**Sanika Kadam**

**2124UCSF1112**

**ASSIGNMENT NO.9**

1. **Introduction**

Send SMS Program: Write a program to send SMS messages. For instance, create a messaging app where users can input a phone number and message, then click "Send" to deliver the SMS.

1. **2. Tools & Technologies Used**

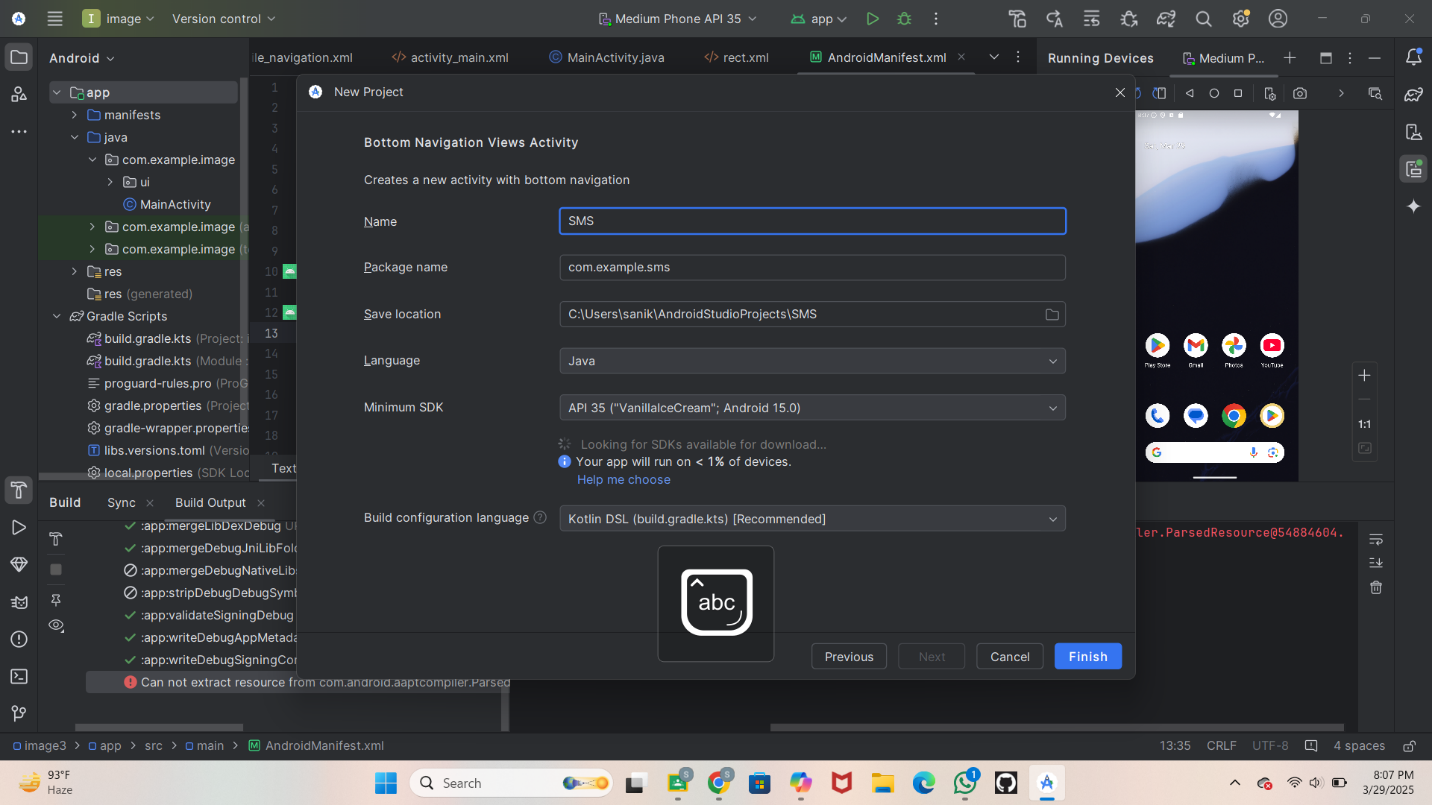
* Android Studio
* Language-Java

**3. Procedure & Steps**

**Step 1: Create a New Project**

* Open Android Studio and create a new project.
* Select second page activity by app-activity-page you selected
* Set the project name(SMS) and package name.(com.example.sms)
* Select the programming language (Java).

