Code Inspection Document

for

'*RING ME*

– *A Mobile management application*' Phase III requirements

Prepared by: Team Mode Changer (Venkata Vikas Chirumamilla, Chenchu Sai Krishna Kolli, Siri Gogineni, Revanth Reddy Malreddy, Sai Teja Malle)

1. Structure of Android code

To understand the code flow, one must understand how Android is built. Each android application is associated with screens which is built using XML code. This GUI is interlinked to work with server using java code at the backend. IDE builds/generates .apk file for the application using gradle build. This .apk file needs to install/deployed on the android phone and can be used.

1.1. Execution flow:

1. All resource files are combined together by AAP[Android Asset Packing Tool]. Resource files are like audio video images other asset related files. 2.Java files converted into .class files by JVM.So, the out of the jvm will be .class files, that are heavy weight to put into android. So, that one more level of process will be taken place.
2. So, the .Class files are entered as input to DX tool. Basically, this is a tool which will convert .class files to .dex files. That mean Dalvik executable file. Those files are eligible to execute on DVM (Dalvik Virtual Machine)
3. After getting .dex files, packed them APK builder. Which is basically, Application Packaging. So, this packed files kept into devices and that will be executed by DVM.

The below figure 1.1. explains about the android code execution flow.

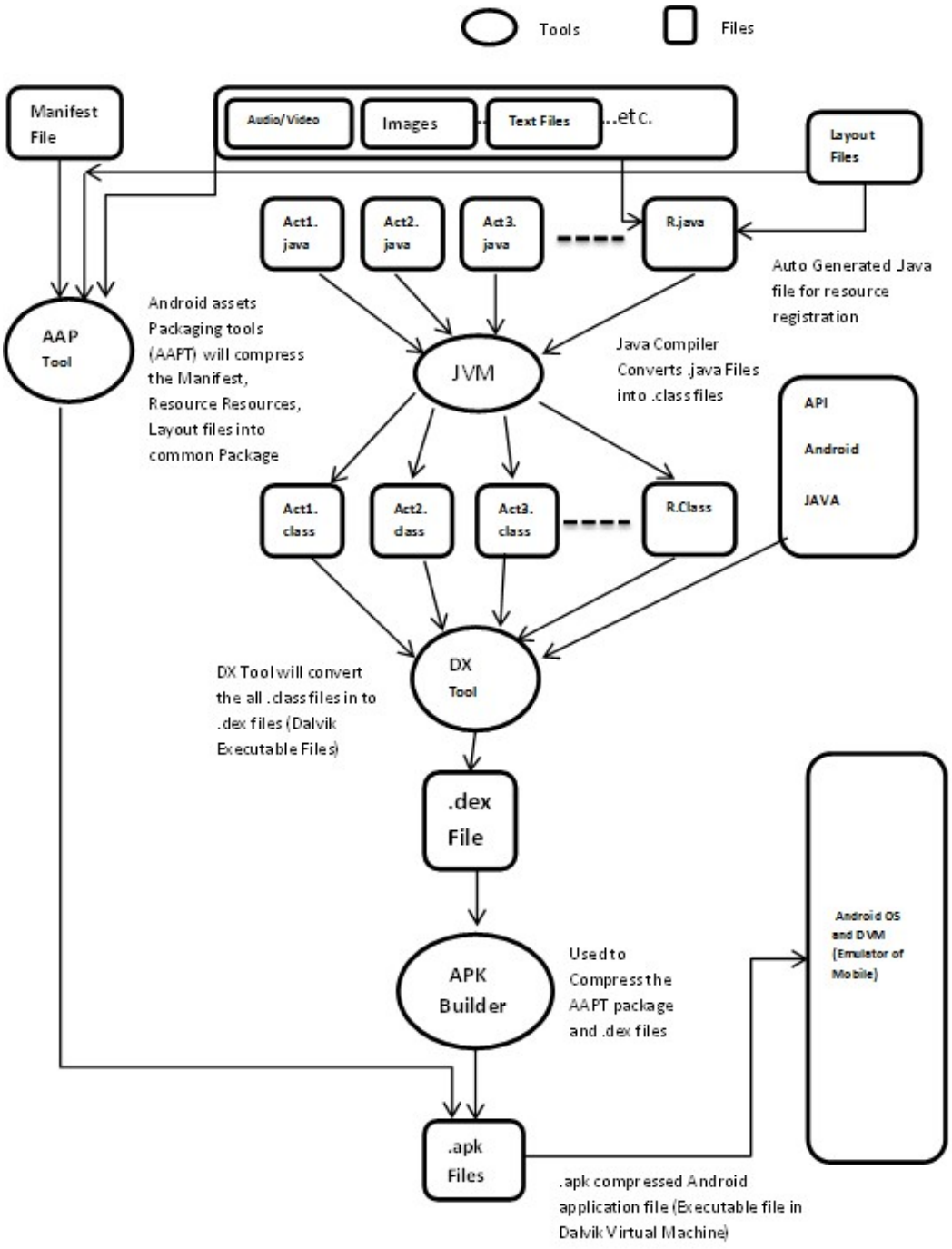

Figure 1.1. Android code execution flow

[Figure 1.1. Reference: https://stackoverflow.com/questions/5749436/android-application-control-flow]

2. Source code

2.1. AndroidManifest.xml

The **manifest** presents essential information about the application to the **Android** system, information the system must have before it can run any of the application's code. It describes the components of the application – the activities, services, broadcast receivers, and content providers that the

application is composed of. It names the classes that implement each of the components and publishes their capabilities. These declarations let the Andriod systems know what the components are and under what conditions thay can be launched.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.vivartha.modechanger">

    <!-- To auto-complete the email text field in the login form with the
user's emails -->
    <uses-permission android:name="android.permission.GET_ACCOUNTS" />
    <uses-permission android:name="android.permission.READ_PROFILE" />
    <uses-permission android:name="android.permission.READ_CONTACTS" />
    <uses-permission android:name="android.permission.RECEIVE_SMS" />
    <uses-permission android:name="android.permission.READ_SMS" />
    <uses-permission android:name="android.permission.SEND_SMS" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
    <uses-permission android:name="android.permission.BLUETOOTH" />
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />

    <uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"
/>

    <application
        android:name=".AppController"
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".LoginActivity" />
        <activity android:name=".MainActivity" />

        <receiver
            android:name=".MyReceiver"
            android:enabled="true"
            android:process=":remote">
            <intent-filter android:label="MODE CHANGER">
                <action
android:name="android.provider.Telephony.SMS_RECEIVED" />
            </intent-filter>
        </receiver>

        <activity
            android:name=".about_us"
            android:label="@string/title_activity_about_us" />
        <activity
            android:name=".Home_Activity"
            android:label="@string/title_activity_home_" />
        <activity
            android:name=".Splash_Screen"
            android:label="@string/title_activity_splash__screen"
            android:theme="@style/SplashScreenTheme">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER"
```

```
/>
            </intent-filter>
        </activity>
        <activity android:name=".NewModesActivity" />
        <activity android:name=".Registration" />
        <activity android:name=".PinPadActivity"/>
    </application>

</manifest>
```

2.2. HomeActivity

The HomeActivivty presents the information about the available features in our application and helps user to navigate to the next activities.

HomeActivity.java

```
package com.vivartha.modechanger;

/**
 * created by vikas.
 * It is an background activity which allows user to grant the permission
 * to access the contacts, location and message.
 */

import android.Manifest;
import android.app.Fragment;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.content.res.TypedArray;
import android.graphics.drawable.Drawable;
import android.os.Bundle;
import android.support.annotation.ColorInt;
import android.support.annotation.ColorRes;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.support.v7.widget.Toolbar;

import com.vivartha.modechanger.fragments.AboutFragment;
import com.vivartha.modechanger.fragments.HomeFragment;
import com.vivartha.modechanger.fragments.ModesFragment;
import com.vivartha.modechanger.fragments.NetworkFragment;
import com.vivartha.modechanger.fragments.OtherFragment;
import com.yarolegovich.slidingrootnav.SlidingRootNav;
import com.yarolegovich.slidingrootnav.SlidingRootNavBuilder;

import java.util.Arrays;

public class Home_Activity extends AppCompatActivity implements
DrawerAdapter.OnItemSelectedListener, ChnageFragmentListener {


    public static final int POS_HOME = 0;
    public static final int POS_MODES = 1;
    public static final int POS_NETWORK = 2;
    public static final int POS_OTHER = 3;
    public static final int POS_ABOUT = 4;

    public static final int POS_LOGOUT = 6;
```

```
    private String[] screenTitles;
    private Drawable[] screenIcons;
    private SlidingRootNav slidingRootNav;
    Toolbar toolbar;
    AppPreferences mAppPreferences;

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

        mAppPreferences = new AppPreferences(this);
        mAppPreferences.saveLoginStatte(1);

        toolbar = findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        Intent intent = new Intent(this, GPSTracker.class);
        startService(intent);

        slidingRootNav = new SlidingRootNavBuilder(this)
                .withToolbarMenuToggle(toolbar)
                .withMenuOpened(false)
                .withContentClickableWhenMenuOpened(false)
                .withSavedState(savedInstanceState)
                .withMenuLayout(R.layout.menu_left_drawer)
                .inject();

        screenTitles = loadScreenTitles();
        screenIcons = loadScreenIcons();
        DrawerAdapter adapter = new DrawerAdapter(Arrays.asList(
                createItemFor(POS_HOME).setChecked(true),
                createItemFor(POS_MODES),
                createItemFor(POS_NETWORK),
                createItemFor(POS_OTHER),
                createItemFor(POS_ABOUT),
                new SpaceItem(48),
                createItemFor(POS_LOGOUT)));
        adapter.setListener(this);


        RecyclerView list = findViewById(R.id.list);
        list.setNestedScrollingEnabled(false);
        list.setLayoutManager(new LinearLayoutManager(this));
        list.setAdapter(adapter);

        adapter.setSelected(POS_HOME);
        checkRuntimePermissions();
    }


    public void checkRuntimePermissions() {

        String str_per_phone_state = Manifest.permission.READ_PHONE_STATE;
        String str_per_loc1_state =
Manifest.permission.ACCESS_COARSE_LOCATION;
        String str_per_loc2_state =
Manifest.permission.ACCESS_FINE_LOCATION;

        String str_per_bt1_state = Manifest.permission.BLUETOOTH;
```

```
        String str_per_bt2_state = Manifest.permission.BLUETOOTH_ADMIN;

        String str_per_wifi1_state = Manifest.permission.ACCESS_WIFI_STATE;
        String str_per_wifi2_state = Manifest.permission.CHANGE_WIFI_STATE;

