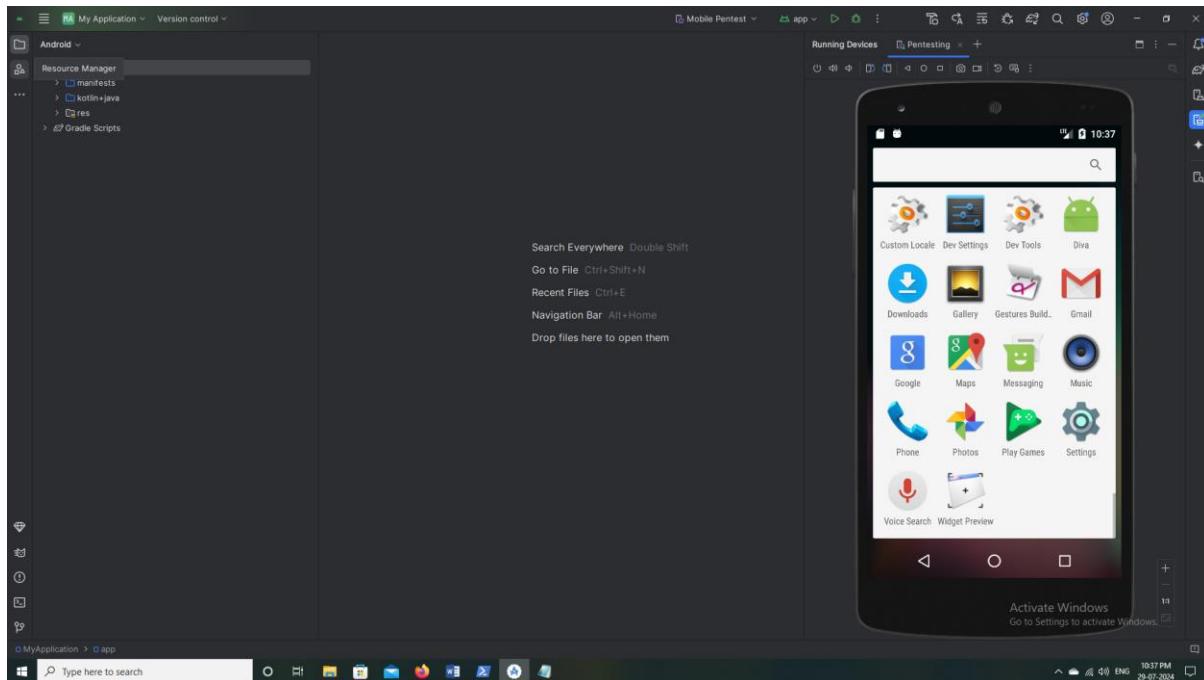


## Diva Apk Mobile Pentest - Static Analysis

### Introduction

The interface of Android Studio and the Emulator. Install DIVA application from playstore.



To Prints all packages - `adb shell pm list packages`

```
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> adb shell pm list packages
package:com.android.smoketest
package:com.example.android.livecubes
package:com.android.providers.telephony
package:com.google.android.providers.settingssearchbox
package:com.android.providers.calendar
package:com.android.providers.media
package:com.android.providers.media
package:com.android.documentsui
package:com.android.gallery
package:com.android.externalstorage
package:com.android.htmlviewer
package:com.android.usbdevice
package:com.android.usbdevice
```

Print the path of the.apk of the given installed package name - `adb shell pm path`