**Screenshot:**



**Step 2: Designing the UI**

* Open activity\_main.xml and design the layout using XML.
* Add UI components such as TextView, EditText, Button, etc.

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:gravity="center"

android:padding="15sp"

tools:context="MainActivity">

<EditText

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Enter Phone Number"

android:id="@+id/editText"

android:padding="15sp"

android:maxLength="10"

android:inputType="phone"

android:background="@drawable/rect"

android:layout\_marginTop="15dp"/>

<EditText

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/editTextMessage"

android:hint="Enter Message"

android:padding="15sp"

android:inputType="textMultiLine"

android:lines="7"

android:background="@drawable/rect"

android:layout\_marginTop="15dp"/>

<Button

android:id="@+id/btnSent"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="30dp"

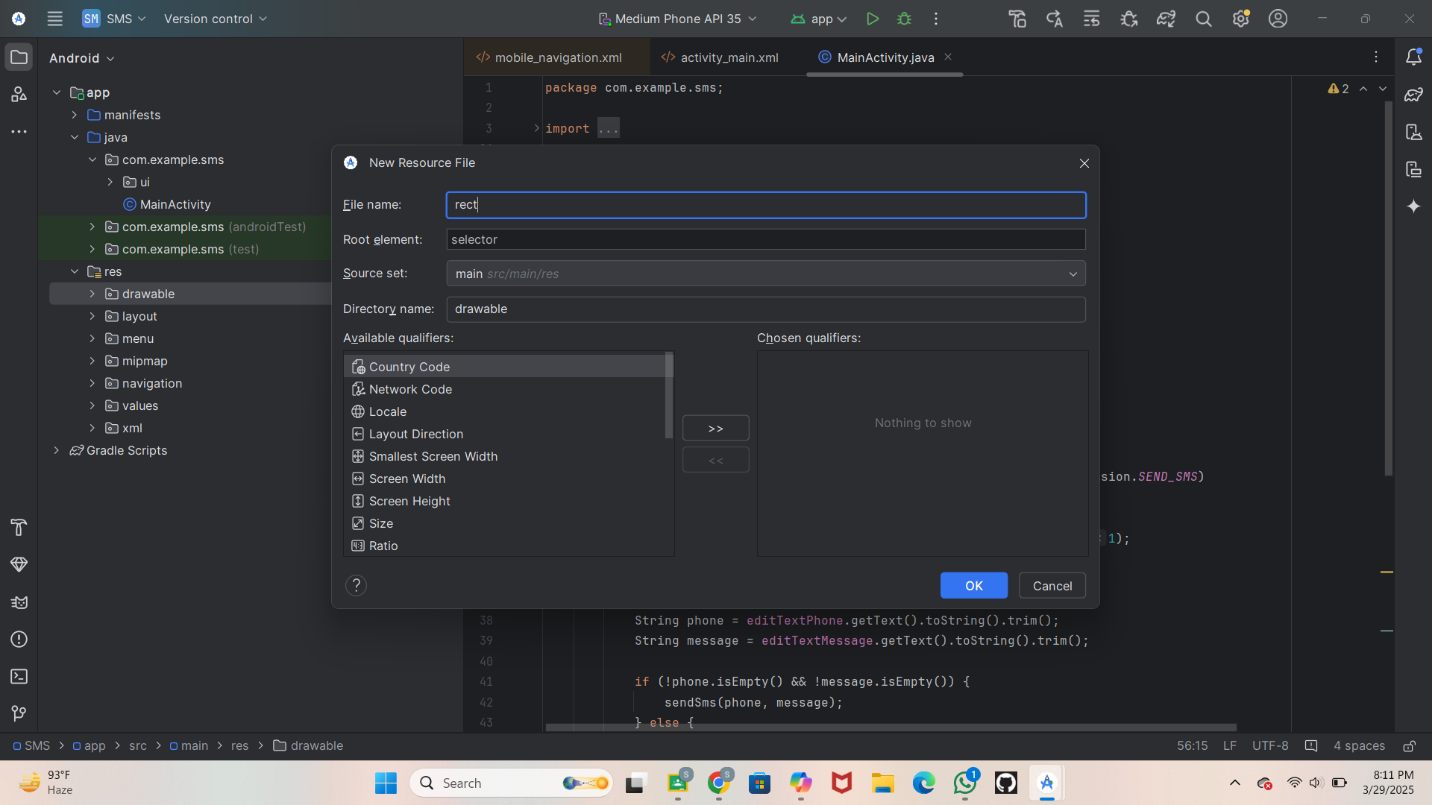
android:text="Send SMS" />

</LinearLayout>

**Step:3**

In Left side go on res then drawable Click on drawble

Set file name as rect

* 
* Write a code in rect
* <?xml version="1.0" encoding="utf-8"?>  
  <shape xmlns:android="http://schemas.android.com/apk/res/android">  
   <solid android:color="@android:color/white" />  
   <stroke  
   android:width="1dp"  
   android:color="@android:color/black" />  
   <corners android:radius="5dp" />  
  </shape>