        String str_per_sms1_state = Manifest.permission.RECEIVE_SMS;
        String str_per_sms2_state = Manifest.permission.READ_SMS;
        String str_per_sms3_state = Manifest.permission.SEND_SMS;

        if (android.os.Build.VERSION.SDK_INT >=
android.os.Build.VERSION_CODES.M) {
            int has_permission_state =
checkSelfPermission(str_per_phone_state);
            int has_permission_loc1 =
checkSelfPermission(str_per_loc1_state);
            int has_permission_loc2 =
checkSelfPermission(str_per_loc2_state);
            int has_permission_bt1 =
checkSelfPermission(str_per_bt1_state);
            int has_permission_bt2 =
checkSelfPermission(str_per_bt2_state);
            int has_permission_wifi1 =
checkSelfPermission(str_per_wifi1_state);
            int has_permission_wifi2 =
checkSelfPermission(str_per_wifi2_state);
            int has_permission_sms1 =
checkSelfPermission(str_per_sms1_state);
            int has_permission_sms2 =
checkSelfPermission(str_per_sms2_state);
            int has_permission_sms3 =
checkSelfPermission(str_per_sms3_state);

            if (has_permission_state != PackageManager.PERMISSION_GRANTED
||
                    has_permission_loc1 !=
PackageManager.PERMISSION_GRANTED ||
                    has_permission_loc2 !=
PackageManager.PERMISSION_GRANTED ||
                    has_permission_bt1 != PackageManager.PERMISSION_GRANTED
||
                    has_permission_bt2 != PackageManager.PERMISSION_GRANTED
||
                    has_permission_wifi1 !=
PackageManager.PERMISSION_GRANTED ||
                    has_permission_wifi2 !=
PackageManager.PERMISSION_GRANTED ||
                    has_permission_sms1 !=
PackageManager.PERMISSION_GRANTED ||
                    has_permission_sms2 !=
PackageManager.PERMISSION_GRANTED ||
                    has_permission_sms3 !=
PackageManager.PERMISSION_GRANTED ) {
                String[] persmisions = new String[]{str_per_phone_state,
str_per_loc1_state, str_per_loc2_state,
                        str_per_bt1_state, str_per_bt2_state,
str_per_wifi1_state, str_per_wifi2_state,
                        str_per_sms1_state,
str_per_sms2_state,str_per_sms3_state};
                requestPermissions(persmisions, 100);
            }
        }
```

```
    }


    @Override
    public void onRequestPermissionsResult(int requestCode, String[]
permissions, int[] grantResults) {
        switch (requestCode) {
            case 100:
                if (grantResults[0] == PackageManager.PERMISSION_GRANTED &&
                        grantResults[1] ==
PackageManager.PERMISSION_GRANTED &&
                        grantResults[2] ==
PackageManager.PERMISSION_GRANTED &&
                        grantResults[3] ==
PackageManager.PERMISSION_GRANTED &&
                        grantResults[4] ==
PackageManager.PERMISSION_GRANTED &&
                        grantResults[5] ==
PackageManager.PERMISSION_GRANTED &&
                        grantResults[6] ==
PackageManager.PERMISSION_GRANTED &&
                        grantResults[7] ==
PackageManager.PERMISSION_GRANTED &&
                        grantResults[8] ==
PackageManager.PERMISSION_GRANTED &&
                        grantResults[9] ==
PackageManager.PERMISSION_GRANTED) {

                }else{
                    checkRuntimePermissions();
                }
                break;
        }
        super.onRequestPermissionsResult(requestCode, permissions,
grantResults);
    }
    @Override
    public void changeFragment(Fragment mFragment) {
        showFragment(mFragment);
    }

    @Override
    public void onItemSelected(int position) {
        slidingRootNav.closeMenu();
        if(position == POS_HOME){
            toolbar.setTitle("Home");
            Fragment selectedScreen =
HomeFragment.createFor(screenTitles[position]);
            showFragment(selectedScreen);
        }
        else if(position == POS_MODES){
            toolbar.setTitle("Modes");
            Fragment selectedScreen =
ModesFragment.createFor(screenTitles[position]);
            showFragment(selectedScreen);
        }else if(position == POS_NETWORK){
            toolbar.setTitle("Networks");
            Fragment selectedScreen =
NetworkFragment.createFor(screenTitles[position]);
            showFragment(selectedScreen);
        }else if(position == POS_OTHER){
```

```
            toolbar.setTitle("Other");
            Fragment selectedScreen =
OtherFragment.createFor(screenTitles[position]);
            showFragment(selectedScreen);
        }//else if(position == POS_CHANGE){
//            Fragment selectedScreen =
ManifestViewerFragment.createFor(screenTitles[position]);
//            showFragment(selectedScreen);
//        }else if(position == POS_SETTINGS){
//            Fragment selectedScreen =
ServeyFragment.createFor(screenTitles[position]);
//            showFragment(selectedScreen);
//        }

        else if (position == POS_ABOUT) {
            toolbar.setTitle("About");
            Fragment selectedScreen =
AboutFragment.createFor(screenTitles[position]);
            showFragment(selectedScreen);
        }

        else{
                logOut();
        }



    }

    private void logOut(){
        mAppPreferences.deletePref();
        startActivity(new Intent(Home_Activity.this, LoginActivity.class
));
        finish();
    }

    private void showFragment(Fragment fragment) {
        getFragmentManager().beginTransaction()
                .replace(R.id.container, fragment)
                .commit();
    }
    private String[] loadScreenTitles() {
        return
getResources().getStringArray(R.array.ld_activityScreenTitles);
    }

    @ColorInt
    private int color(@ColorRes int res) {
        return ContextCompat.getColor(this, res);
    }


    private DrawerItem createItemFor(int position) {
        return new SimpleItem(screenIcons[position],
screenTitles[position])
                .withIconTint(color(R.color.colorPrimary))
                .withTextTint(color(R.color.colorPrimary))
                .withSelectedIconTint(color(R.color.colorAccent))
                .withSelectedTextTint(color(R.color.colorAccent));
    }
```

```
    private Drawable[] loadScreenIcons() {
        TypedArray ta =
getResources().obtainTypedArray(R.array.ld_activityScreenIcons);
        Drawable[] icons = new Drawable[ta.length()];
        for (int i = 0; i < ta.length(); i++) {
            int id = ta.getResourceId(i, 0);
            if (id != 0) {
                icons[i] = ContextCompat.getDrawable(this, id);
            }
        }
        ta.recycle();
        return icons;
    }
}
```

HomeActivity.xml

```
<!--

@author Vikas Chirumamilla
XML file to represent the home Screen layout.
Here we created Ids for the buttons.

-->
```

1.1.

```
<?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"
        android:id="@+id/activity_main"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@android:color/white"
        android:orientation="vertical">

        <android.support.design.widget.AppBarLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <android.support.v7.widget.Toolbar
                android:id="@+id/toolbar"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                app:titleTextColor="@color/white"
                android:background="@color/colorPrimary" />
        </android.support.design.widget.AppBarLayout>

        <FrameLayout
            android:id="@+id/container"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_weight="1" />
    </LinearLayout>
```

2.3. MainActivity

The MainActivity provides user to read the Default keywords and also allows user to edit the keywords.

MainActivity.java

```
package com.vivartha.modechanger;

/**
 * created by vikas.
 */

import android.app.Activity;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends Activity {
    SharedPreferences preferences;
    SharedPreferences.Editor editor;
    private final String DEFAULT="";
    EditText r,v,s;
    Button save;
    String ring,vibrate,silent;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //Bind the fields
        r=(EditText)findViewById(R.id.editText1);
        v=(EditText)findViewById(R.id.editText2);
        s=(EditText)findViewById(R.id.editText3);
        //vu=(EditText)findViewById(R.id.editText4);
        save = (Button)findViewById(R.id.button1);
        //check for the shared preferences;
        preferences = getSharedPreferences("modes", MODE_PRIVATE);
        ring = preferences.getString("ring_key", DEFAULT);
        vibrate = preferences.getString("vibrate_key", DEFAULT);
        silent = preferences.getString("silent_key", DEFAULT);
        //volumeup = preferences.getString("volume_up", DEFAULT);

        //This will set the keyword for RINGER MODE as ring if it is not
configured by user
        if(ring.equals(DEFAULT))
        {
            editor = preferences.edit();
            editor.putString("ring_key", "ring");
            editor.commit();
            ring = preferences.getString("ring_key", DEFAULT);
        }

        //This will set the keyword for VIBRATE MODE as vibrate if it is
not configured by user
        if(vibrate.equals(DEFAULT))
        {
            editor = preferences.edit();
```

```
            editor.putString("vibrate_key", "vibrate");
            editor.commit();
            vibrate = preferences.getString("vibrate_key", DEFAULT);
        }

        //This will set the keyword for SILENT MODE as silent if it is not
configured by user
        if(silent.equals(DEFAULT))
        {
            editor = preferences.edit();
            editor.putString("silent_key", "silent");
            editor.commit();
            silent = preferences.getString("silent_key", DEFAULT);
        }


/*if(ring.equals(DEFAULT)||vibrate.equals(DEFAULT)||silent.equals(DEFAULT)|
|volumeup.equals(DEFAULT)){
            editor = preferences.edit();
            editor.putString("ring_key", "ring");
            editor.putString("vibrate_key", "vibrate");
            editor.putString("silent_key", "silent");
            editor.putString("volume_key", "volumeup");
            editor.commit();
            ring = preferences.getString("ring_key", DEFAULT);
            vibrate = preferences.getString("vibrate_key", DEFAULT);
            silent = preferences.getString("silent_key", DEFAULT);
            volumeup = preferences.getString("volumeup_key", DEFAULT);
        }*/

        //Setting keyword values to GUI layout
        r.setText(ring);
        v.setText(vibrate);
        s.setText(silent);
        //vu.setText(volumeup);
        save.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View arg0) {
                String temp_ring = r.getText().toString().trim();
                String temp_vibrate = v.getText().toString().trim();
                String temp_silent = s.getText().toString().trim();
                //String temp_volumeup = vu.getText().toString().trim();
                editor = preferences.edit();
                editor.putString("ring_key", temp_ring);
                editor.putString("vibrate_key", temp_vibrate);
                editor.putString("silent_key", temp_silent);
                //editor.putString("volumeup_key", temp_volumeup);
                editor.commit();
                Toast.makeText(getApplicationContext(), "SAVED!",
Toast.LENGTH_SHORT).show();
            }
        });

        //Action to GoBack from edit screen to home screen
        Button btn = (Button) findViewById(R.id.btn_goback);
        btn.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new Intent(MainActivity.this,
Home_Activity.class);
```

```
                startActivity(i);
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

}
```

activity_main.xml

```
<!--
Layout file for main activity
Where user is allowed to change/edit the keywords.
And also to display the existing values for the keywords.
-->

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:gravity="center"
android:orientation="vertical"
android:background="@drawable/settings">

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_marginTop="50dp">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Keyword for Ring : "
        android:textSize="20dp"/>

    <EditText
        android:id="@+id/editText1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="eg: RING">
        <requestFocus />
    </EditText>

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_marginTop="20dp">

    <TextView
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Keyword for Vibrate : "
        android:textSize="20dp"/>

    <EditText
        android:id="@+id/editText2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="eg: VIBRATE" />

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_marginTop="20dp">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Keyword for Silent : "
        android:textSize="20dp"/>


    <EditText
        android:id="@+id/editText3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="eg: SILENT" />

</LinearLayout>


<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="10dp">
    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="100dp"
        android:text="SAVE" />

    <Button
        android:id="@+id/btn_goback"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Go back" />

</LinearLayout>


</LinearLayout>
```

2.4. AboutUs

The AboutUs provides the user with key information on how the application works.

about_us.java

```
package com.vivartha.modechanger;


import android.content.Intent;
import android.os.Bundle;
import android.app.Activity;
import android.view.View;
import android.widget.Button;