To copy a file or directory and its sub-directories from the Android device - `adb pull`

```
package:jakhar.aseem.diva
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> adb shell pm path jakhar.aseem.diva
package:/data/app/jakhar.aseem.diva-1/base.apk
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> adb pull /data/app/jakhar.aseem.diva-1/base.apk
/data/app/jakhar.aseem.diva-1/base.apk: 1 file pulled, 0 skipped. 18.6 MB/s (1502294 bytes in 0.077s)
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> ls

Directory: D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting

Mode                LastWriteTime        Length Name
----                -----        ---- 
-a---- 29-07-2024 10:35 PM      1502294 base.apk

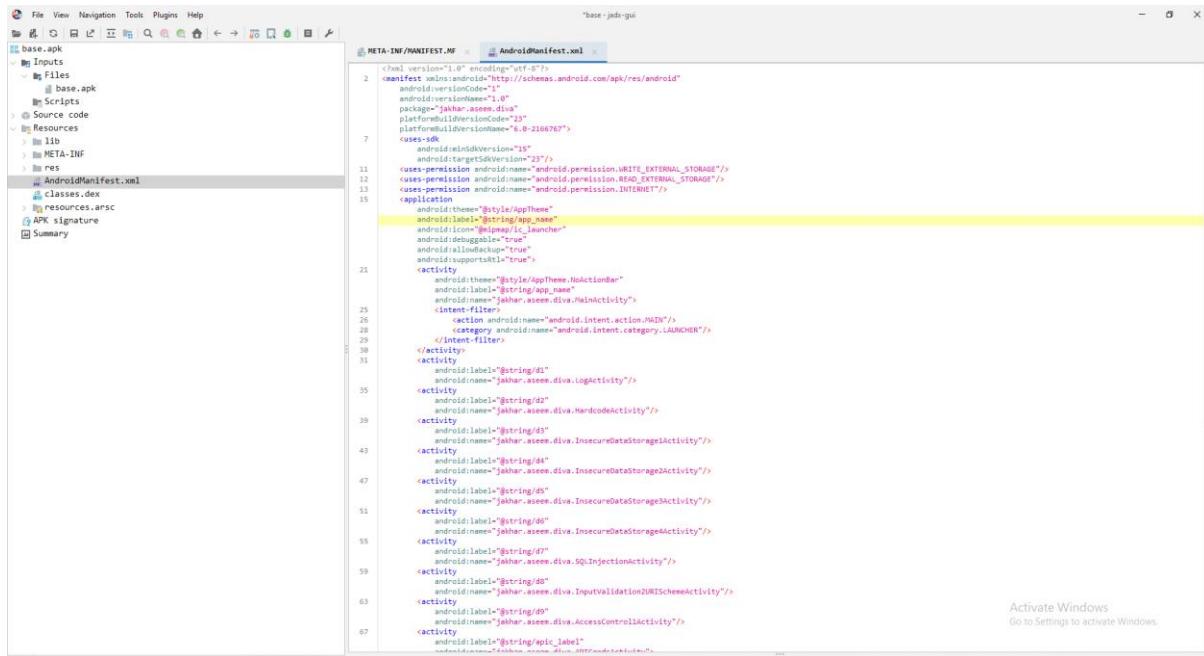
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting>
```

**Base APK:** It is a file format that Android uses to distribute and install apps. It is the original, unmodified version of an app that contains all the elements needed for installation. You can use Base APK to customize your apps, backup your data, or access apps that are not available on Google Play.

Now open the base.apk with jadx.

**JADeX:** Users can analyse and comprehend the source code structure of an Android application.

- decompile Dalvik bytecode or DEX code to Java source code from APK, dex, aar, aab and zip files.
- decode `AndroidManifest.xml` and other resources from `resources.arsc`



