orientation="vertical"  
 tools:context=".MainActivity"  
 android:weightSum="1">  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/textView2"  
 android:layout\_weight="0.37" />  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:id="@+id/textView3"  
 android:layout\_weight="0.37" />  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textAppearance="?android:attr/textAppearanceLarge"  
 android:text="My Android Sensor"  
 android:id="@+id/textView"  
 android:layout\_weight="0.37"  
 android:layout\_marginTop="78dp"  
 android:layout\_below="@+id/textView2"  
 android:layout\_centerHorizontal="true" />  
  
  
  
 <EditText  
 android:id="@+id/address"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Sink IP address"  
 android:layout\_below="@+id/textView"  
 android:layout\_toRightOf="@+id/textView2"  
 android:layout\_toEndOf="@+id/textView2"  
 android:layout\_marginTop="57dp" />  
  
 <EditText  
 android:id="@+id/port"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Sink Port"  
 android:layout\_below="@+id/address"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true" />  
  
 <EditText  
 android:id="@+id/number"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Enter Phone Number with country code"  
 android:layout\_below="@+id/port"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true" />  
  
 <Button  
 android:id="@+id/connect"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Connect"  
 android:layout\_below="@+id/number"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true" />  
  
 <Button  
 android:id="@+id/clear"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Reset"  
 android:layout\_below="@+id/connect"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true" />  
  
 <TextView  
 android:id="@+id/response"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Response from Sink"  
 android:layout\_weight="0.09"  
 android:layout\_gravity="center\_horizontal"  
 android:layout\_below="@+id/clear"  
 android:layout\_centerHorizontal="true" />  
  
</RelativeLayout>

**AndroidClient.java:** Source code for sensing, socket programming and actions to take after server reply

package com.example.akash.androidclient;  
  
import java.io.ByteArrayOutputStream;  
import java.io.IOException;  
import java.io.InputStream;  
import java.net.Socket;  
import java.net.UnknownHostException;  
  
import android.content.Intent;  
import android.hardware.Sensor;  
import android.hardware.SensorEvent;  
import android.hardware.SensorEventListener;  
import android.hardware.SensorManager;  
import android.media.MediaPlayer;  
import android.net.Uri;  
import android.os.AsyncTask;  
import android.os.Bundle;  
import android.app.Activity;  
import android.telephony.SmsManager;  
import android.util.Log;  
import android.view.View;  
import android.view.View.OnClickListener;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.TextView;  
import android.widget.Toast;  
  
/\*  
 // This is the Android Sensor  
 //@Author: Akash Rajendra Ventekar, Sneha Dipankar Roy  
 \*/  
public class AndroidClient extends Activity implements SensorEventListener {  
  
 /\*Declaration section\*/  
 //Layout variables  
 TextView textResponse;  
 EditText editTextAddress, editTextPort;  
 Button buttonConnect, buttonClear;  
  
 //Sensor variables  
 private SensorManager mSensorManager;  
 private Sensor mSensor;  
  
 //Socket variable  
 Socket socket;  
  
 //Flag not to send data to sink  
 int flag=0;  
  
 //Variables for sending SMS  
 SmsManager smsManager;  
 EditText Number;  
 String number;  
  
 //variable for playing media  
 MediaPlayer mp;  
  
 /\*  
 Overridden methods of the class Sensor Event Listener  
 \*/  
 public void onAccuracyChanged(Sensor sensor, int accuracy) {  
 }  
  
 //Detect intruder or proximity sensor  
 public void onSensorChanged(SensorEvent event) {  
 System.*out*.println("Sensor value is :" + event.values[0]);  
  
 //Send the values to the sink if the IPaddress and port are retrieved from the text fields  
 if(flag==1) {  
  
 MyClientTask myClientTask = new MyClientTask(  
 editTextAddress.getText().toString(),  
 Integer.*parseInt*(editTextPort.getText().toString()));  
  
 myClientTask.execute();  
  
  
 }  
 }  
 protected void onPause() {  
 super.onPause();  
 mSensorManager.unregisterListener(this);  
 }  
  
 protected void onResume() {  
 super.onResume();  
 mSensorManager.registerListener(this, mSensor,  
 SensorManager.*SENSOR\_DELAY\_NORMAL*);  
 }  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 //Set layout as activity\_android\_client.xml  
 setContentView(R.layout.*activity\_android\_client*);  
  