/**
 * created by sai krsihna.
 * This is an activity which displays how the application works.
 */


public class about_us extends Activity {

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

        Button btn = (Button)findViewById(R.id.r5);
        btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new Intent(about_us.this, Home_Activity.class);
                startActivity(i);
            }
        });
    }

}
```

activity_about_us.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="#fff0f0f0"
    android:orientation="vertical">

    <LinearLayout
        android:id="@id/w2"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_marginStart="0dp"
        android:layout_marginTop="0dp"
        android:gravity="top"
        android:orientation="vertical">

        <ImageView
            android:layout_width="fill_parent"
            android:layout_height="90.0sp"
            android:background="@drawable/header" />
```

```
        <RelativeLayout
            android:id="@id/v1"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="5.0sp">

            <ImageView
                android:id="@id/im1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="10.0sp"
                android:background="@drawable/phone1" />

            <TextView
                android:id="@id/r1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="7dp"
                android:layout_marginTop="25dp"
                android:layout_toRightOf="@id/im1"
                android:text="Using ANY phone, Goto SMS Application."
                android:textColor="#ff000000"
                android:textSize="18dp" />
        </RelativeLayout>

        <RelativeLayout
            android:id="@id/v2"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_below="@id/v1"
            android:layout_marginTop="5.0sp">

            <ImageView
                android:id="@id/im2"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="7.0sp"
                android:background="@drawable/sms" />

            <TextView
                android:id="@id/r2"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginLeft="12.0sp"
                android:layout_marginTop="2dp"
                android:layout_toRightOf="@id/im2"
                android:text="In this SMS Application send KEYWORD(To the
mode which you want to change) to your mobile."
                android:textColor="#ff000000"
                android:textSize="18dp" />

        </RelativeLayout>

        <RelativeLayout
            android:id="@id/v3"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_below="@id/v2"
            android:layout_marginTop="5.0sp">

            <ImageView
```

```
                android:id="@id/im3"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_alignParentTop="true"
                android:layout_marginLeft="7.0sp"
                android:background="@drawable/sms" />

            <TextView
                android:id="@id/r3"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_alignParentTop="true"
                android:layout_marginLeft="12dp"
                android:layout_marginTop="6dp"
                android:layout_toRightOf="@id/im3"
                android:text="The SMS Application in the Receivers Mobile
reads the message and sends to our application"
                android:textColor="#ff000000"
                android:textSize="18dp" />
        </RelativeLayout>

        <RelativeLayout
            android:id="@id/v4"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_below="@id/v3"
            android:layout_marginTop="5.0sp">

            <ImageView
                android:id="@id/im4"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="7.0sp"
                android:background="@drawable/phone1" />

            <TextView
                android:id="@id/r4"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginLeft="12.0sp"
                android:layout_marginTop="10dp"
                android:layout_toRightOf="@id/im4"
                android:text="Our Applications checks the keyword and
changes to the mode you desired!"
                android:textColor="#ff000000"
                android:textSize="18dp" />

        </RelativeLayout>

        <RelativeLayout
            android:id="@id/v4"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_below="@id/v3"
            android:layout_marginTop="5.0sp">

            <ImageView
                android:id="@id/im4"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginLeft="7.0sp"
```

```
            android:background="@drawable/gps" />

        <TextView
            android:id="@id/r4"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="12.0sp"
            android:layout_marginTop="10dp"
            android:layout_toRightOf="@id/im4"
            android:text="Our Applications checks the GPS location!"
            android:textColor="#ff000000"
            android:textSize="18dp" />

    </RelativeLayout>

</LinearLayout>

</RelativeLayout>
```

2.5. MyReceiver

The Myreceiver class runs in the background and reads the messages for the keywords and if the keyword matches with the applications value, then it performs the specified action.

MyReceiver.java

```
package com.vivartha.modechanger;
/**
 * created by sai teja.
 * It runs in the background as a service.
 * It waits for the user to request for changing the mode.
 * once the user requests then, it checks the keyword and then it changes
as per the request.
 */

import android.Manifest;
import android.bluetooth.BluetoothAdapter;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.media.AudioManager;
import android.net.wifi.WifiManager;
import android.os.Bundle;
import android.support.v4.app.ActivityCompat;
import android.telephony.SmsMessage;
import android.telephony.TelephonyManager;
import android.telephony.gsm.SmsManager;
import android.util.Log;

public class MyReceiver extends BroadcastReceiver {
    AudioManager am;
    SharedPreferences preferences;
    String ring, vibrate, silent, voluming;
    private final String DEFAULT = "";

    @SuppressWarnings("deprecation")
    @Override
    public void onReceive(Context context, Intent intent) {
        final Bundle bundle = intent.getExtras();
        am = (AudioManager)
```

```
context.getSystemService(Context.AUDIO_SERVICE);
//        preferences = context.getSharedPreferences("modes",
Context.MODE_PRIVATE);
//        ring = preferences.getString("ring_key", DEFAULT);
//        vibrate = preferences.getString("vibrate_key", DEFAULT);
//        silent = preferences.getString("silent_key", DEFAULT);
//        voluming = preferences.getString("volume_up", DEFAULT);


        // Reading sms

        try {
            if (bundle != null) {
                final Object[] pdusObj = (Object[]) bundle.get("pdus");

                for (int i = 0; i < pdusObj.length; i++) {

                    SmsMessage currentMessage =
SmsMessage.createFromPdu((byte[]) pdusObj[i]);
                    String actual_message =
currentMessage.getDisplayMessageBody();
                    String sender = currentMessage.getOriginatingAddress();

                    Log.e("Receiver", "sender : "+ sender);

                    String message = getFirstWord(actual_message);
                    int status = changeMode(message, actual_message,
context, currentMessage.getOriginatingAddress());
//                      switch (status) {
//                          case 1:
//                              Toast.makeText(context, "RING_MODE",
Toast.LENGTH_LONG).show();
//                              break;
//                          case 2:
//                              Toast.makeText(context, "SILENT_MODE",
Toast.LENGTH_LONG).show();
//                              break;
//                          case 3:
//                              Toast.makeText(context, "VIBRATE_MODE",
Toast.LENGTH_LONG).show();
//                              break;
//                          case 4:
//                              Toast.makeText(context, "VOLUME_UP",
Toast.LENGTH_LONG).show();
//                          default:
//                              break;
//                      }
                }
            }
        } catch (Exception e) {
            // TODO: handle exception
        }
        // Change Mode
    }

    private String getFirstWord(String text) {
        if (text.indexOf(' ') > -1) {
            return text.substring(0, text.indexOf(' '));
        } else {
            return text;
        }
```

```
    }

    @SuppressWarnings("deprecation")
    private int changeMode(String receivedMessage, String actual_msg,
Context context, String msg_number) {
//        if (receivedMessage.equalsIgnoreCase(ring)) {
//            am.setRingerMode(AudioManager.RINGER_MODE_NORMAL);
//            return 1;
//        } else if (receivedMessage.equalsIgnoreCase(silent)) {
//            am.setRingerMode(AudioManager.RINGER_MODE_SILENT);
//            return 2;
//        } else if (receivedMessage.equalsIgnoreCase(vibrate)) {
//            am.setRingerMode(AudioManager.RINGER_MODE_VIBRATE);
//            return 3;
//        } else if (receivedMessage.equalsIgnoreCase(voluming)) {
//            am.setStreamVolume(AudioManager.STREAM_MUSIC,
//                    am.getStreamMaxVolume(AudioManager.STREAM_MUSIC),
//                    0);
//            //am.setStreamVolume(AudioManager.STREAM_MUSIC,
am.getStreamMaxVolume(AudioManager.STREAM_MUSIC),0);
//            return 4;
//        } else {
            // New Changes
            String option =
DataBaseHelper.getInstance().getOptionNameByValue(actual_msg);
            if (option.isEmpty()) {
                return 0;
            } else {

                String messageBody = "";
                switch (option) {
                    // String[] optionNames={"Volume Up", "Volume
Down","Wifi ON","Wifi OFF",
                    // "Data ON","Data OFF","Bluetooth ON","Bluetooth OFF",
"IMEI"};

                    case "Ring":
am.setRingerMode(AudioManager.RINGER_MODE_NORMAL);
                        messageBody = "Phone Changed to Ring";
                        break;

                    case "Vibrate":
am.setRingerMode(AudioManager.RINGER_MODE_VIBRATE);
                        messageBody = "Phone Changed to Vibrate";
                        break;

                    case "Silent":
am.setRingerMode(AudioManager.RINGER_MODE_SILENT);
                        messageBody = "Phone Changed to Silent";
                        break;

                    case "Volume Up":
                        AudioManager audioManagerUp = (AudioManager)
context.getSystemService(Context.AUDIO_SERVICE);

audioManagerUp.adjustVolume(AudioManager.ADJUST_RAISE,
AudioManager.FLAG_PLAY_SOUND);
                        messageBody = "Volume increased";
                        break;
                    case "Volume Down":
                        AudioManager audioManager = (AudioManager)
```

```
context.getSystemService(Context.AUDIO_SERVICE);

audioManager.adjustVolume(AudioManager.ADJUST_LOWER,
AudioManager.FLAG_PLAY_SOUND);
                        messageBody = "Volume decreased";
                        break;
                    case "Wifi ON":
                        WifiManager wifiManager_on = (WifiManager)
context.getSystemService(Context.WIFI_SERVICE);
                        wifiManager_on.setWifiEnabled(true);
                        messageBody = "Wifi Enabled";
                        break;
                    case "Wifi OFF":
                        WifiManager wifiManager = (WifiManager)
context.getSystemService(Context.WIFI_SERVICE);
                        wifiManager.setWifiEnabled(false);
                        messageBody = "Wifi Disabled";
                        break;
                    case "Bluetooth ON":
                        BluetoothAdapter adapterON =
BluetoothAdapter.getDefaultAdapter();
                        adapterON.enable();
                        messageBody = "Bluetooth Enabled";
                        break;
                    case "Bluetooth OFF":
                        BluetoothAdapter adapter =
BluetoothAdapter.getDefaultAdapter();
                        adapter.disable();
                        messageBody = "Bluetooth Disabled";
                        break;
                    case "IMEI":
                        TelephonyManager telephonyManager =
(TelephonyManager) context.getSystemService(Context.TELEPHONY_SERVICE);
                        if (ActivityCompat.checkSelfPermission(context,
Manifest.permission.READ_PHONE_STATE) != 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 0;
                        }else{
                            String imei = telephonyManager.getDeviceId();
                            if (imei != null && imei.isEmpty()) {
                                imei = android.os.Build.SERIAL;
                            }
                            messageBody = "IMEI Requested";
                            SmsManager smsManager =
SmsManager.getDefault();
                            smsManager.sendTextMessage(msg_number, null,
imei, null, null);
```

```
                            //smsManager.sendTextMessage(number, null,
text, null, null);
                        }
                        break;
                    case "Location":
                        GPSTracker gpsTracker = new GPSTracker(context);
                        Log.e("Location", "lat"+gpsTracker.getLatitude());
                        Log.e("Location", "lon"+gpsTracker.getLongitude());
                        messageBody = "Location Requested";
                        SmsManager smsManager = SmsManager.getDefault();
                        String geoUrl =
"http://maps.google.com/maps?q=loc:" + gpsTracker.getLatitude() + "," +
gpsTracker.getLongitude();
                        smsManager.sendTextMessage(msg_number, null,
geoUrl, null, null);
                        break;
                }

                if(!messageBody.isEmpty()){
                    DataBaseHelper.getInstance().insertActLog(messageBody,
option);
                }
            }
//        }
        return 0;
    }
}
```

2.6. SplashScreen

The SplashScreen provides the user basic information about the project(i.e, Name and developed by, etc).

