Ex. No. : 07 **Date:**

Register No.: 221701029 **Name: Keerthana V**

Telephony services

Aim

Implement an application to get Telephony services.

Procedure:

1. Create a new Android project in Android Studio with an empty activity.

2. In AndroidManifest.xml, add the necessary permissions like <uses-permission android:name="android.permission.READ_PHONE_STATE" /> for accessing telephony services.

3. In MainActivity.java, import the required telephony classes such as TelephonyManager.

4. Initialize the TelephonyManager using getSystemService(Context.TELEPHONY_SERVICE).

5. Check if the app has permission to access telephony services using ContextCompat.checkSelfPermission().

6. If permission is granted, use the TelephonyManager instance to get device information (e.g., SIM card state, network operator, phone number).

7. Use methods like getDeviceId(), getLine1Number(), getNetworkOperatorName(), etc., to retrieve telephony data.

8. Display the retrieved information (e.g., phone number, network operator) in the app's UI using TextView elements.

9. Handle runtime permissions properly to request permissions when needed for telephony services.

10. Test the app on a real device to ensure it successfully retrieves and displays telephony data.

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.example.ex7">

  <!-- Permission for telephony access -->
  <uses-permission android:name="android.permission.READ_PHONE_STATE" />

  <application
    android:allowBackup="true"
    android:label="Telephony Info"
    android:icon="@mipmap/ic_launcher"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.Material3.DayNight.NoActionBar">

    <activity android:name=".MainActivity"
      android:exported="true">

      <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category android:name="android.intent.category.LAUNCHER"/>
      </intent-filter>
    </activity>

  </application>

</manifest>
```

Activity_main.xml

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

  <LinearLayout
    android:padding="16dp"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

    <Button
      android:id="@+id/btnFetch"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="Get Telephony Info" />

    <TextView
      android:id="@+id/txtInfo"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="Details will appear here..."
      android:layout_marginTop="20dp"
      android:textSize="16sp" />
  </LinearLayout>
</ScrollView>
```