**Step 4: :** Writing the Code

* Open MainActivity.java or MainActivity.kt.
* Implement functionality such as button clicks, form validation, etc.
* Use necessary Android components like Intents, RecyclerView, Fragments, etc.
* package com.example.sms;

import android.Manifest;

import android.content.pm.PackageManager;

import android.os.Bundle;

import android.telephony.SmsManager;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.app.ActivityCompat;

import androidx.core.content.ContextCompat;

public class MainActivity extends AppCompatActivity {

private EditText editTextPhone, editTextMessage;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main); // Corrected

editTextPhone = findViewById(R.id.editText); // Corrected

editTextMessage = findViewById(R.id.editTextMessage);

Button btnSend = findViewById(R.id.btnSent);

// Check SMS permission

if (ContextCompat.checkSelfPermission(this, Manifest.permission.SEND\_SMS)

!= PackageManager.PERMISSION\_GRANTED) {

ActivityCompat.requestPermissions(this,

new String[]{Manifest.permission.SEND\_SMS}, 1);

}

btnSend.setOnClickListener(v -> {

String phone = editTextPhone.getText().toString().trim();

String message = editTextMessage.getText().toString().trim();

if (!phone.isEmpty() && !message.isEmpty()) {

sendSms(phone, message);

} else {

Toast.makeText(MainActivity.this, "Please enter phone number and message!", Toast.LENGTH\_SHORT).show();

}

});

}

private void sendSms(String phone, String message) {

try {

SmsManager smsManager = SmsManager.getDefault();

smsManager.sendTextMessage(phone, null, message, null, null);

Toast.makeText(this, "SMS sent successfully!", Toast.LENGTH\_SHORT).show();

} catch (Exception e) {

Toast.makeText(this, "Failed to send SMS!", Toast.LENGTH\_SHORT).show();

e.printStackTrace();

}

}

**Step 5:**

* Write a Code In Android Mniefest

<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android">

<!-- Permissions -->

<uses-permission android:name="android.permission.SEND\_SMS" />

<uses-feature android:name="android.hardware.telephony" android:required="true" />

<application

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/Theme.AppCompat.Light">

<activity

android:name=".MainActivity"

android:exported="true"

android:label="@string/app\_name">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