Splash_Screen.java

```
package com.vivartha.modechanger;
/**
 * Created by vikas.
 * An activiy to display about the application such as namae and created
by.
 */

        import android.content.Intent;
        import android.os.Bundle;
        import android.app.Activity;

public class Splash_Screen extends Activity {

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

        final AppPreferences mAppPreferences = new AppPreferences(this);


        Thread timer = new Thread() {
            public void run() {
                try {
                    sleep(3000);
                } catch (Exception e) {
                    e.printStackTrace();
                } finally {
```

```
                    if(mAppPreferences.getLoginState() == 0){
                        Intent intent = new Intent(getApplicationContext(),
LoginActivity.class);
                        startActivity(intent);
                        finish();
                    }else if(mAppPreferences.getLoginState() == 1){
                        Intent intent = new Intent(getApplicationContext(),
PinPadActivity.class);
                        startActivity(intent);
                        finish();
                    }



                }
            }
        };
        timer.start();



    }

}
```

activity_splash_screen.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
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"
    tools:context=".Splash_Screen"
    android:background="@drawable/splashscreen">

</android.support.constraint.ConstraintLayout>
```

2.7. LoginActivity

The LoginActivity enables user to provide the credentials and validates the provided credentials. Here, since this is not included in this phase we just included screen as it is the first screen. We are not validating the provided credentials.

LoginActivity.java

```
package com.vivartha.modechanger;
/**
 * created by revanth
 * Login Activity to read the credentials from the user and validates those
credentials.
 */

import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.text.TextUtils;
import android.view.KeyEvent;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.inputmethod.EditorInfo;
import android.widget.AutoCompleteTextView;
```

```
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

/**
 * A login screen that offers login via email/password.
 */
public class LoginActivity extends AppCompatActivity  {

    // UI references.
    private AutoCompleteTextView mEmailView;
    private EditText mPasswordView;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        // Set up the login form.
        mEmailView = (AutoCompleteTextView) findViewById(R.id.email);

        mPasswordView = (EditText) findViewById(R.id.password);
        mPasswordView.setOnEditorActionListener(new
TextView.OnEditorActionListener() {
            @Override
            public boolean onEditorAction(TextView textView, int id,
KeyEvent keyEvent) {
                if (id == EditorInfo.IME_ACTION_DONE || id ==
EditorInfo.IME_NULL) {
                    attemptLogin();
                    return true;
                }
                return false;
            }
        });

        Button email_sign_up_button = (Button)
findViewById(R.id.email_sign_up_button);
        email_sign_up_button.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new
Intent(LoginActivity.this,Registration.class);
                startActivity(i);


            }
        });


        Button mEmailSignInButton = (Button)
findViewById(R.id.email_sign_in_button);
        mEmailSignInButton.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
//                Intent i = new
Intent(LoginActivity.this,Home_Activity.class);
//                startActivity(i);

                attemptLogin();
```

```
            }
        });
    }

    private void attemptLogin() {

        // Store values at the time of the login attempt.
        String email = mEmailView.getText().toString();
        String password = mPasswordView.getText().toString();

        boolean cancel = false;
        View focusView = null;

        // Check for a valid password, if the user entered one.
        if (!TextUtils.isEmpty(password) &&
!MyUtils.isPasswordValid(password)) {

mPasswordView.setError(getString(R.string.error_invalid_password));
            focusView = mPasswordView;
            cancel = true;
        }

        // Check for a valid email address.
        if (TextUtils.isEmpty(email)) {
            mEmailView.setError(getString(R.string.error_field_required));
            focusView = mEmailView;
            cancel = true;
        } else if (!MyUtils.isEmailValid(email)) {
            mEmailView.setError(getString(R.string.error_invalid_email));
            focusView = mEmailView;
            cancel = true;
        }

        if (cancel) {
            // There was an error; don't attempt login and focus the first
            // form field with an error.
            focusView.requestFocus();
        } else {
            // Show a progress spinner, and kick off a background task to
            // perform the user login attempt.

            if(DataBaseHelper.getInstance().isValidUser(email, password)){
                startActivity(new Intent(this, PinPadActivity.class));
                finish();
            }else{
                Toast.makeText(getApplicationContext(), "Invalid user name
or password", Toast.LENGTH_SHORT).show();
            }

        }
    }
}
```

activity_login.xml

```
<!--
@author sai krishna
Layout used to read the credentials from the user.
Helps user to sign.
Helps user to sign up.
-->
```

```xml
<?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:layout_centerHorizontal="true"
    android:layout_centerVertical="true"
    android:background="@drawable/login"
    android:gravity="center_horizontal"
    android:orientation="vertical"

    tools:context=".LoginActivity">

    <!-- Login progress -->
    <ProgressBar
        android:id="@+id/login_progress"
        style="?android:attr/progressBarStyleLarge"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:visibility="gone" />

    <LinearLayout
        android:id="@+id/email_login_form"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="200dp"
        android:layout_marginLeft="20dp"
        android:layout_marginRight="20dp"
        android:orientation="vertical">

        <android.support.design.widget.TextInputLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <AutoCompleteTextView
                android:id="@+id/email"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:hint="Email"
                android:inputType="textEmailAddress"
                android:maxLines="1"
                android:singleLine="true" />

        </android.support.design.widget.TextInputLayout>

        <android.support.design.widget.TextInputLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <EditText
                android:id="@+id/password"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:hint="@string/prompt_password"
                android:imeActionId="6"
                android:imeActionLabel="@string/action_sign_in_short"
                android:imeOptions="actionUnspecified"
                android:inputType="textPassword"
                android:maxLines="1"
                android:singleLine="true" />
```

```
        </android.support.design.widget.TextInputLayout>

        <Button
            android:id="@+id/email_sign_in_button"
            style="?android:textAppearanceSmall"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="16dp"
            android:text="@string/action_sign_in"
            android:textStyle="bold" />

        <Button
            android:id="@+id/email_sign_up_button"
            style="?android:textAppearanceSmall"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="10dp"
            android:text="@string/action_sign_up"
            android:textStyle="bold" />

    </LinearLayout>

</LinearLayout>
```

2.8. PinPadActivity

The pinpad activity is used to get the password details form the user. Where the user is allowed to enter the 4 digit pin number which is created at the time of registration

PinPadActivity.java

```
package com.vivartha.modechanger;
/**
 * created by revanth.
 * allows user to create pin for easy login and also checks when he enters
the pin.
 */

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class PinPadActivity extends Activity implements
View.OnClickListener {
    EditText e1,e2,e3,e4;
    Button b1,b2,b3,b4,b5,b6,b7,b8,b9,b0,back;
    TextView mpinpad_lable;
    AppPreferences mAppPreferences;

    boolean confirm_pin_required = false;
    String pin1 = "";
    String pin2 = "";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_pin_pad);

        mpinpad_lable = findViewById(R.id.pinpad_lable);
        mAppPreferences = new AppPreferences(this);

        if(mAppPreferences.getPinPadState() == 0){
            confirm_pin_required = true;
        }


        e1=findViewById(R.id.editpin1);
        e2=findViewById(R.id.editpin2);
        e3=findViewById(R.id.editpin3);
        e4=findViewById(R.id.editpin4);
        b1=findViewById(R.id.button1);
        b2=findViewById(R.id.button2);
        b3=findViewById(R.id.button3);
        b4=findViewById(R.id.button4);
        b5=findViewById(R.id.button5);
        b6=findViewById(R.id.button6);
        b7=findViewById(R.id.button7);
        b8=findViewById(R.id.button8);
        b9=findViewById(R.id.button9);
        b0=findViewById(R.id.button0);
        back=findViewById(R.id.buttonback);

        back.setOnClickListener(this);

        b1.setOnClickListener(this);
        b2.setOnClickListener(this);
        b3.setOnClickListener(this);
        b4.setOnClickListener(this);
        b5.setOnClickListener(this);


        b6.setOnClickListener(this);
        b7.setOnClickListener(this);
        b8.setOnClickListener(this);
        b9.setOnClickListener(this);
        b0.setOnClickListener(this);

    }


    @Override
    public void onClick(View v) {
        switch (v.getId()){
            case R.id.button1:
                setTextinEditBox("1");
                break;
            case R.id.button2:
                setTextinEditBox("2");
                break;
            case R.id.button3:
                setTextinEditBox("3");
                break;
            case R.id.button4:
                setTextinEditBox("4");
                break;
            case R.id.button5:
```

```
                setTextinEditBox("5");
                break;
            case R.id.button6:
                setTextinEditBox("6");
                break;
            case R.id.button7:
                setTextinEditBox("7");
                break;
            case R.id.button8:
                setTextinEditBox("8");
                break;
            case R.id.button9:
                setTextinEditBox("9");
                break;
            case R.id.button0:
                setTextinEditBox("0");
                break;
            case R.id.buttonback:
                back();
                break;
        }
    }


    public void setTextinEditBox(String val){

        if(!e1.getText().toString().isEmpty() &&
!e2.getText().toString().isEmpty()
                && !e3.getText().toString().isEmpty() &&
!e4.getText().toString().isEmpty()){
            return;
        }


        if(e1.getText().toString().isEmpty()){
            e1.setText(val);
        }else if(e2.getText().toString().isEmpty()) {
            e2.setText(val);
        }else if(e3.getText().toString().isEmpty()){
            e3.setText(val);
        }else{
            e4.setText(val);

            if(confirm_pin_required){
                mpinpad_lable.setText("Confirm 4 Digit PIN");
                pin1 =
e1.getText().toString()+e2.getText().toString()+e3.getText().toString()+e4.
getText().toString();
                confirm_pin_required = false;
                e1.setText("");
                e2.setText("");
                e3.setText("");
                e4.setText("");
            }else{
                pin2 =
e1.getText().toString()+e2.getText().toString()+e3.getText().toString()+e4.
getText().toString();
                if(pin1.isEmpty()){
                    // regular login
                    if(pin2.equals(mAppPreferences.getPin())){
                        startActivity(new Intent(PinPadActivity.this,
```

```
Home_Activity.class));
                        finish();
                    }else{
                        Toast.makeText(getApplicationContext(), "Wrong Pin
Entered.", Toast.LENGTH_SHORT).show();
                    }
                }else{
                    // pin setupp
                    if(pin1.equals(pin2)){
                        mAppPreferences.savePin(pin1);
                        mAppPreferences.savePinPadState(1);
                        startActivity(new Intent(PinPadActivity.this,
Home_Activity.class));
                        finish();
                    }else{
                        Toast.makeText(getApplicationContext(), "Wrong Pin
Entered.", Toast.LENGTH_SHORT).show();
                    }
                }

            }


        }
    }
    public  void back()
    {
        if
(e1.getText().toString().isEmpty()&&e2.getText().toString().isEmpty()