 //Initialize Media player with Alarm sound  
 mp=MediaPlayer.*create*(AndroidClient.this, R.raw.*school*);  
  
 //Get individual ID of the Layout elements  
 editTextAddress = (EditText)findViewById(R.id.*address*);  
 editTextPort = (EditText)findViewById(R.id.*port*);  
 buttonConnect = (Button)findViewById(R.id.*connect*);  
 buttonClear = (Button)findViewById(R.id.*clear*);  
 textResponse = (TextView)findViewById(R.id.*response*);  
 Number = (EditText)findViewById(R.id.*number*);  
 buttonConnect.setOnClickListener(buttonConnectOnClickListener);  
  
 //Listener for Clear or Reset Button  
 buttonClear.setOnClickListener(new OnClickListener() {  
  
 @Override  
 public void onClick(View v) {  
 textResponse.setText("");  
 //Pause the alarm sound  
 mp.pause();  
 }  
 });  
  
 //Declare the sensor variables to use proximity sensor  
 mSensorManager = (SensorManager) getSystemService(*SENSOR\_SERVICE*);  
 mSensor = mSensorManager.getDefaultSensor(Sensor.*TYPE\_PROXIMITY*);  
  
 //Declare the sms variables  
 smsManager = SmsManager.*getDefault*();  
  
 }  
  
 //Listener for connect button  
 OnClickListener buttonConnectOnClickListener =  
 new OnClickListener(){  
  
 @Override  
 public void onClick(View arg0) {  
 //Create a socket  
 MyClientTask myClientTask = new MyClientTask(  
 editTextAddress.getText().toString(),  
 Integer.*parseInt*(editTextPort.getText().toString()));  
  
 //Extract the phone number  
 number=Number.getText().toString();  
 if(Number.getText().equals(null))  
 number="+12564689279";  
  
 //Disable few of the layout elements  
 buttonConnect.setEnabled(false);  
 editTextAddress.setEnabled(false);  
 editTextPort.setEnabled(false);  
 Number.setEnabled(false);  
  
 //Connect to the server  
 myClientTask.execute();  
 }};  
  
 public class MyClientTask extends AsyncTask<Void, Void, Void> {  
  
 //Initialization of class variables  
 String dstAddress;  
 int dstPort;  
 String response = "";  
  
 //Set the IP address and port  
 MyClientTask(String addr, int port){  
 dstAddress = addr;  
 dstPort = port;  
 }  
  
 @Override  
 protected Void doInBackground(Void... arg0) {  
  
 socket = null;  
  
 try {  
 //Create Socket  
 socket = new Socket(dstAddress, dstPort);  
  
 ByteArrayOutputStream byteArrayOutputStream =  
 new ByteArrayOutputStream(1024);  
 byte[] buffer = new byte[1024];  
  
 int bytesRead;  
 InputStream inputStream = socket.getInputStream();  
  
  
 //Blocking receive  
 while ((bytesRead = inputStream.read(buffer)) != -1){  
  
 byteArrayOutputStream.write(buffer, 0, bytesRead);  
 response += byteArrayOutputStream.toString("UTF-8");  
 System.*out*.println("Buffer" +response);  
 //Raise alarm after setup phase  
 if(response.equals("Raise Alarm")){  
  
 mp.start();  
 smsManager.sendTextMessage(number, null, "Intruder Alert!", null, null);  
  
 }  
 }  
  
 } catch (UnknownHostException e) {  
 // *TODO Auto-generated catch block* e.printStackTrace();  
 response = "UnknownHostException: " + e.toString();  
 } catch (IOException e) {  
 // *TODO Auto-generated catch block* e.printStackTrace();  
 response = "IOException: " + e.toString();  
 }finally{  
 if(socket != null){  
 try {  
 socket.close();  
 } catch (IOException e) {  
 // *TODO Auto-generated catch block* e.printStackTrace();  
 }  
 }  
 }  
 return null;  
 }  
  
 @Override  
 protected void onPostExecute(Void result) {  
 textResponse.setText(response);  
 super.onPostExecute(result);  
 flag=1;  
 }  
  
  
  