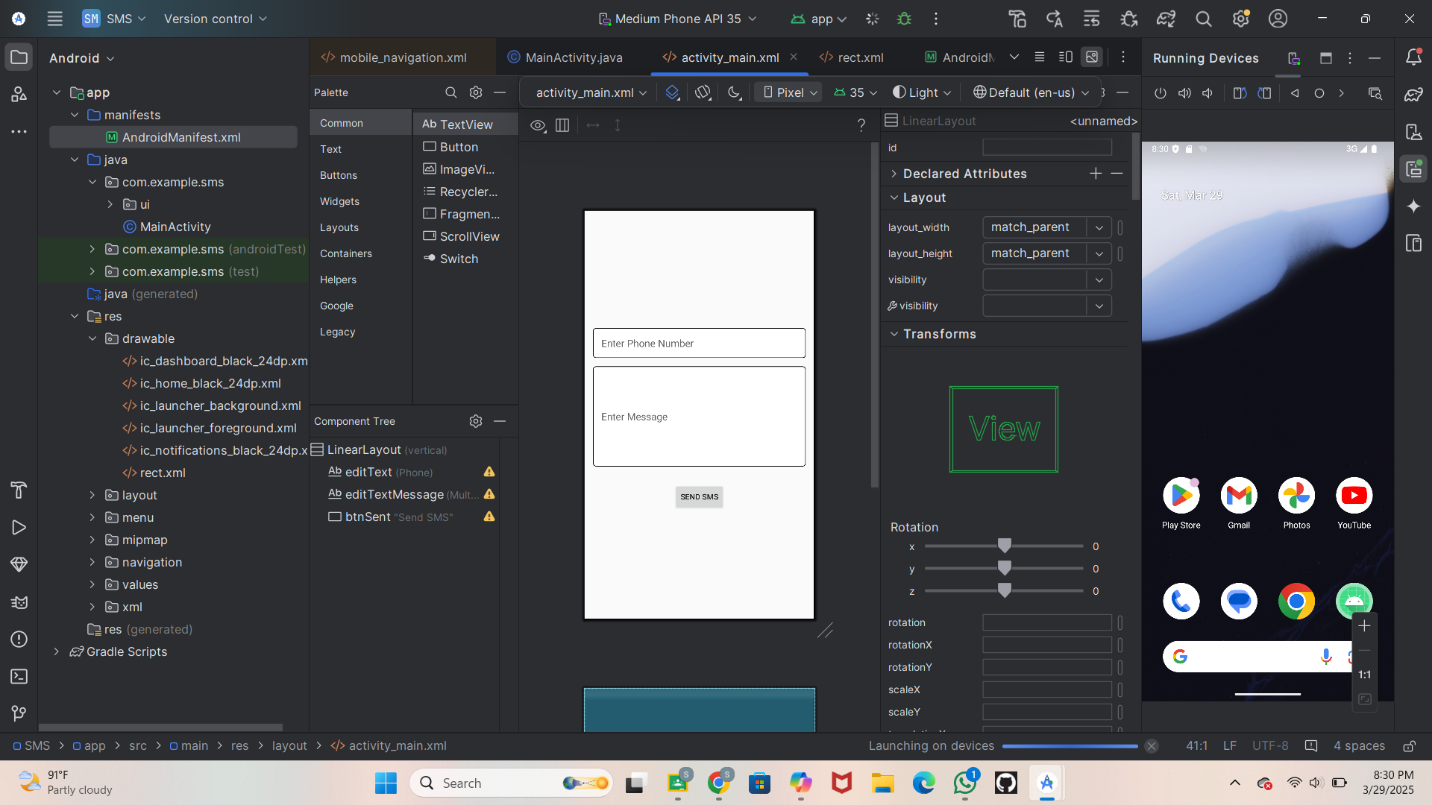
</activity>

</application>

</manifest>

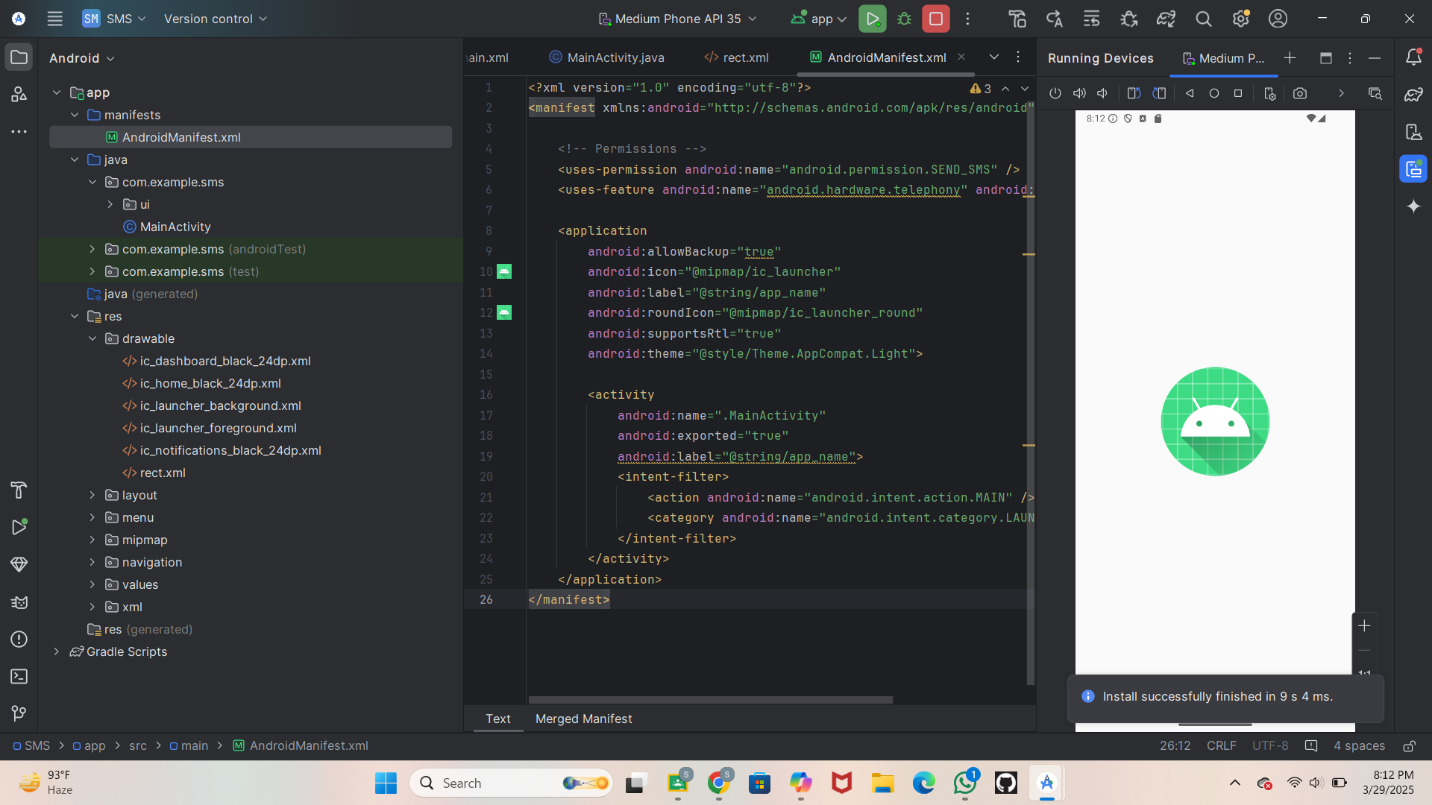