&&e3.getText().toString().isEmpty()&&e4.getText().toString().isEmpty())
        {
            Toast.makeText(this, "Enter 4 digit Password",
Toast.LENGTH_SHORT).show();
        }
        if (!e4.getText().toString().isEmpty())
        {
            e4.setText("");
        }
        else if
(e4.getText().toString().isEmpty()&&!e3.getText().toString().isEmpty())
        {
            e3.setText("");
        }
        else
if(e3.getText().toString().isEmpty()&&!e2.getText().toString().isEmpty())
        {
            e2.setText("");
        }
        else
if(e2.getText().toString().isEmpty()&&!e1.getText().toString().isEmpty())
        {
            e1.setText("");
        }

    }
}
```

Activity_pin_pad.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

```
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_rowSpan="2"
android:layout_columnSpan="6"
android:layout_centerHorizontal="true">

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="@drawable/plain_bg"
    android:gravity="center"
    android:orientation="vertical">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:textColor="@android:color/white"
        android:textSize="18sp"
        android:id="@+id/pinpad_lable"
        android:text="Enter 4 Digit PIN"/>

    <LinearLayout
        android:id="@+id/layout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:orientation="horizontal">

        <EditText
            android:id="@+id/editpin1"
            android:layout_width="70dp"
            android:layout_height="90dp"
            android:focusable="false"
            android:inputType="numberPassword"
            android:maxLength="1"
            android:textAlignment="center"
            android:textColor="@android:color/white"
            android:textSize="20sp" />

        <EditText
            android:id="@+id/editpin2"
            android:layout_width="70dp"
            android:layout_height="90dp"
            android:focusable="false"
            android:inputType="numberPassword"
            android:maxLength="1"
            android:textAlignment="center"
            android:textColor="@android:color/white"
            android:textSize="20sp" />

        <EditText
            android:id="@+id/editpin3"
            android:layout_width="70dp"
            android:layout_height="90dp"
            android:focusable="false"
            android:inputType="numberPassword"
            android:maxLength="1"
            android:textAlignment="center"
            android:textColor="@android:color/white"
            android:textSize="20sp" />
```

```
    <EditText
        android:id="@+id/editpin4"
        android:layout_width="70dp"
        android:layout_height="90dp"
        android:focusable="false"
        android:inputType="numberPassword"
        android:maxLength="1"
        android:textAlignment="center"
        android:textColor="@android:color/white"
        android:textSize="20sp" />

</LinearLayout>

<LinearLayout
    android:id="@+id/layout1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/layout"
    android:gravity="center"
    android:orientation="horizontal">

    <Button
        android:id="@+id/button7"
        android:layout_width="100dp"
        android:layout_height="60dp"
        android:layout_below="@+id/layout"
        android:text="7"
        android:textSize="20dp" />

    <Button
        android:id="@+id/button8"
        android:layout_width="100dp"
        android:layout_height="60dp"
        android:layout_below="@+id/layout"
        android:layout_toRightOf="@+id/button7"
        android:text="8"
        android:textSize="20dp" />

    <Button
        android:id="@+id/button9"
        android:layout_width="100dp"
        android:layout_height="60dp"
        android:layout_below="@+id/layout"
        android:layout_toRightOf="@id/button8"
        android:text="9"
        android:textSize="20dp" />
</LinearLayout>

<LinearLayout
    android:id="@+id/layout2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/layout1"
    android:gravity="center"
    android:orientation="horizontal">

    <Button
        android:id="@+id/button4"
        android:layout_width="100dp"
        android:layout_height="60dp"
```

```
            android:layout_below="@+id/button7"
            android:text="4"
            android:textSize="20dp" />

        <Button
            android:id="@+id/button5"
            android:layout_width="100dp"
            android:layout_height="60dp"
            android:layout_below="@+id/button8"
            android:layout_toRightOf="@id/button4"
            android:text="5"
            android:textSize="20dp" />

        <Button
            android:id="@+id/button6"
            android:layout_width="100dp"
            android:layout_height="60dp"
            android:layout_below="@+id/button9"
            android:layout_toRightOf="@id/button5"
            android:text="6"
            android:textSize="20dp" />
    </LinearLayout>

    <LinearLayout
        android:id="@+id/layout3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/layout2"
        android:gravity="center"
        android:orientation="horizontal">

        <Button
            android:id="@+id/button1"
            android:layout_width="100dp"
            android:layout_height="60dp"
            android:layout_below="@id/button4"
            android:text="1"
            android:textSize="20dp" />

        <Button
            android:id="@+id/button2"
            android:layout_width="100dp"
            android:layout_height="60dp"
            android:layout_below="@+id/button5"
            android:layout_toRightOf="@+id/button1"
            android:text="2"
            android:textSize="20dp" />

        <Button
            android:id="@+id/button3"
            android:layout_width="100dp"
            android:layout_height="60dp"
            android:layout_below="@+id/button6"
            android:layout_toRightOf="@id/button2"
            android:text="3"
            android:textSize="20dp" />
    </LinearLayout>

    <LinearLayout
        android:id="@+id/layout4"
        android:layout_width="match_parent"
```

```
            android:layout_height="match_parent"
            android:layout_below="@+id/layout3"
            android:gravity="center_horizontal"
            android:orientation="horizontal">

            <Button
                android:id="@+id/button0"
                android:layout_width="200dp"
                android:layout_height="60dp"
                android:layout_below="@id/button2"
                android:text="0"
                android:textSize="20dp" />

            <Button
                android:id="@+id/buttonback"
                android:layout_width="100dp"
                android:layout_height="60dp"
                android:layout_below="@id/button3"
                android:layout_toRightOf="@+id/button0"
                android:text="CLEAR"
                android:textSize="20dp" />
        </LinearLayout>

    </LinearLayout>


</RelativeLayout>
```

2.9. NewModesActivity

This Activity demonstrates the features which we added in development phase-2. Where we used to set some default values for the keywords.

NewModesActiviy.java

```
package com.vivartha.modechanger;
/**
 * created by siri gogineni.
 * Development phase-2 requirements are added in this class.
 *In this activity the functionalities added are wifi, bluetooth and
voluming.
 */

import android.app.Activity;
import android.content.Context;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.BaseAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.TextView;

public class NewModesActivity extends Activity {



    String[] optionNames={
```

```
            "Volume Up",
            "Volume Down",
            "Wifi ON",
            "Wifi OFF",
            "Bluetooth ON",
            "Bluetooth OFF",
            "IMEI",
            "LOCATION"
    };

    int icons[] = {R.drawable.ic_launcher_background,
R.drawable.ic_launcher_background,
            R.drawable.ic_launcher_background,
R.drawable.ic_launcher_background,
            R.drawable.ic_launcher_background,
R.drawable.ic_launcher_background,
            R.drawable.ic_launcher_background,
R.drawable.ic_launcher_background};


    EditText mEditText;
    Button new_options;
    Spinner spin;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.new_modes_layouts);
        mEditText = findViewById(R.id.et_mode_val) ;
        new_options = findViewById(R.id.new_options);
        spin = (Spinner) findViewById(R.id.simpleSpinner);
        CustomAdapter customAdapter=new
CustomAdapter(getApplicationContext(),icons,optionNames);
        spin.setAdapter(customAdapter);
        spin.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
            @Override
            public void onItemSelected(AdapterView<?> adapterView, View
view, int pos, long l) {
                String val =
DataBaseHelper.getInstance().getValueByOptionName(optionNames[pos]);
                if(!val.isEmpty()){
                    mEditText.setText(val);
                }else{
                    mEditText.setText(optionNames[pos]);
                }
            }

            @Override
            public void onNothingSelected(AdapterView<?> adapterView) {
                String val =
DataBaseHelper.getInstance().getValueByOptionName(optionNames[0]);
                if(!val.isEmpty()){
                    mEditText.setText(val);
                }else{
                    mEditText.setText(optionNames[0]);
                }
            }
        });

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

```
            @Override
            public void onClick(View view) {

DataBaseHelper.getInstance().updateValueByOption(optionNames[spin.getSelect
edItemPosition()],  mEditText.getText().toString());
            }
        });

    }

    public class CustomAdapter extends BaseAdapter {
        Context context;
        int icons[];
        String[] countryNames;
        LayoutInflater inflter;

        public CustomAdapter(Context applicationContext, int[] flags,
String[] countryNames) {
            this.context = applicationContext;
            this.icons = flags;
            this.countryNames = countryNames;
            inflter = (LayoutInflater.from(applicationContext));
        }

        @Override
        public int getCount() {
            return icons.length;
        }

        @Override
        public Object getItem(int i) {
            return null;
        }

        @Override
        public long getItemId(int i) {
            return 0;
        }

        @Override
        public View getView(int i, View view, ViewGroup viewGroup) {
            view = inflter.inflate(R.layout.layout_spinner_row, null);
            ImageView icon = (ImageView) view.findViewById(R.id.imageView);
            TextView names = (TextView) view.findViewById(R.id.textView);
            icon.setImageResource(icons[i]);
            names.setText(countryNames[i]);
            return view;
        }
    }

}
```

2.10. New Modes Layout

This Layout is used here to enable user to change/edit the keywords. By using drop down menu, user can choose the mode and he can change the keyword by using the edit text provided below.

new_modes_layouts.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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"
    tools:context="com.vivartha.modechanger.NewModesActivity">

    <Spinner
        android:id="@+id/simpleSpinner"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="50dp" />

    <EditText
        android:id="@+id/et_mode_val"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="80dp"
        android:layout_marginRight="80dp"
        android:layout_below="@+id/simpleSpinner"
        android:layout_marginTop="50dp"/>

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Save"
        android:layout_marginLeft="80dp"
        android:layout_marginRight="80dp"
        android:layout_marginTop="50dp"
        android:layout_below="@+id/et_mode_val"
        android:id="@+id/new_options"/>

</RelativeLayout>
```

2.11. Database Helper

This class is used to integrate SQLite database into the application and store the user details and keywords.

DatabaseHelper.java

```
package com.vivartha.modechanger;

import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;

/**
 * Created by Vikas.
 * Here a table is created which stores the user details and the keywords.
 */

public class DataBaseHelper extends SQLiteOpenHelper {
```

```
    /**
     * The name of the database.
     */
    public static final String DB_NAME = "mode.db";

    /**
     * The DB's version number. This needs to be increased on schema
changes.
     */
    public static final int DB_VERSION = 1;
    private static final String TAG = "ServicePulseDbHelper";

    /**
     * Singleton instance of {@link DataBaseHelper}.
     */
    private static DataBaseHelper instance = null;
    private SQLiteDatabase db;

    /**
     * @return the {@link DataBaseHelper} singleton.
     */
    public static DataBaseHelper getInstance() {
        if (instance != null) {
            return instance;
        } else {
            return new DataBaseHelper();
        }
    }

    private DataBaseHelper() {
        super(AppController.getInstance().getApplicationContext(), DB_NAME,
null, DB_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase) {
        String modes_table = "CREATE TABLE modes ("
                + "row_id INTEGER PRIMARY KEY NOT NULL,"
                + "mode_name TEXT,"
                + "mode_value TEXT,"
                + "other1 TEXT,"
                + "other2 TEXT,"
                + "other3 TEXT,"
                + "other4 TEXT)";
        sqLiteDatabase.execSQL(modes_table);