  
 }  
  
}

**Source code for Android Socket Server:**

**AndroidManifest.xml:** Manifest file to provide the required permissions

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.example.akash.androidsocketserver">  
 <uses-permission android:name="android.permission.INTERNET"/>  
 <application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme">  
 <activity  
 android:name=".AndroidServer"  
 android:label="@string/app\_name"  
 android:theme="@style/AppTheme.NoActionBar">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
 </application>  
  
</manifest>

**content\_android\_server.xml:** Layout of the screen

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:paddingBottom="@dimen/activity\_vertical\_margin"  
 android:paddingLeft="@dimen/activity\_horizontal\_margin"  
 android:paddingRight="@dimen/activity\_horizontal\_margin"  
 android:paddingTop="@dimen/activity\_vertical\_margin"  
 tools:context=".MainActivity" >  
  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center\_horizontal"  
 android:autoLink="web"  
 android:text="My Android Sink"  
 android:textStyle="bold" />  
  
 <TextView  
 android:id="@+id/info"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content" />  
 <TextView  
 android:id="@+id/infoip"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content" />  
 <EditText  
 android:id="@+id/editClient"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Number of clients"  
 android:layout\_below="@+id/textView"  
 android:layout\_toRightOf="@+id/textView2"  
 android:layout\_toEndOf="@+id/textView2"  
 android:layout\_marginTop="57dp" />  
  
 <Button  
 android:id="@+id/buttonSubmit"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Submit"  
 android:layout\_below="@+id/number"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true" />  
  
 <Button  
 android:id="@+id/buttonClear"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Clear"  
 android:layout\_below="@+id/connect"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true" />  
  
 <ScrollView  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent" >  
  
 <TextView  
 android:id="@+id/msg"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content" />  
 </ScrollView>  
  
</LinearLayout>

**AndroidServer.java:** Server program to accept connection from client, process information from client and send actions to take by the client

package com.example.akash.androidsocketserver;  
  
import java.io.ByteArrayOutputStream;  
import java.io.IOException;  
import java.io.InputStream;  
import java.io.OutputStream;  
import java.io.PrintStream;  
import java.net.InetAddress;  
import java.net.NetworkInterface;  
import java.net.ServerSocket;  
import java.net.Socket;  
import java.net.SocketException;  
import java.util.Enumeration;  
import java.util.logging.Logger;  
  
import android.media.MediaPlayer;  
import android.os.Bundle;  
import android.app.Activity;  
import android.provider.MediaStore;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.TextView;  
  
/\*  
 // This is the Android Sink  
 //@Author: Akash Rajendra Ventekar, Sneha Dipankar Roy  
 \*/  
  
public class AndroidServer extends Activity {  
  
 //Declaration section  
  
 //Layout Elements declaration  
 TextView info, infoip, msg;  
 String message = "";  
 EditText client;  
 private Button buttonSubmit;  
 private String clientCount;  
 private Button buttonClear;  
  
 //Declare socket variable  
 ServerSocket serverSocket;  
  
 //Declare media player variable  
 MediaPlayer mp;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 //Set layout as content\_android\_Server.xml  
 setContentView(R.layout.*content\_android\_server*);  
  
 //Extract ID from the layout  
 info = (TextView) findViewById(R.id.*info*);  
 infoip = (TextView) findViewById(R.id.*infoip*);  
 msg = (TextView) findViewById(R.id.*msg*);  
 client=(EditText) findViewById(R.id.*editClient*);  
 buttonSubmit=(Button) findViewById(R.id.*buttonSubmit*);  
 buttonClear=(Button) findViewById(R.id.*buttonClear*);  
  
 //Initialize media player  
 mp=MediaPlayer.*create*(AndroidServer.this, R.raw.*school*);  
  
 //Set the IP address of the server in the plaintext view  
 infoip.setText(getIpAddress());  
  
 //Listener for submit button  
 buttonSubmit.setOnClickListener(new View.OnClickListener() {  
  
 @Override  
 public void onClick(View v) {  
 clientCount=client.getText().toString();  
 client.setEnabled(false);  
 buttonSubmit.setEnabled(false);  
 }  
 });  
  
 //Listener for Clear button  
 buttonClear.setOnClickListener(new View.OnClickListener() {  
  
 @Override  
 public void onClick(View v) {  
  
 mp.pause();  
 }  
 });  
  