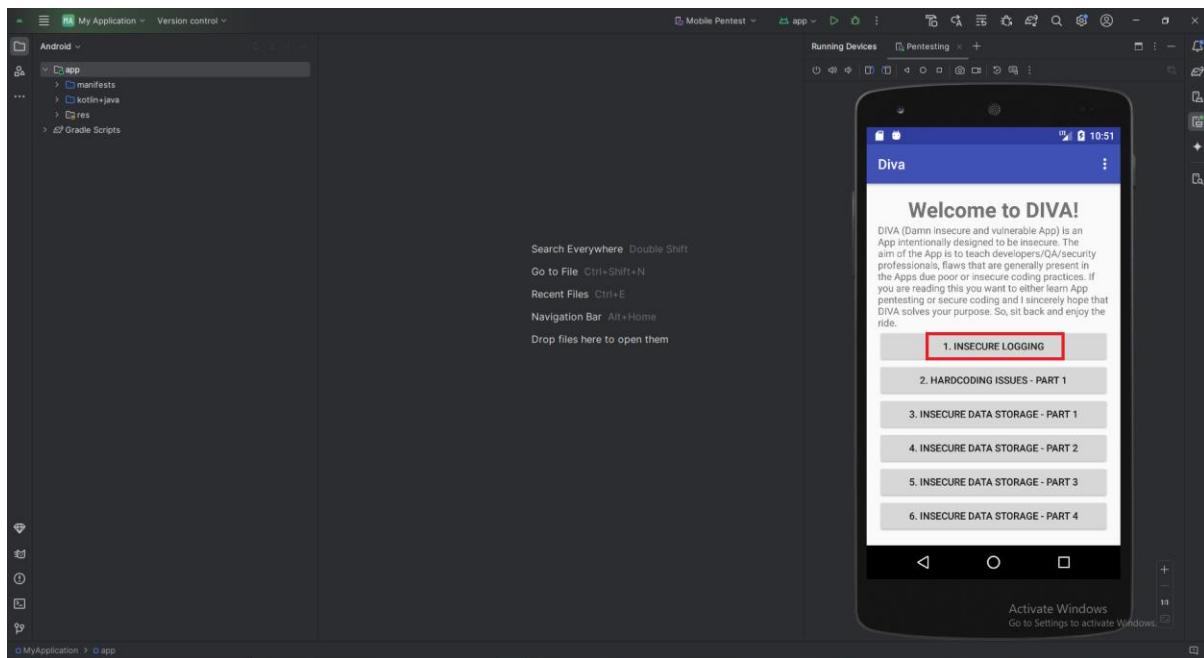
The screenshot shows the jadx-gui application window. On the left, there's a file tree with 'base.apk' selected. The main area displays the decompiled code of the `AndroidManifest.xml` file. The code includes manifest tags like `<manifest>`, `<application>`, and `<activity>`, along with various attributes such as `android:label`, `android:icon`, and `android:theme`. Permissions like `uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"` and `uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"` are also visible. The jadx-gui interface has a top navigation bar with File, View, Navigation, Tools, Plugins, Help, and a toolbar below it.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    android:minSdkVersion="1"
    android:versionCode="1"
    android:versionName="1.0"
    package="com.jahar.aseen.diva"
    platformBuildVersionCode="23"
    platformBuildVersionName="8.0-2166767">
    <uses-sdk
        android:minSdkVersion="18"
        android:targetSdkVersion="23"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.INTERNET"/>
<application
    android:allowBackup="true"
    android:icon="@string/app_name"
    android:label="@string/app_name"
    android:debuggable="true"
    android:theme="@style/AppTheme"
    android:supportRtl="true">
        <activity
            android:label="@string/d1"
            android:name="jahar.aseen.diva.LoginActivity"/>
            <intent-filter
                android:label="jahar.aseen.diva.LoginActivity">
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            
        </activity>
        <activity
            android:label="@string/d2"
            android:name="jahar.aseen.diva.HardcodeActivity"/>
            <activity
                android:label="@string/d3"
                android:name="jahar.aseen.diva.InsecureDataStorageActivity"/>
            <activity
                android:label="@string/d4"
                android:name="jahar.aseen.diva.InsecureDataStorage2Activity"/>
            <activity
                android:label="@string/d5"
                android:name="jahar.aseen.diva.InsecureDataStorage3Activity"/>
            <activity
                android:label="@string/d6"
                android:name="jahar.aseen.diva.InsecureDataStorage4Activity"/>
            <activity
                android:label="@string/d7"
                android:name="jahar.aseen.diva.SQLInjectionActivity"/>
            <activity
                android:label="@string/d8"
                android:name="jahar.aseen.diva.InputValidation2URISchemeActivity"/>
            <activity
                android:label="@string/d9"
                android:name="jahar.aseen.diva.AccessControlActivity"/>
            <activity
                android:label="@string/app_label"
                android:name="jahar.aseen.diva.MainActivity"/>

```

# 1. Insecure Logging

Screenshot-1: The activity **LogActivity** and its source code.



The screenshot shows the JD-GUI decompiler interface. The left sidebar lists the package structure: 'Base.apk' contains 'Inputs', 'base.apk', 'Source code' (which is highlighted with a red box), 'jakkhar.aseem.diva' (which is also highlighted with a red box), and 'LogActivity'. The right pane shows the Java code for 'LogActivity'. The code includes imports for android.os.Bundle, android.support.v7.app.AppCompatActivity, android.util.Log, android.view.View, android.widget.EditText, and android.widget.Toast. It defines a class 'LogActivity' that extends AppCompatActivity. The onCreate method saves the instance state and sets the content view. The checkOut method retrieves an EditText from the view and processes its text. The processCC method handles a runtime exception. The code ends with a closing brace for the LogActivity class. The JD-GUI interface has tabs for 'Code', 'Small', 'Simple', 'Fallback', and 'Split view' at the bottom.