        String users_table = "CREATE TABLE users ("
                + "row_id INTEGER PRIMARY KEY NOT NULL,"
                + "name TEXT,"
                + "phone TEXT,"
                + "email TEXT,"
                + "password TEXT,"
                + "other3 TEXT,"
                + "other4 TEXT)";
        sqLiteDatabase.execSQL(users_table);

        String activity_log_table = "CREATE TABLE activity_log ("
                + "row_id INTEGER PRIMARY KEY NOT NULL,"
                + "msg_desc TEXT,"
                + "date_time TEXT,"
```

```
                + "op_name TEXT,"
                + "other1 TEXT,"
                + "other2 TEXT,"
                + "other3 TEXT)";
        sqLiteDatabase.execSQL(activity_log_table);
    }


    public ArrayList<ActivityLogModel> getActivityLogRecords() {
        ArrayList<ActivityLogModel> mList = new ArrayList();

        int count = 0;
        String query;
        Cursor mCursor = null;
        try {
            query = "select * from activity_log order by row_id desc";
            db = getReadableDatabase();
            mCursor = db.rawQuery(query, null);
            if (mCursor.getCount() > 0) {
                while (mCursor.moveToNext()) {
                    ActivityLogModel model = new ActivityLogModel();
                    model.activity_log_desc =
mCursor.getString(mCursor.getColumnIndex("msg_desc"));
                    model.date_time =
mCursor.getString(mCursor.getColumnIndex("date_time"));
                    model.option_code =
mCursor.getString(mCursor.getColumnIndex("op_name"));
                    mList.add(model.addItems(model));
                }
            }
        }
        catch (Exception e) {
            e.printStackTrace();
        }finally {
            mCursor.close();
        }
        return mList;
    }


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

    }
    public String getValueByOptionName(String option){
        db = getReadableDatabase();
        Cursor c = null;
        try {
            c = db.rawQuery("SELECT mode_value FROM modes WHERE mode_name
='" + option + "'", null);
            if (c != null)
                if (c.getCount() > 0){
                    c.moveToFirst();
                    return  c.getString(0);
                }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (c != null && !c.isClosed()) c.close();
        }
        return "";
```

```
    }
    public String getOptionNameByValue(String value){
        db = getReadableDatabase();
        Cursor c = null;
        try {
            c = db.rawQuery("SELECT mode_name FROM modes WHERE mode_value
='" + value + "'", null);
            if (c != null)
                if (c.getCount() > 0){
                    c.moveToFirst();
                    return  c.getString(0);
                }else{
                    return value;
                }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (c != null && !c.isClosed()) c.close();
        }
        return "";
    }
    public boolean isValidUser(String email, String passowrd){
        db = getReadableDatabase();
        Cursor c = null;
        try {
            c = db.rawQuery("SELECT * FROM users WHERE email ='" + email +
"' AND password = '"+passowrd+"'", null);
            if (c != null)
                if (c.getCount() > 0){
                   return true;
                }else{
                    return false;
                }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (c != null && !c.isClosed()) c.close();
        }
        return false;
    }
    public void updateValueByOption(String optionName, String new_val) {
        ContentValues cv = new ContentValues();
        db = getWritableDatabase();
        db.rawQuery("delete from modes where mode_name = '"+optionName+"'",
null);
        try {
            db.beginTransaction();
            cv.put("mode_name", optionName);
            cv.put("mode_value", new_val);

            db.insert("modes", null, cv);
            db.setTransactionSuccessful();

        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            db.endTransaction();
            db.close();
        }
    }
    public void insertActLog(String msg_desc, String op_name) {
```

```
        db = getWritableDatabase();
        ContentValues cv = new ContentValues();
        try {
            db.beginTransaction();
            cv.put("msg_desc", msg_desc);
            cv.put("op_name", op_name);
            cv.put("date_time", getDateTime());
            db.insert("activity_log", null, cv);
            db.setTransactionSuccessful();
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            db.endTransaction();
            db.close();
        }
    }
    private String getDateTime() {
        DateFormat dateFormat = new SimpleDateFormat("yyyy/MM/dd
HH:mm:ss");
        Date date = new Date();
        return dateFormat.format(date);
    }
    public void inserNewUser(String name, String email,String  phone,String
password ) {
        db = getWritableDatabase();
        ContentValues cv = new ContentValues();
        try {
            db.beginTransaction();
            cv.put("name", name);
            cv.put("phone", phone);
            cv.put("email", email);
            cv.put("password", password);
            db.insert("users", null, cv);
            db.setTransactionSuccessful();
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            db.endTransaction();
            db.close();
        }
    }


}
```

2.12. User Registration

This enables user to register by giving some details such as name, email, password, phone number.

Registration.java

```
package com.vivartha.modechanger;
/**
 * Created by vikas
 * Allows user to register.
 */

import android.app.Activity;
import android.content.ContentValues;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.EditText;
```

```
public class Registration extends Activity {

    EditText et_name, et_password ,et_email, et_phonenumber;

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

        et_name = findViewById(R.id.et_name);
        et_password = findViewById(R.id.et_password);
        et_email = findViewById(R.id.et_email);
        et_phonenumber = findViewById(R.id.et_phonenumber);


        findViewById(R.id.submit1).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {

                // Store values at the time of the login attempt.
                String email = et_email.getText().toString();
                String password = et_password.getText().toString();
                String name = et_name.getText().toString();
                String phone = et_phonenumber.getText().toString();

                // Check for a valid password, if the user entered one.
                if (!TextUtils.isEmpty(name) && !MyUtils.isNameValid(name))
{
                    et_name.setError("Name must be minimum 5 characters");
                    return;
                }


                // Check for a valid password, if the user entered one.
                if (!TextUtils.isEmpty(phone) &&
!MyUtils.isPhoneValid(phone)) {
                    et_phonenumber.setError("Please enter valid phone
number");
                    return;
                }

                // Check for a valid password, if the user entered one.
                if (!TextUtils.isEmpty(password) &&
!MyUtils.isPasswordValid(password)) {

et_password.setError(getString(R.string.error_invalid_password));
                    return;
                }

                // Check for a valid email address.
                if (TextUtils.isEmpty(email)) {

et_email.setError(getString(R.string.error_field_required));
                   return;
                } else if (!isEmailValid(email)) {

et_email.setError(getString(R.string.error_invalid_email));
                    return;
                }
```

```
                DataBaseHelper.getInstance().inserNewUser(name, email,
phone, password);
                finish();
            }
        });

    }

    private boolean isEmailValid(String email) {
        //TODO: Replace this with your own logic
        return email.contains("@");
    }

}
```

Activity_registration.xml

```
<?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:layout_gravity="bottom"
    android:background="@drawable/plain_bg"
    android:orientation="vertical"
    >

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginLeft="15dp"
        android:layout_marginRight="15dp"
        android:gravity="center"
        android:orientation="vertical">

        <TextView
            android:layout_width="match_parent"
            android:layout_gravity="center"
            android:gravity="center"
            android:textColor="@android:color/white"
            android:textSize="20sp"
            android:textStyle="bold"
            android:layout_height="wrap_content"
            android:text="Registration" />

        <EditText
            android:id="@+id/et_name"
            android:layout_width="match_parent"
            android:layout_height="70dp"
            android:layout_margin="2dp"
            android:hint="Enter Name"
            android:textColorHint="@android:color/white" />

        <EditText
            android:id="@+id/et_password"
            android:layout_width="match_parent"
            android:layout_height="70dp"
            android:hint="Enter Password"
            android:inputType="textPassword"
            android:textColorHint="@android:color/white"/>
```

```
        <EditText
            android:id="@+id/et_email"
            android:layout_width="match_parent"
            android:layout_height="70dp"
            android:hint="Enter Email"
            android:inputType="textEmailAddress"
            android:textColorHint="@android:color/white"/>

        <EditText
            android:id="@+id/et_phonenumber"
            android:layout_width="match_parent"
            android:layout_height="70dp"
            android:hint="Enter PhoneNumber"
            android:inputType="phone"
            android:maxLength="10"
            android:textColorHint="@android:color/white"/>

        <Button
            android:id="@+id/submit1"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Register" />

    </LinearLayout>


</LinearLayout>
```

2.13. AppPreferences

This class runs in the background and ensures whether the data is stored in the correct place or not.

AppPreferences.java

```
package com.vivartha.modechanger;
/**
 * created by Vikas.
 * This is used to store and delete the data across the application.
 * It sets the shared preferences.
 */

import android.app.Activity;
import android.content.Context;
import android.content.SharedPreferences;
import android.content.SharedPreferences.Editor;


/**Saving data across the application */
public class AppPreferences {

   private static final String APP_SHARED_PREFS = "com.viv.mode";
   private SharedPreferences appSharedPrefs;
   private Editor prefsEditor;

   /** Saving data in shared preferences which will store life time of
Application */
   public AppPreferences(Context context)
   {
      this.appSharedPrefs = context.getSharedPreferences(APP_SHARED_PREFS,
Activity.MODE_PRIVATE);
      this.prefsEditor = appSharedPrefs.edit();
   }
```

```
    /*
     *    Delete the all the preferences
     */
    public void deletePref() {
        this.prefsEditor.clear();
        this.prefsEditor.commit();
    }

    public void saveLoginStatte(int contactsCount) // 0 - Logg off, 1 -
login success
    {
        prefsEditor.putInt("login_state", contactsCount);
        prefsEditor.commit();
    }

    public int getLoginState() {
        return appSharedPrefs.getInt("login_state",0);
    }


    public void savePinPadState(int contactsCount) // 0 - Logg off, 1 -
login success
    {
        prefsEditor.putInt("pin_pad_state", contactsCount);
        prefsEditor.commit();
    }

    public int getPinPadState() {
        return appSharedPrefs.getInt("pin_pad_state",0);
    }


    public void savePin(String pin){
        prefsEditor.putString("pin", pin);
        prefsEditor.commit();
    }

    public String getPin(){
        return appSharedPrefs.getString("pin", "");
    }


    public void saveLatitude(String latitude){
        prefsEditor.putString("lat", latitude);
        prefsEditor.commit();
    }

    public String getLatitude(){
        return appSharedPrefs.getString("lat", "00.00");
    }

    public void saveLongitude(String longitude){
        prefsEditor.putString("longitude", longitude);
        prefsEditor.commit();
    }

    public String getLongitude(){
        return appSharedPrefs.getString("longitude","00.00");
    }
```

```
public void saveUserName(String name){ // otp
   prefsEditor.putString("user_name", name);
   prefsEditor.commit();
}

public String getUserName(){ //otp
   return appSharedPrefs.getString("user_name","");
}

public void savePassword(String name){ // otp
   prefsEditor.putString("passwd", name);
   prefsEditor.commit();
}

public String getPassword(){ //otp
   return appSharedPrefs.getString("passwd","");
}

public void saveUserId(int id){ //phone no
   prefsEditor.putInt("user_id", id);
   prefsEditor.commit();
}

public int getUserid(){// phone no
   return appSharedPrefs.getInt("user_id",0);
}


public void saveUserPhone(String id){ //phoneno
   prefsEditor.putString("phoneno", id);
   prefsEditor.commit();
}

public String getUserPhone(){// phone no
   return appSharedPrefs.getString("phoneno","");
}



public void saveManifestoUrl(String url) {
   prefsEditor.putString("manifesto", url);
   prefsEditor.commit();
}

public String getManifestoUrl(){
   return appSharedPrefs.getString("manifesto", "");
}

public void saveHistoryUrl(String localUri) {
   prefsEditor.putString("history", localUri);
   prefsEditor.commit();
}
public String getHistoryUrl(){
   return appSharedPrefs.getString("history","");
}

public void saveLogOutRequired(int val) {
   prefsEditor.putInt("LogOutRequired", val);
   prefsEditor.commit();
}
public int getLogOutRequired(){
```