 //Create socket and start listening  
 Thread socketServerThread = new Thread(new SocketServerThread());  
 socketServerThread.start();  
 }  
  
 @Override  
 protected void onDestroy() {  
 super.onDestroy();  
  
  
 }  
  
 private class SocketServerThread extends Thread {  
  
 static final int *SocketServerPORT* = 1024;  
 int count = 0;  
  
 @Override  
 public void run() {  
 try {  
 //Create socket  
 serverSocket = new ServerSocket(*SocketServerPORT*);  
  
 //Update the port on the layout  
 AndroidServer.this.runOnUiThread(new Runnable() {  
  
 @Override  
 public void run() {  
 info.setText("I'm waiting here: "  
 + serverSocket.getLocalPort());  
 }  
 });  
  
 while (true) {  
 //Accept connection from client  
 Socket socket = serverSocket.accept();  
 count++;  
 message += "#" + count + " from " + socket.getInetAddress()  
 + ":" + socket.getPort() + "\n";  
  
 //Set the text view with the client IP and the port from which it is sending message  
 AndroidServer.this.runOnUiThread(new Runnable() {  
  
 @Override  
 public void run() {  
 msg.setText(message);  
 }  
 });  
  
 //Send reply to the client  
 SocketServerReplyThread socketServerReplyThread = new SocketServerReplyThread(  
 socket, count);  
 socketServerReplyThread.run();  
  
 }  
 } catch (IOException e) {  
 // *TODO Auto-generated catch block* e.printStackTrace();  
 }  
 }  
  
 }  
  
 private class SocketServerReplyThread extends Thread {  
  
 private Socket hostThreadSocket;  
 int cnt;  
  
 SocketServerReplyThread(Socket socket, int c) {  
 hostThreadSocket = socket;  
 cnt = c;  
 }  
  
 @Override  
 public void run() {  
 OutputStream outputStream;  
 InputStream inputStream;  
 String msgReply = "Setup Clients" + cnt;  
  
 try {  
 outputStream = hostThreadSocket.getOutputStream();  
 inputStream = hostThreadSocket.getInputStream();  
 PrintStream printStream = new PrintStream(outputStream);  
  
 //Send "Raise Alarm" to client after the setup phase  
 if(cnt>Integer.*parseInt*(clientCount)+1){  
 msgReply = "Raise Alarm";  
 printStream.print(msgReply);  
 mp.start();  
  
 }  
 else  
 printStream.print(msgReply);  
 printStream.close();  
  
 message += "replayed: " + msgReply + "\n";  
 //Set text message on the actions  
 AndroidServer.this.runOnUiThread(new Runnable() {  
  
 @Override  
 public void run() {  
 msg.setText(message);  
 }  
 });  
  
 } catch (IOException e) {  
 // *TODO Auto-generated catch block* e.printStackTrace();  
 message += "Something wrong! " + e.toString() + "\n";  
 }  
  
 AndroidServer.this.runOnUiThread(new Runnable() {  
  
 @Override  
 public void run() {  
 msg.setText(message);  
 }  
 });  
 }  
  
 }  
  
 //Return the IP address of the server  
 private String getIpAddress() {  
 String ip = "";  
 try {  
 //Get all interfaces of the server  
 Enumeration<NetworkInterface> enumNetworkInterfaces = NetworkInterface  
 .*getNetworkInterfaces*();  
 while (enumNetworkInterfaces.hasMoreElements()) {  
 NetworkInterface networkInterface = enumNetworkInterfaces  
 .nextElement();  
 //Get the enumeration of IP address of the server  
 Enumeration<InetAddress> enumInetAddress = networkInterface  
 .getInetAddresses();  
 while (enumInetAddress.hasMoreElements()) {  
 InetAddress inetAddress = enumInetAddress.nextElement();  
  
 if (inetAddress.isSiteLocalAddress()) {  
 ip += "SiteLocalAddress: "  
 + inetAddress.getHostAddress() + "\n";  
 }  
  
 }  
  
 }  
  
 } catch (SocketException e) {  
 // *TODO Auto-generated catch block* e.printStackTrace();  
 ip += "Something Wrong! " + e.toString() + "\n";  
 }  
  
 return ip;  
 }  
}