From all the above code we get

**ScreenShot**

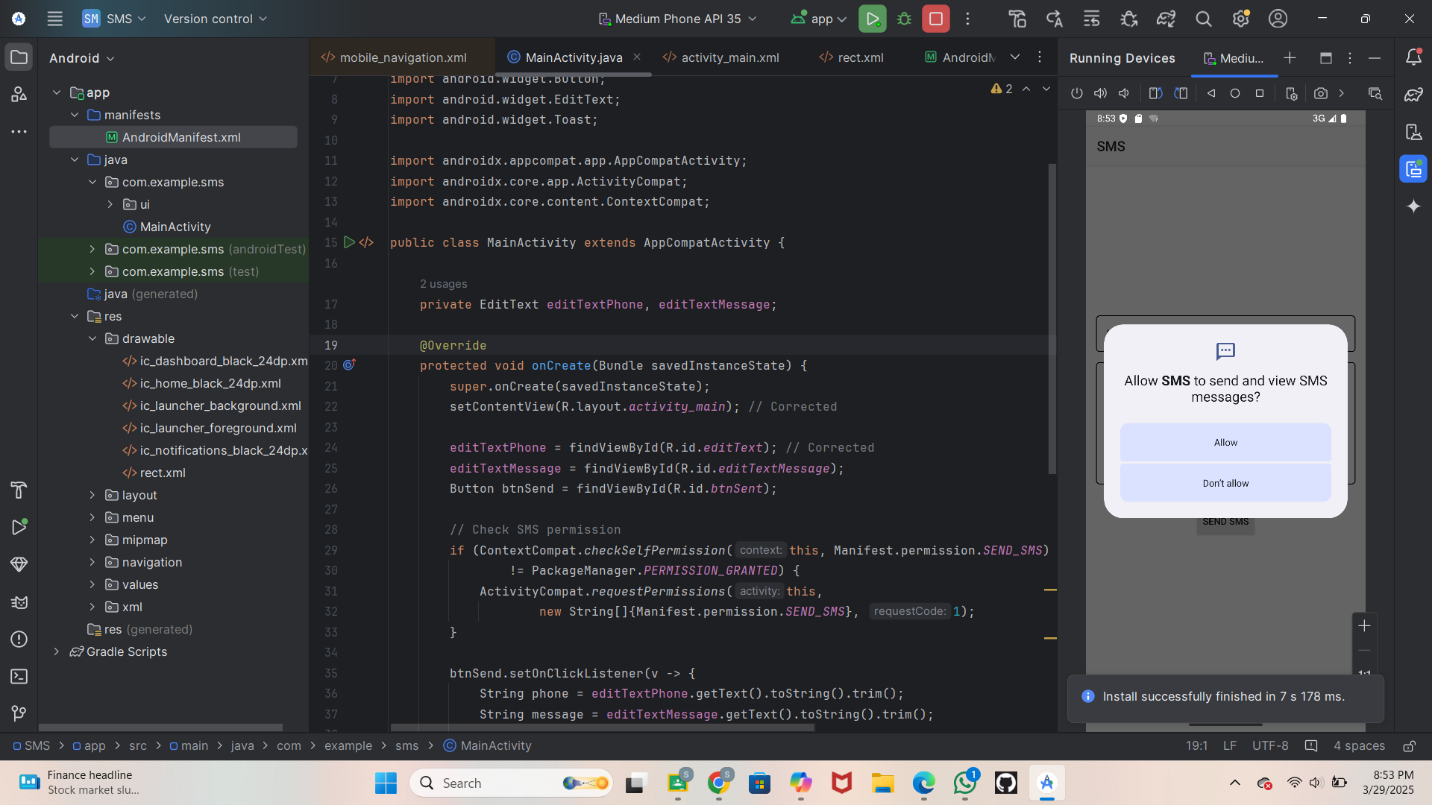


**Step 6:**

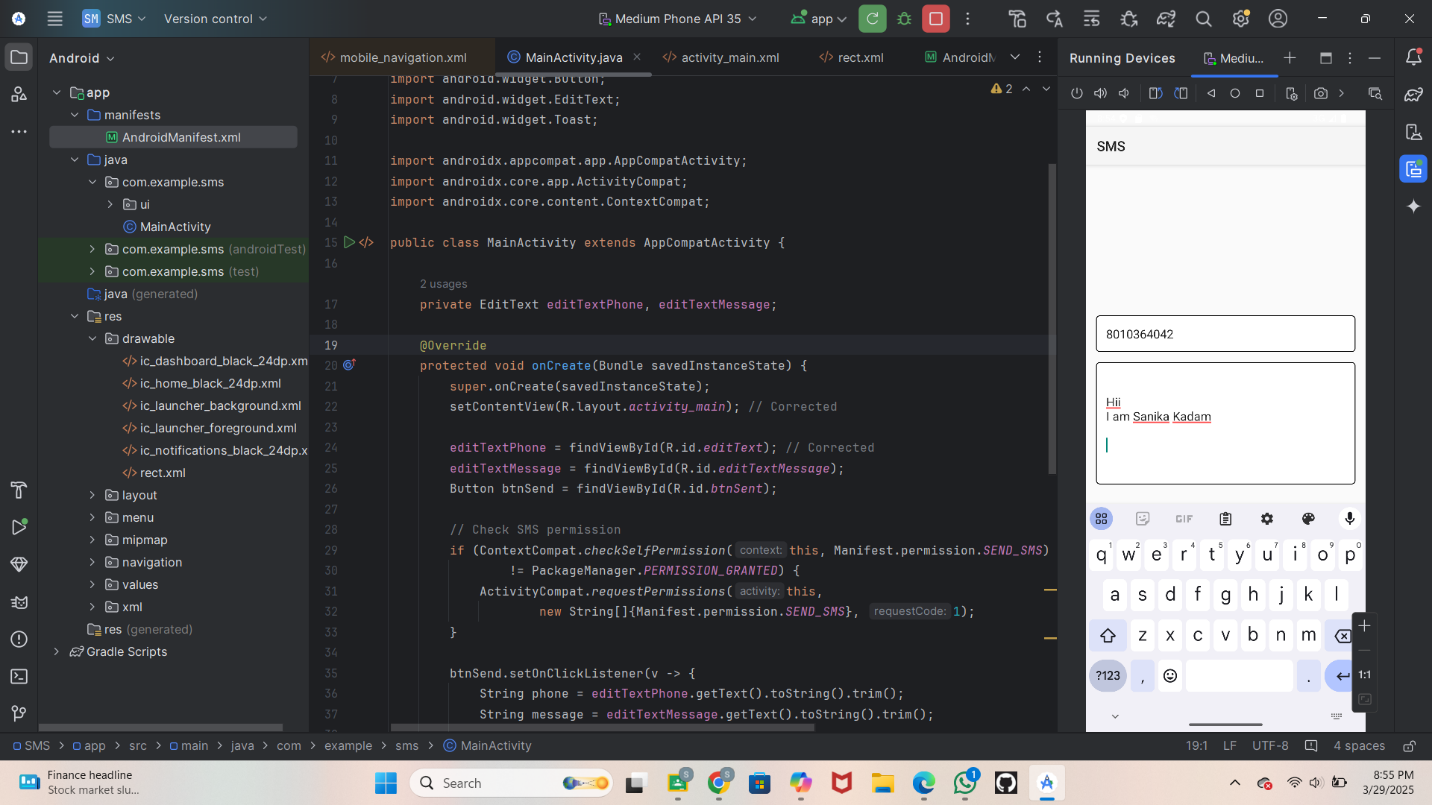
* Running the Application on Emulator
* Click on the **Run** button in Android Studio.
* **ScreenShot**