```
        return appSharedPrefs.getInt("LogOutRequired",0);
    }



    public void saveOrgId(String data) {
        prefsEditor.putString("org_id", data);
        prefsEditor.commit();
    }

    public String getOrgId(){
        return appSharedPrefs.getString("org_id","0");
    }

// public void savePushState(int i) {
//     prefsEditor.putInt(RegistrationIntentService.SENT_TOKEN_TO_SERVER,
i);
//     prefsEditor.commit();
// }
//
// public int getPushState(){
//     return
appSharedPrefs.getInt(RegistrationIntentService.SENT_TOKEN_TO_SERVER, 0);
// }

    public void saveFirebaseToken(String token) {
        prefsEditor.putString("fire_base_token", token);
        prefsEditor.commit();
    }

    public String getFirebaseToken(){
        return appSharedPrefs.getString("fire_base_token","");
    }

    public void saveHomeLat(String data) {
        prefsEditor.putString("lat", data);
        prefsEditor.commit();
    }

    public String getHomeLat(){
        return appSharedPrefs.getString("lat","0");
    }


    public void saveHomeLang(String data) {
        prefsEditor.putString("lang", data);
        prefsEditor.commit();
    }

    public String getHomeLang(){
        return appSharedPrefs.getString("lang","0");
    }


    public void saveFCMState(int state) {
        prefsEditor.putInt("state", state);
        prefsEditor.commit();
    }

    public int getFCMState(){
        return appSharedPrefs.getInt("state",0);
```

```
    }


}
```

2.14. MyUtils

This class is used to validate the input from the user(i.e., Validating user credentials).

```
package com.vivartha.modechanger;

import java.util.regex.Pattern;
/**
*@author revanth
*/

public class MyUtils {


    public static boolean isEmailValid(String email) {
        //TODO: Replace this with your own logic
        String emailPattern = "[a-zA-Z0-9._-]+@[a-z]+\\.+[a-z]+";
        if (email.matches(emailPattern))
        {
            return true;
        }
        else
        {
            return false;
        }
    }

/**

 ^                 # start-of-string
 (?=.*[0-9])       # a digit must occur at least once
 (?=.*[a-z])       # a lower case letter must occur at least once
 (?=.*[A-Z])       # an upper case letter must occur at least once
 (?=.*[@#$%^&+=])  # a special character must occur at least once
 (?=\S+$)          # no whitespace allowed in the entire string
 .{8,}             # anything, at least eight places though
 $                 # end-of-string

 * **/
    public static boolean isPasswordValid(String password) {
       String password_pattern = "^(?=.*[0-9])(?=.*[a-z])(?=.*[A-
Z])(?=.*[@#$%^&+=])(?=\\S+$).{8,}$";
        if(password.matches(password_pattern)){
            return true;
        }else{
            return false;
        }

    }


    /**  Length >=3
     Valid characters: a-z, A-Z, 0-9 **/

    public static boolean isNameValid(String name) {
        String regex = "^[a-zA-Z0-9._-]{3,}$";
```

```
        if(name.matches(regex)){return  true;}else{return false;}

    }

    public static  boolean isPhoneValid(String phone) {
        boolean check=false;
        if(!Pattern.matches("[a-zA-Z]+", phone)) {
            if(phone.length() < 10 || phone.length() > 13) {
                // if(phone.length() != 10) {
                check = false;

            } else {
                check = true;
            }
        } else {
            check=false;
        }
        return check;
    }

}
```

2.15. Unit Test Cases

Test cases ensure that developed code is working properly. A **TEST CASE** is a set of conditions or variables under which a tester will determine whether a system under **test** satisfies **requirements** or works correctly. The process of developing **test cases** can also help find problems in the **requirements** or design of an application.

MyUtilsTest.java

```
package com.vivartha.modechanger;

import org.junit.Test;

import static org.junit.Assert.*;
/**
*created by vikas.
*Last Modified by siri,sai,sai,revanth.
*Testcases for email and password validations.
*/

public class MyUtilsTest {

    @Test
    public void isEmailValidEmail1() {
        boolean expected =  true;
        boolean output;
        MyUtils myUtils = new MyUtils();
        output = myUtils.isEmailValid("saikrishna.andydev@gmail.com");
        assertEquals(expected, output);
    }

    @Test
    public void isEmailValidEmail2() {
        boolean expected =  true;
        boolean output;
        MyUtils myUtils = new MyUtils();
        output = myUtils.isEmailValid("saikrishnaandydev@gmail.com");
        assertEquals(expected, output);

    }

    @Test
```

```
    public void isEmailValidEmail3() {
        boolean expected =  false;
        boolean output;

        MyUtils myUtils = new MyUtils();
        output = myUtils.isEmailValid("saikrishnaandydevgmail.com");//no @
        assertEquals(expected, output);

    }


    @Test
    public void isEmailValidEmail4() {
        boolean expected =  false;
        boolean output;

        MyUtils myUtils = new MyUtils();
        output = myUtils.isEmailValid("saikrishnaandydev@gmailcom");//no
.com
        assertEquals(expected, output);

    }



    @Test
    public void isEmailValidEmail5() {
        boolean expected =  true;
        boolean output;

        MyUtils myUtils = new MyUtils();
        output = myUtils.isEmailValid("saikrishnaandydev@gmail.in");// .in
        assertEquals(expected, output);

    }

    @Test
    public void isEmailValidEmail6() {
        boolean expected =  true;
        boolean output;

        MyUtils myUtils = new MyUtils();
        output = myUtils.isEmailValid("krishna@gmail.com");// small length
        assertEquals(expected, output);

    }


    @Test
    public void isPasswordValid() {

        boolean expected = true;
        MyUtils myUtils = new MyUtils();
        boolean output = myUtils.isPasswordValid("Saikrishna@123");
        assertEquals(expected, output);

    }

    @Test
    public void isPasswordVali2() {
```

```
    boolean expected = true;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isPasswordValid("sAikrishna@123");
    assertEquals(expected, output);

}


@Test
public void isPasswordVali3() {

    boolean expected = true;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isPasswordValid("saikrishnA#123");
    assertEquals(expected, output);

}
@Test
public void isPasswordVali4() {

    boolean expected = true;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isPasswordValid("saiKrishna#0");
    assertEquals(expected, output);

}

@Test
public void isPasswordVali5() {

    boolean expected = false;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isPasswordValid("");
    assertEquals(expected, output);

}
@Test
public void isPasswordVali6() {

    boolean expected = false;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isPasswordValid("saikrishna");
    assertEquals(expected, output);

}

@Test
public void isPasswordVali7() {

    boolean expected = false;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isPasswordValid("123456789");
    assertEquals(expected, output);

}

@Test
public void isPasswordVali8() {

    boolean expected = false;
    MyUtils myUtils = new MyUtils();
```

```
    boolean output = myUtils.isPasswordValid("MSDHONI#123");
    assertEquals(expected, output);

}


@Test
public void isPasswordVali9() {

    boolean expected = true;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isPasswordValid("MsDhoni#123");
    assertEquals(expected, output);

}

// Registration


@Test
public void isNameValid1(){
    boolean expected = true;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isNameValid("SaiKrishna");
    assertEquals(expected, output);
}

@Test
public void isNameValid2(){
    boolean expected = false;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isNameValid("");
    assertEquals(expected, output);
}


@Test
public void isNameValid3(){
    boolean expected = false;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isNameValid("SK");
    assertEquals(expected, output);
}



@Test
public void isNameValid4(){
    boolean expected = true;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isNameValid("SaiKrishna12");
    assertEquals(expected, output);
}


@Test
public void isNameValid5(){
    boolean expected = true;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isNameValid("Sai");
    assertEquals(expected, output);
```

```
}


@Test
public void isNameValid6(){
    boolean expected = true;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isNameValid("SaiK");
    assertEquals(expected, output);
}


@Test
public void isNameValid7(){
    boolean expected = false;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isNameValid("S");
    assertEquals(expected, output);
}

@Test
public void isValidPhoneNumber(){
    boolean expected = false;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isPhoneValid(" ");
    assertEquals(expected, output);
}

@Test
public void isValidPhoneNumber1(){
    boolean expected = false;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isPhoneValid("123");
    assertEquals(expected, output);
}

@Test
public void isValidPhoneNumber2(){
    boolean expected = true;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isPhoneValid("9848022338");
    assertEquals(expected, output);
}

@Test
public void isValidPhoneNumber3(){
    boolean expected = false;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isPhoneValid("784569321");
    assertEquals(expected, output);
}

@Test
public void isValidPhoneNumber4(){
    boolean expected = false;
    MyUtils myUtils = new MyUtils();
    boolean output = myUtils.isPhoneValid("00000000");
    assertEquals(expected, output);
}

@Test
```

```
    public void isValidPhoneNumber5(){
        boolean expected = false;
        MyUtils myUtils = new MyUtils();
        boolean output = myUtils.isPhoneValid("1234567");
        assertEquals(expected, output);
    }

    @Test
    public void isValidPhoneNumber6(){
        boolean expected = false;
        MyUtils myUtils = new MyUtils();
        boolean output = myUtils.isPhoneValid("12345");
        assertEquals(expected, output);
    }

    @Test
    public void isValidPhoneNumber7(){
        boolean expected = true;
        MyUtils myUtils = new MyUtils();
        boolean output = myUtils.isPhoneValid("9985785724");
        assertEquals(expected, output);
    }

    @Test
    public void isValidPhoneNumber8(){
        boolean expected = true;
        MyUtils myUtils = new MyUtils();
        boolean output = myUtils.isPhoneValid("998-578-5724");
        assertEquals(expected, output);
    }

    @Test
    public void isValidPhoneNumber9(){
        boolean expected = true;
        MyUtils myUtils = new MyUtils();
        boolean output = myUtils.isPhoneValid("998 578 5724");
        assertEquals(expected, output);
    }

    @Test
    public void isValidPhoneNumber10(){
        boolean expected = false;
        MyUtils myUtils = new MyUtils();
        boolean output = myUtils.isPhoneValid("143143");
        assertEquals(expected, output);
    }

}
```

2.16. Activity Log Model

This class is used to display the activity log of the application. It displays the log, when the application is used to change the mode.

