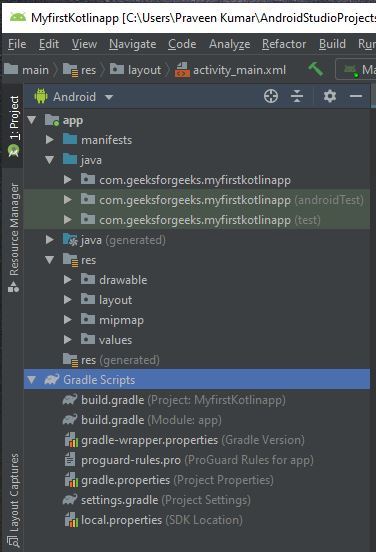
**Lab Exercise-9:** SMS Banking

Project Structure (Example)



**Open app > res > layout > activity\_main.xml. This file defines the layout for the user interface (UI). A UI in Android is defined in XML files**.

**activity\_main.xml**

**Root folder of the application**

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

<LinearLayout

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<EditText

android:id="@+id/phone\_number\_edittext"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Phone Number"/>

<EditText

android:id="@+id/message\_edittext"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Message"/>

<Button

android:id="@+id/send\_encrypted\_button"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Send Encrypted"/>

<Button

android:id="@+id/send\_plain\_text\_button"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Send Plain Text"/>

</LinearLayout>

**AndroidManifest.xml**

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

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

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

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

<application

android:allowBackup="true"

android:dataExtractionRules="@xml/data\_extraction\_rules"

android:fullBackupContent="@xml/backup\_rules"

android:icon="@mipmap/ic\_launcher"

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

android:supportsRtl="true"

android:theme="@style/Theme.SMSEncrptAndPt"

tools:targetApi="31">

<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>

**Application root folder 🡪 java 🡪**

**MainActivity.java**

// Importing necessary tools and libraries for the app.

package com.example.smsencrptandpt;

import android.Manifest;

import android.content.pm.PackageManager;

import android.os.Bundle;

import android.telephony.SmsManager;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Switch;

import android.widget.Toast;

import androidx.annotation.NonNull;

import androidx.appcompat.app.AppCompatActivity;

import java.nio.charset.StandardCharsets;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

import java.util.ArrayList;

// Main part of the app begins.

public class MainActivity extends AppCompatActivity {

// Constants and variables for managing permissions and UI elements.

private static final int PERMISSION\_REQUEST\_SEND\_SMS = 1;

private EditText mMessageEditText;

private EditText mPhoneNumberEditText;

//private Switch mEncryptionSwitch;

private Button mSendEncryptedButton;

private Button mSendPlainTextButton;

private boolean mIsEncrypted = false;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

// Initializing UI elements.

mMessageEditText = findViewById(R.id.message\_edittext);

mPhoneNumberEditText = findViewById(R.id.phone\_number\_edittext);

mSendEncryptedButton = findViewById(R.id.send\_encrypted\_button);

mSendPlainTextButton = findViewById(R.id.send\_plain\_text\_button);

// Setting click listeners for send buttons.

mSendEncryptedButton.setOnClickListener(view -> {

mIsEncrypted = true;

sendSMS();

});

mSendPlainTextButton.setOnClickListener(view -> {

mIsEncrypted = false;

sendSMS();

});

}

// Function to send SMS.

private void sendSMS() {

String message = mMessageEditText.getText().toString();

String phoneNumber = mPhoneNumberEditText.getText().toString();

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

if (checkSelfPermission(Manifest.permission.SEND\_SMS) == PackageManager.PERMISSION\_GRANTED) {

SmsManager smsManager = SmsManager.getDefault();

String encryptedMessage = mIsEncrypted ? encryptMessage(message) : message;

ArrayList<String> messageParts = smsManager.divideMessage(encryptedMessage);

smsManager.sendMultipartTextMessage(phoneNumber, null, messageParts, null, null);

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

} else {

requestPermissions(new String[]{Manifest.permission.SEND\_SMS}, PERMISSION\_REQUEST\_SEND\_SMS);

}

}

}

// Function to encrypt the message.

private String encryptMessage(String message) {

try {

MessageDigest md = MessageDigest.getInstance("SHA-256");

byte[] hash = md.digest(message.getBytes(StandardCharsets.UTF\_8));

return bytesToHex(hash);

} catch (NoSuchAlgorithmException e) {

e.printStackTrace();

}

return null;

}

// Function to convert bytes to hexadecimal representation.

private String bytesToHex(byte[] hash) {

StringBuilder hexString = new StringBuilder();

for (byte b : hash) {

String hex = Integer.toHexString(0xff & b);

if (hex.length() == 1) hexString.append('0');

hexString.append(hex);

}

return hexString.toString();

}

// Handling permission results after user interaction.

@Override

public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {

super.onRequestPermissionsResult(requestCode, permissions, grantResults);

if (requestCode == PERMISSION\_REQUEST\_SEND\_SMS) {

if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION\_GRANTED) {

sendSMS();

}

}

}

}