```
package jakkhar.aseem.diva;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
/* Loaded from: classes.jar */
public class LogActivity extends AppCompatActivity {
    /* SAFX INFO: Access modifiers changed from: protected */
    @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.FragmentActivity, android.support.v4.app.BaseFragmentActivityDonut, android.app.Activity
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_log);
    }
    public void checkout(View view) {
        EditText ccText = (EditText) findViewById(R.id.ccText);
        try {
            processCC(ccText.getText().toString());
        } catch (Runtimeexception e) {
            Log.e("diva-log", "Error while processing transaction with credit card: " + ccText.getText().toString());
            Toast.makeText(this, "An error occurred. Please try again later!", 0).show();
        }
    }
    private void processCC(String ccstr) {
        Runtimeexception e = new Runtimeexception();
        throw e;
    }
}
```

Screenshot-2: List processes - `adb shell ps`

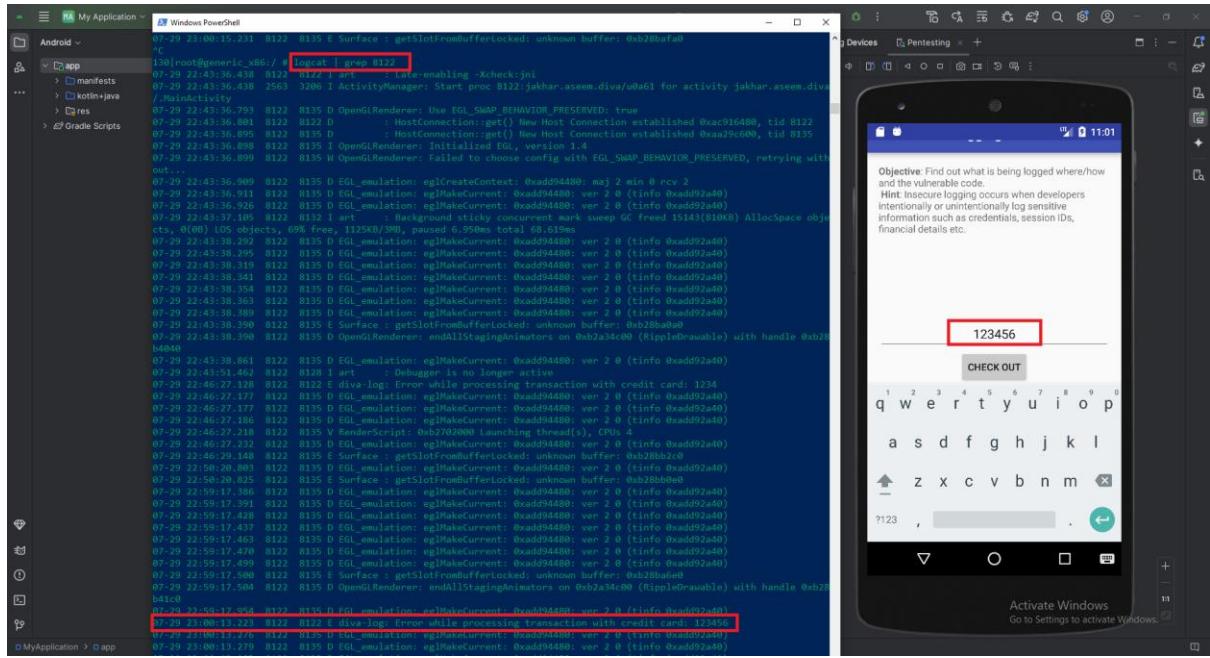
```

PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> adb shell ps
USER   PID  PPID  VSIZE  RSS  WCHAN          PC  NAME
root    1     0    2964  1064  SyS_epoll_081249c5 S /init
root    2     0    0      0      kthreadd 00000000 S kthreadd
root    3     2    0    0      0      smpboot_th 00000000 S ksoftirqd/0
root    5     2    0    0      0      worker_thr 00000000 S kworker/0:0H
root    6     2    0    0      0      worker_thr 00000000 S kworker/u8:0
root    7     2    0    0      0      smpboot_th 00000000 S migration/0
root    8     2    0    0      0      rcu_gp_kth 00000000 S rCU_prempt