* **Step 7: Testing & Output**
* Test different functionalities of the app.
* the output results.
* Allow to send the SMS
* **Screenshot:**

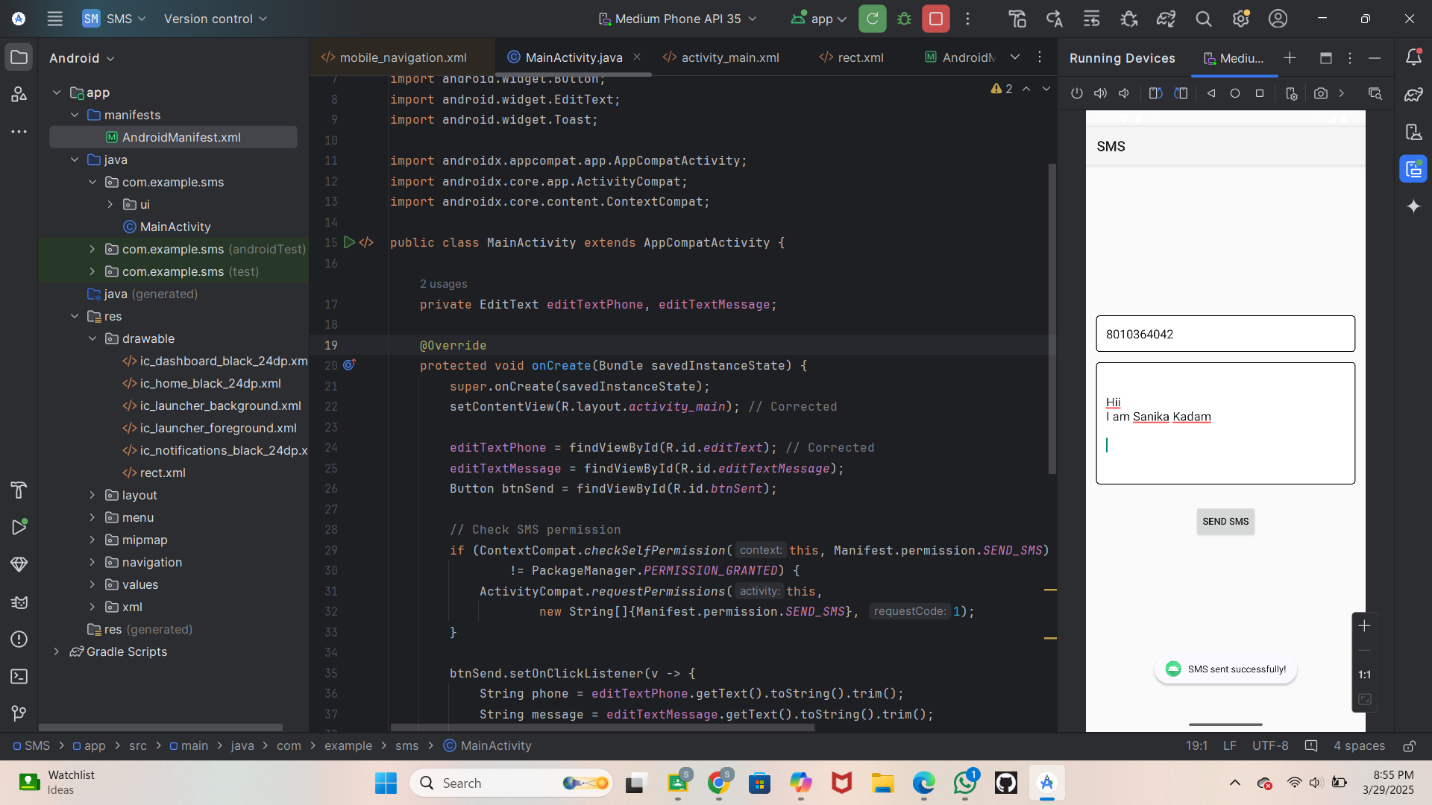


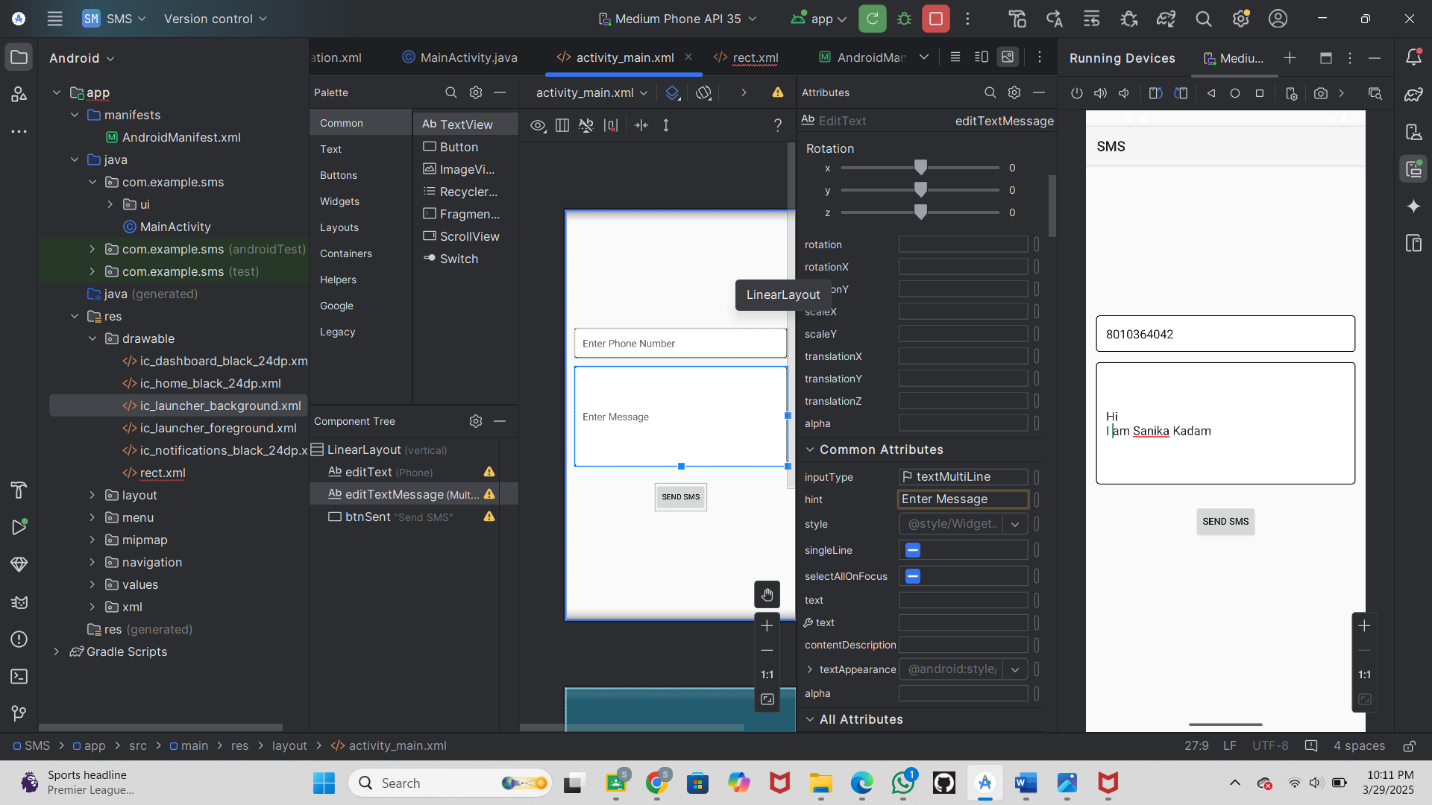
**Step 8: Write Phone no and what message we have send**



**Step 9: Click on send**

**ScreenShot**



****

**Step 10 :Conclusion**

An SMS sending app allows users to input a phone number and message, then send it easily. It highlights key Android development aspects like managing permissions securely, designing an intuitive user interface, and handling errors effectively.