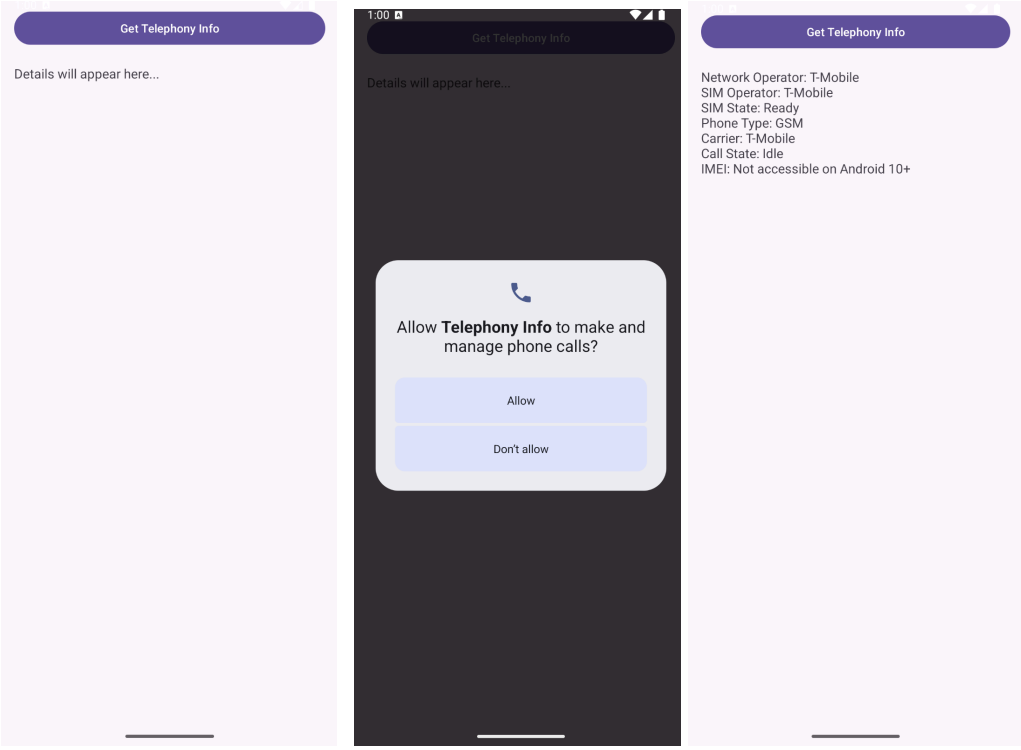

MainActivity.kt

```
package com.example.ex7
import android.Manifest
import android.content.pm.PackageManager
import android.os.Bundle
import android.telephony.TelephonyManager
import android.widget.Button
import android.widget.TextView
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.ActivityCompat
import androidx.core.content.ContextCompat

class MainActivity : AppCompatActivity() {

    private lateinit var txtInfo: TextView
    private lateinit var btnFetch: Button
    private val PERMISSION_REQUEST_CODE = 101

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        txtInfo = findViewById(R.id.txtInfo)
        btnFetch = findViewById(R.id.btnFetch)

        btnFetch.setOnClickListener {
                                            if     (ContextCompat.checkSelfPermission(this,
Manifest.permission.READ_PHONE_STATE)
                != PackageManager.PERMISSION_GRANTED) {
                ActivityCompat.requestPermissions(
                    this,
                    arrayOf(Manifest.permission.READ_PHONE_STATE),
                    PERMISSION_REQUEST_CODE
                )
            } else {
                fetchTelephonyInfo()
            }
        }
    }

    private fun fetchTelephonyInfo() {
        val tm = getSystemService(TELEPHONY_SERVICE) as TelephonyManager
```

```
        val info = StringBuilder()
        info.append("Network Operator: ${tm.networkOperatorName}\n")
        info.append("SIM Operator: ${tm.simOperatorName}\n")
        info.append("SIM State: ${getSimStateName(tm.simState)}\n")
        info.append("Phone Type: ${getPhoneTypeName(tm.phoneType)}\n")
        info.append("Carrier: ${tm.networkOperatorName}\n")
        info.append("Call State: ${getCallStateName(tm.callState)}\n")

        // IMEI requires special permission and works only for Android < 10
        try {
                                            if     (android.os.Build.VERSION.SDK_INT     <
android.os.Build.VERSION_CODES.Q) {
                val imei = tm.imei
                info.append("IMEI: $imei\n")
            } else {
                info.append("IMEI: Not accessible on Android 10+\n")
            }
        } catch (e: SecurityException) {
            info.append("IMEI: Permission not granted\n")
        }

        txtInfo.text = info.toString()
    }

    private fun getSimStateName(state: Int): String = when (state) {
        TelephonyManager.SIM_STATE_ABSENT -> "Absent"
        TelephonyManager.SIM_STATE_READY -> "Ready"
        TelephonyManager.SIM_STATE_PIN_REQUIRED -> "PIN Required"
        TelephonyManager.SIM_STATE_PUK_REQUIRED -> "PUK Required"
        else -> "Unknown"
    }

    private fun getPhoneTypeName(type: Int): String = when (type) {
        TelephonyManager.PHONE_TYPE_GSM -> "GSM"
        TelephonyManager.PHONE_TYPE_CDMA -> "CDMA"
        TelephonyManager.PHONE_TYPE_NONE -> "None"
        else -> "Unknown"
    }

    private fun getCallStateName(state: Int): String = when (state) {
        TelephonyManager.CALL_STATE_IDLE -> "Idle"
        TelephonyManager.CALL_STATE_RINGING -> "Ringing"
        TelephonyManager.CALL_STATE_OFFHOOK -> "Off Hook"
```


```
        else -> "Unknown"
    }

    override fun onRequestPermissionsResult(
        requestCode: Int, permissions: Array<out String>, grantResults: IntArray
    ) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults)
        if (requestCode == PERMISSION_REQUEST_CODE) {
                                if ((grantResults.isNotEmpty() && grantResults[0] ==
PackageManager.PERMISSION_GRANTED)) {
                fetchTelephonyInfo()
            } else {
                Toast.makeText(this, "Permission denied", Toast.LENGTH_SHORT).show()
            }
        }
    }
}
```

Output



Result:

The application successfully retrieves and displays telephony data such as the phone number and network operator using the TelephonyManager service in Android.