root    7906  2     0    0      0      worker_thr 00000000 S kworker/2:1
root    8034  1     21040  548  poll_sched 0808e495 S /sbin/adbd
root    8056  8034  3868  1204  hrtimer_na b7617e01 S /data/local/tmp/.studio/process-tracker
u0_a61  8122  2299  1561180 46004  SyS_epoll_b72e1685 S jakhar.aseem.diva
root    8157  2     0    0      0      worker_thr 00000000 S kworker/1:0
root    8164  8034  3980  1108          0 b754b7d6 R ps

```

**Screenshot-3:** The Logcat window in Android Studio helps you debug your app by displaying logs from your device in real time.



## 2. Hardcoding Issue \_Part-1

**Screenshot-1:** Sensitive information are hardcoded in source code of the .apk file.

```

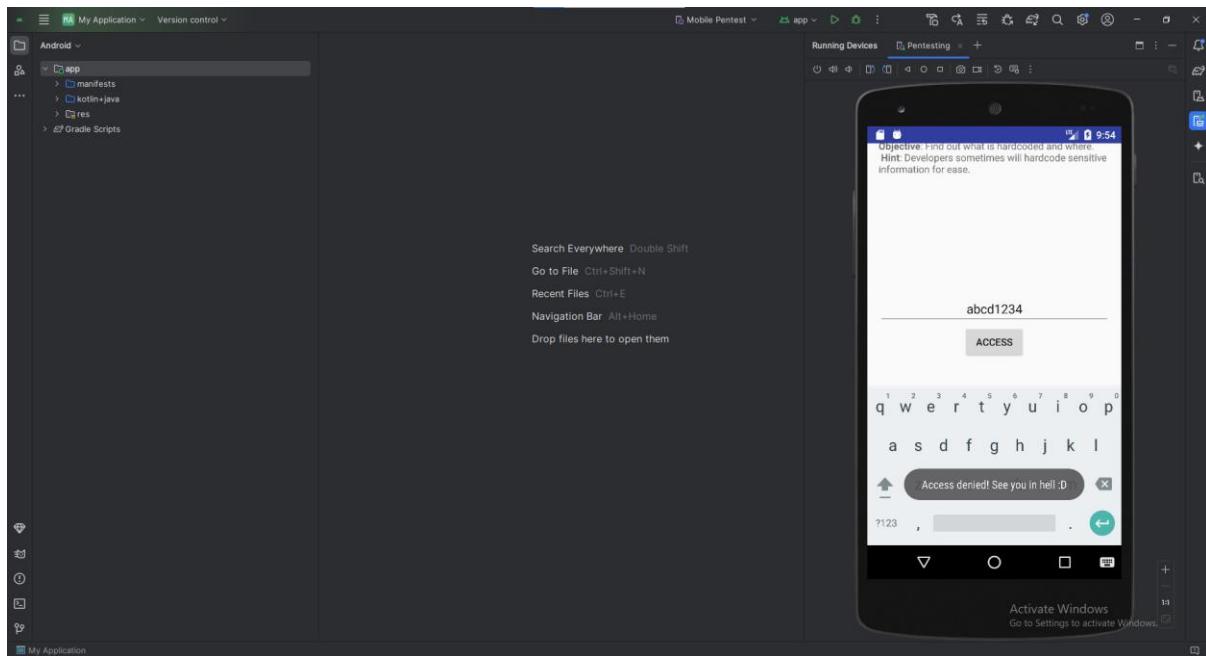
    package jahar.aseem.diva;
    import android.os.Bundle;
    import android.support.v7.app.AppCompatActivity;
    import android.view.View;
    import android.widget.EditText;
    import android.widget.Toast;

    /* Loaded from: classes.dex */
    public class HandcodeActivity extends AppCompatActivity {
        /* JADIC INFO: Access modifiers changed from: protected */
        /* JADIC INFO: protected void onCreate(Bundle savedInstanceState) { */
        public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_handcode);
        }