```
package com.vivartha.modechanger;

import java.io.Serializable;

/**
 * Created by sai krishna
 * This is an activity log controller
 * It triggers the changes made, and stores that into database
 * The stored content will be displayed to the user once the user opens the
activity.
```

```
 * since this is the launch activity, the user will be navigated to this
once he enters the correct pin.
 **/

public class ActivityLogModel implements Serializable {

    public  String option_code = "";
    public  String activity_log_desc = "";
    public  String date_time = "";

    public ActivityLogModel(String option_code, String activity_log_desc,
String date_time) {
        this.option_code = option_code;
        this.activity_log_desc = activity_log_desc;
        this.date_time = date_time;
    }

    public ActivityLogModel() { }

    public ActivityLogModel addItems (ActivityLogModel mandalModel){
        return new ActivityLogModel(mandalModel.option_code,
                mandalModel.activity_log_desc, mandalModel.date_time);
    }

    public String getOption_code() {
        return option_code;
    }

    public void setOption_code(String option_code) {
        this.option_code = option_code;
    }

    public String getActivity_log_desc() {
        return activity_log_desc;
    }

    public void setActivity_log_desc(String activity_log_desc) {
        this.activity_log_desc = activity_log_desc;
    }

    public String getDate_time() {
        return date_time;
    }

    public void setDate_time(String date_time) {
        this.date_time = date_time;
    }
}
```

2.17. App controller

This class inherits the properties of the onCreate and Saved Instances to execute the application properly.

```
package com.vivartha.modechanger;

import android.app.Application;

/**
 * created by Revanth
 * implements onCreate and instance classes.
 */

public class AppController extends Application {
```

```
    public static final String TAG = AppController.class.getSimpleName();



    private static AppController mInstance;

    @Override
    public void onCreate() {
        super.onCreate();
        mInstance = this;
    }

    public static synchronized AppController getInstance() {
        return mInstance;
    }

}
```

2.18. GPS Tracker

This is used to store the recent location of the phone and then whenever the user requests for the location, it triggers the latest location and sends back to the user.
GPSTracker.java

```
package com.vivartha.modechanger;

/**
 * created by sai krishna.
 * Class to get the location coordinates.
 * It selects the recently triggered location and sends to the requested
user.
 */

import android.annotation.SuppressLint;
import android.app.AlertDialog;
import android.app.Service;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.os.IBinder;
import android.util.Log;

public class GPSTracker extends Service implements LocationListener {

    private final Context mContext;

    // flag for GPS status
    boolean isGPSEnabled = false;

    // flag for network status
    boolean isNetworkEnabled = false;

    // flag for GPS status
    boolean canGetLocation = false;

    Location location; // location
```

```
    public double latitude; // latitude
    public double longitude; // longitude

    // The minimum distance to change Updates in meters
    private static final long MIN_DISTANCE_CHANGE_FOR_UPDATES = 0; // 10
meters

    // The minimum time between updates in milliseconds
    private static final long MIN_TIME_BW_UPDATES = 6000; // 1 minute

    // Declaring a Location Manager
    protected LocationManager locationManager;

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

    @SuppressLint("MissingPermission")
    public Location getLocation() {
        try {
            locationManager = (LocationManager)
mContext.getSystemService(LOCATION_SERVICE);

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

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

            if (!isGPSEnabled && !isNetworkEnabled) {
                // no network provider is enabled
            } else {
                this.canGetLocation = true;
                if (isNetworkEnabled) {

locationManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER,
                            MIN_TIME_BW_UPDATES,
MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
                        if (locationManager != null) {
                            location =
locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
                            onLocationChanged(location);
                        }
                    }
                    // if GPS Enabled get lat/long using GPS Services
                    if (isGPSEnabled) {
                        if (location == null) {
                            locationManager.requestLocationUpdates(
LocationManager.GPS_PROVIDER,
                                    MIN_TIME_BW_UPDATES,
MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
                            if (locationManager != null) {
                                location =
locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER);
                                onLocationChanged(location);
                            }
                        }
                    }
```

```
                    return location;
                }
            } catch (Exception e) {
                e.printStackTrace();
            }
            return location;
        }
    public void stopUsingGPS(){
        if(locationManager != null){
            locationManager.removeUpdates(GPSTracker.this);
        }
    }

    public double getLatitude(){
        if(location != null){
            latitude = location.getLatitude();
        }
        return latitude;
    }

    /**
     * Function to get longitude
     * */
    public double getLongitude(){
        if(location != null){
            longitude = location.getLongitude();
        }
        // return longitude
        return longitude;
    }

    /**
     * Function to check GPS/wifi enabled
     * @return boolean
     * */
    public boolean canGetLocation() {
        return this.canGetLocation;
    }

    /**
     * Function to show settings alert dialog
     * On pressing Settings button will lauch Settings Options
     * */
    public void showSettingsAlert(){
        AlertDialog.Builder alertDialog = new
AlertDialog.Builder(mContext);
        // Setting Dialog Title
        alertDialog.setTitle("GPS is settings");
        // Setting Dialog Message
        alertDialog.setMessage("GPS is not enabled. Do you want to go to
settings menu?");
        // on pressing cancel button
        alertDialog.setNegativeButton("Cancel", new
DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) {
            dialog.cancel();
            }
        });
        // Showing Alert Message
        alertDialog.show();
    }
```

```
    @Override
    public void onLocationChanged(Location location) {
        if (location != null) {
           stopUsingGPS();
             latitude = location.getLatitude();
             longitude = location.getLongitude();
         }
    }

    @Override
    public void onProviderDisabled(String provider) {
    }

    @Override
    public void onProviderEnabled(String provider) {
    }

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

    @Override
    public IBinder onBind(Intent arg0) {
        return null;
    }

}
```

2.19. SimpleItem

This is mainly used for UserInterface purpose.

SimpleItem.java

```
package com.vivartha.modechanger;

import android.graphics.drawable.Drawable;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;


/**
 * Created by sai teja.
 * Class designed for user Interface.
 * */
public class SimpleItem extends DrawerItem<SimpleItem.ViewHolder> {

    private int selectedItemIconTint;
    private int selectedItemTextTint;

    private int normalItemIconTint;
    private int normalItemTextTint;

    private Drawable icon;
    private String title;
```

```
    public SimpleItem(Drawable icon, String title) {
        this.icon = icon;
        this.title = title;
    }

    @Override
    public ViewHolder createViewHolder(ViewGroup parent) {
        LayoutInflater inflater = LayoutInflater.from(parent.getContext());
        View v = inflater.inflate(R.layout.item_option, parent, false);
        return new ViewHolder(v);
    }

    @Override
    public void bindViewHolder(ViewHolder holder) {
        holder.title.setText(title);
        holder.icon.setImageDrawable(icon);

        holder.title.setTextColor(isChecked ? selectedItemTextTint :
normalItemTextTint);
        holder.icon.setColorFilter(isChecked ? selectedItemIconTint :
normalItemIconTint);
    }

    public SimpleItem withSelectedIconTint(int selectedItemIconTint) {
        this.selectedItemIconTint = selectedItemIconTint;
        return this;
    }

    public SimpleItem withSelectedTextTint(int selectedItemTextTint) {
        this.selectedItemTextTint = selectedItemTextTint;
        return this;
    }

    public SimpleItem withIconTint(int normalItemIconTint) {
        this.normalItemIconTint = normalItemIconTint;
        return this;
    }

    public SimpleItem withTextTint(int normalItemTextTint) {
        this.normalItemTextTint = normalItemTextTint;
        return this;
    }

    static class ViewHolder extends DrawerAdapter.ViewHolder {

        private ImageView icon;
        private TextView title;

        public ViewHolder(View itemView) {
            super(itemView);
            icon = (ImageView) itemView.findViewById(R.id.icon);
            title = (TextView) itemView.findViewById(R.id.title);
        }
    }
}
```

SpaceItem.java

```
package com.vivartha.modechanger;

import android.content.Context;
```

```
import android.view.View;
import android.view.ViewGroup;

/**
 * Created by sai teja.
 * class for user interface.
 */

public class SpaceItem extends DrawerItem<SpaceItem.ViewHolder> {

    private int spaceDp;

    public SpaceItem(int spaceDp) {
        this.spaceDp = spaceDp;
    }

    @Override
    public ViewHolder createViewHolder(ViewGroup parent) {
        Context c = parent.getContext();
        View view = new View(c);
        int height = (int) (c.getResources().getDisplayMetrics().density *
spaceDp);
        view.setLayoutParams(new ViewGroup.LayoutParams(
                ViewGroup.LayoutParams.MATCH_PARENT,
                height));
        return new ViewHolder(view);
    }

    @Override
    public void bindViewHolder(ViewHolder holder) {

    }

    @Override
    public boolean isSelectable() {
        return false;
    }

    static class ViewHolder extends DrawerAdapter.ViewHolder {

        public ViewHolder(View itemView) {
            super(itemView);
        }
    }
}
```

Values

The values Directory contains already defined values such as id, strings, colors, dimens, style. We can directly inherit these values into the required classes.

colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#008577</color>
    <color name="colorPrimaryDark">#00574B</color>
    <color name="colorAccent">#D81B60</color>
</resources>
```

dimens.xml

```
<resources>
    <!-- Default screen margins, per the Android Design guidelines. -->
    <dimen name="activity_horizontal_margin">16dp</dimen>
    <dimen name="activity_vertical_margin">16dp</dimen>
    <dimen name="fab_margin">16dp</dimen>
</resources>
```

ids.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <item name="w1" type="id"></item>
    <item name="w2" type="id"></item>
    <item name="v1" type="id"></item>
    <item name="im1" type="id"></item>
    <item name="r1" type="id"></item>
    <item name="v2" type="id"></item>
    <item name="im2" type="id"></item>
    <item name="r2" type="id"></item>
    <item name="v3" type="id"></item>
    <item name="im3" type="id"></item>
    <item name="r3" type="id"></item>
    <item name="v4" type="id"></item>
    <item name="im4" type="id"></item>
    <item name="r4" type="id"></item>
    <item name="r5" type="id"></item>
</resources>
```

strings.xml

```
<resources>
    <string name="app_name">ModeChanger</string>
    <!-- Strings related to login -->
    <string name="prompt_email">Email</string>
    <string name="prompt_password">Password (optional)</string>
    <string name="action_sign_in">Sign in or register</string>
    <string name="action_sign_in_short">Sign in</string>
    <string name="error_invalid_email">This email address is
invalid</string>
    <string name="error_invalid_password">This password is too
short</string>
    <string name="error_incorrect_password">This password is
incorrect</string>
    <string name="action_settings">Settings</string>
    <string name="error_field_required">This field is required</string>
    <string name="permission_rationale">"Contacts permissions are needed
for providing email
        completions."
    </string>
    <string name="title_activity_home">homeActivity</string>
```

```
    <string name="title_activity_about_us">about_us</string>
    <string name="title_activity_home_">Home_Activity</string>
    <string name="title_activity_splash__screen">Splash_Screen</string>
</resources>
```

styles.xml

```
<resources>

    <!-- Base application theme. -->
    <style name="AppTheme" parent="Theme.AppCompat.NoActionBar">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>
    </style>

    <style name="Theme.AppCompat.NoActionBar">
        <item name="windowActionBar">false</item>
        <item name="windowNoTitle">true</item>
    </style>

    <style name="AppTheme.AppBarOverlay"
parent="Theme.AppCompat.NoActionBar" />

    <style name="AppTheme.PopupOverlay"
parent="Theme.AppCompat.NoActionBar" />
    <style name = "NoActionBar" parent = "@android:style/Theme.Holo.Light">
        <item name = "android:windowActionBar">false</item>
        <item name = "android:windowNoTitle">true</item>
    </style>
    <style name="SplashScreenTheme"
parent="Theme.AppCompat.Light.DarkActionBar">
        <item name ="android:windowBackground">
@drawable/splashscreen</item>
    </style>

</resources>
```