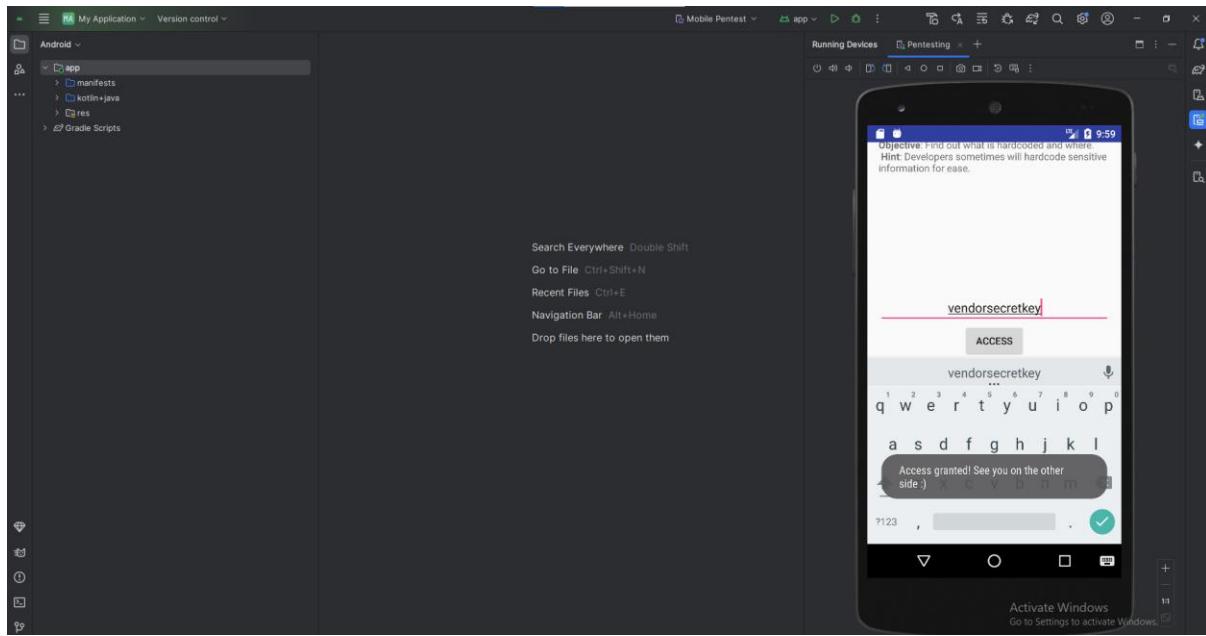
        public void access(View v) {
            EditText hkey = (EditText) findViewById(R.id.hkey);
            if (hkey.getText().toString().equals("vendorsecretkey")) {
                Toast.makeText(this, "Access granted! See you on the other side :)", 0).show();
            } else {
                Toast.makeText(this, "Access denied! See you in hell :D", 0).show();
            }
        }
    }

```

**Screenshot-2:** Type wrong vendor code and get the error “Access denied! See you in hell :D”

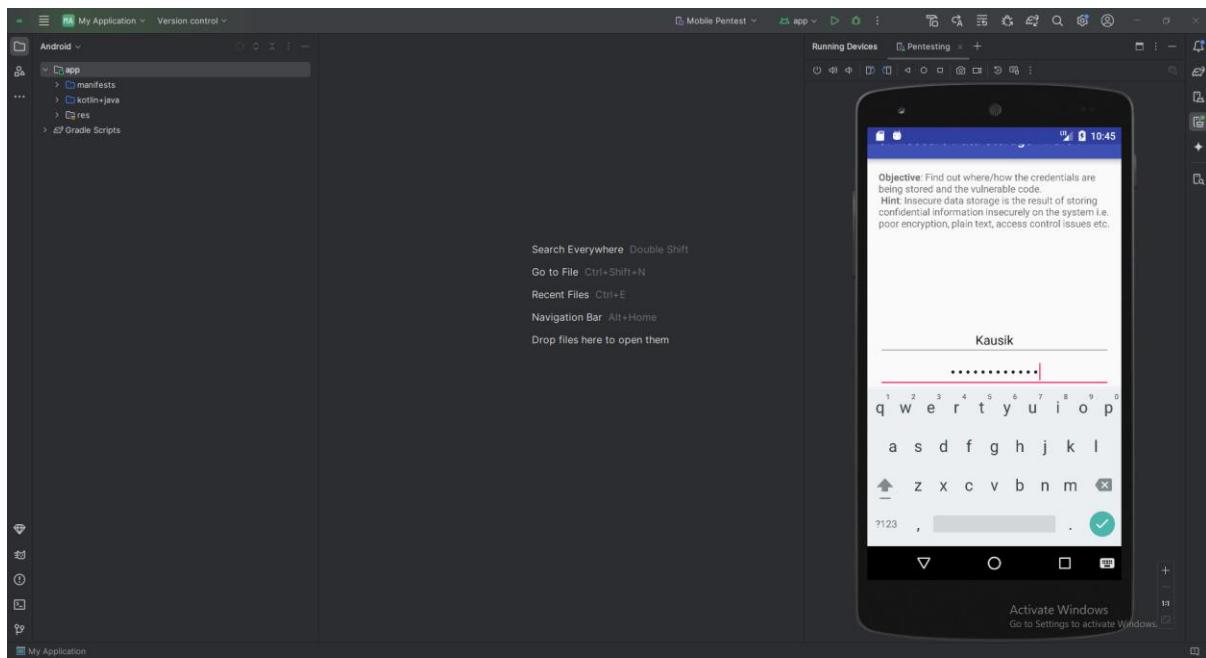


**Screenshot-3:** Type correct vendor code which is hardcoded in source code and get the error “Access granted! See you on the other side :)”



### 3. Insecure Data Storage Part-1

**Screenshot-1:** Enter any random User ID and Password.



**Screenshot-2:** Respective activity & its source code. It could observe that credentials are storing inside of Shared Preferences file.

```

    package jakhar.aseem.diva;
    import android.content.SharedPreferences;
    import android.os.Bundle;
    import android.preference.PreferenceManager;
    import android.support.v7.app.AppCompatActivity;
    import android.view.View;
    import android.widget.EditText;
    import android.widget.Toast;

    /* Loaded from: classes.dex */
    public class InsecureDataStorageActivity extends AppCompatActivity {
        @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.FragmentActivity, android.support.v4.app.BaseFragmentActivityOnDonut, android.app.Activity
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_insecure_data_storage);
        }

        public void saveCredentials(View view) {
            SharedPreferences spref = PreferenceManager.getDefaultSharedPreferences(this);
            SharedPreferences.Editor spedit = spref.edit();
            EditText uid = (EditText) findViewById(R.id.uid);
            EditText pwd = (EditText) findViewById(R.id.pwd);
            spedit.putString("user", uid.getText().toString());
            spedit.putString("password", pwd.getText().toString());
            spedit.commit();
            Toast.makeText(this, "3rd party credentials saved successfully!", 0).show();
        }
    }

```

**Screenshot-3:**

Go to root using `adb shell` and go to the path `/data/data/jakhar.aseem.diva/shared_prefs #`  
Now, we can find the file `jakhar.aseem.diva_preferences.xml` and also can see the entered User ID and Password as the content of the file.

```

root@generic_x86:/data # exit
PS D:\Security All\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting [adb shell]
root@generic_x86:/ # ls
acct
cache
charger
config
d ←
data
default.prop
dev
etc
file_contexts
fstab.goldfish
fstab.ranchu
fstab.ranchu-encrypt
init
init.environ.rc
init.goldfish.rc
init.ranchu-common.rc
init.ranchu-encrypt.rc
init.ranchu.rc
init.rc
init.trace.rc
init.usb.rc
init.zygote32.rc
mnt
pem
proc
property_contexts
root
sbin
sdcard
seapp_contexts
selinux_version
sepolicy
service_contexts
storage
sys
system
ueventd.goldfish.rc
ueventd.ranchu.rc
ueventd.rc
vendor
root@generic_x86:/ # cd data
root@generic_x86:/data # ls

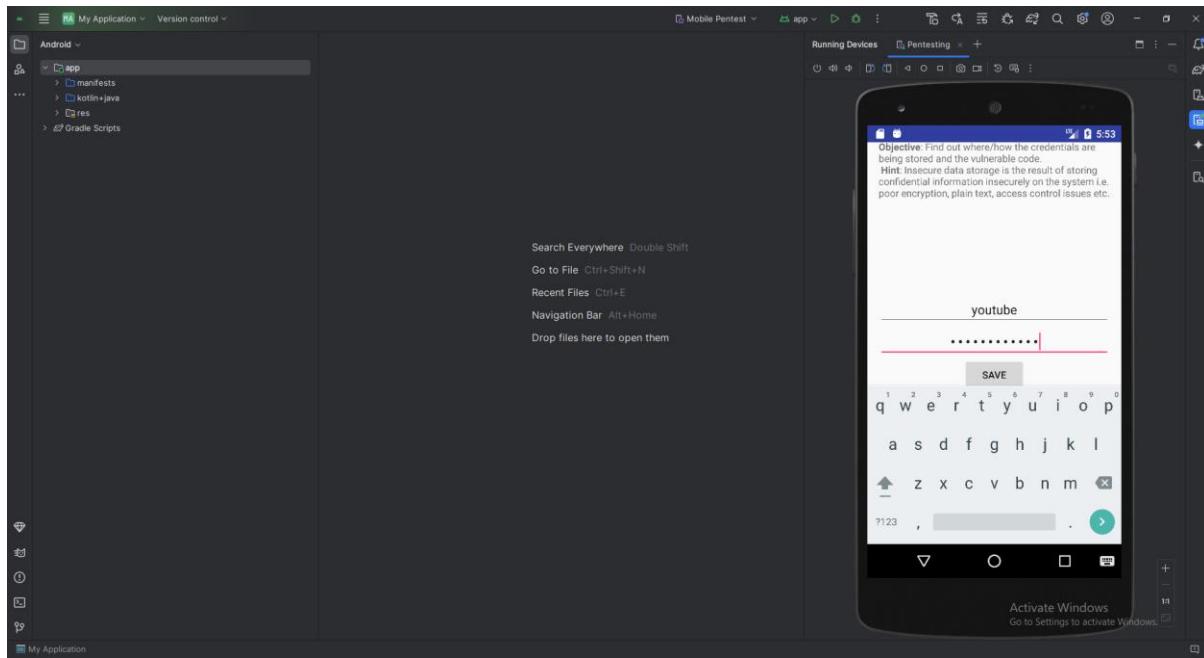
```

```
root@generic_x86:/data # ls
adb
amr
app
app-asec
app-lib
app-private
backup
bootchart
bugreports
dalvik-cache
data ←
drm
local
lost+found
media
mediadrm
misc
nativebenchmark
nativetest
property
resource-cache
security
ss
system
tombstones
user
root@generic_x86:/data # cd data
root@generic_x86:/data/data # ls
```

```
root@generic_x86:/data/data # cd jakhar.aseem.diva/
root@generic_x86:/data/data/jakhar.aseem.diva # ls
cache
code_cache
databases
lib
root@generic_x86:/data/data/jakhar.aseem.diva # ls -la
drwxrwx--x u0_a61  u0_a61          2024-07-11 23:14 cache
drwxrwx--x u0_a61  u0_a61          2024-07-11 23:14 code_cache
drwxrwxrwx u0_a61  u0_a61          2024-07-11 23:14 databases
lrwxrwxrwx root   root           2024-07-11 23:14 lib -> /data/app/jakhar.aseem.diva-1/lib/x86
root@generic_x86:/data/data/jakhar.aseem.diva # ls -a
cache
code_cache
databases
lib
root@generic_x86:/data/data/jakhar.aseem.diva # ls -a
cache
code_cache
databases
lib
shared_prefs → cd shared prefs
d shared_prefs/
root@generic_x86:/data/data/jakhar.aseem.diva/shared_prefs # ls
jakhar.aseem.diva_preferences.xml → cat jakhar.aseem.diva_preferences.xml
at jakhar.aseem.diva_preferences.xml <
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
    <string name="user">Kausik</string>
    <string name="password">welcome@1234</string>
</map>
root@generic_x86:/data/data/jakhar.aseem.diva/shared_prefs #
```

## 4. Insecure Data Storage\_Part-2

**Screenshot-1:** Enter any random User ID and Password.



**Screenshot-2:** Respective activity & its source code. We can find the name of database and table name.

```
*base.apk [D:\Security\All\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting] - jd-gui
File View Navigation Tools Plugins Help
META-INF/MANIFEST.MF AndroidManifest.xml LogActivity InsecureDataStorage2Activity
base.apk
Inputs
Files jakhar.apk
Scripts
Source code
android.support
jakhar.assem.diva
AccessControlActivity
AccessControl2Activity
AccessControl3Activity
AccessControl3MatesActivity
ICRedi2Activity
pocActivity
BuildConfig
DivAni
HandcodeActivity
HandcodeSchemeActivity
InputValidationURLSchemeActivity
InputValidation3Activity
InsecureDataStorage1Activity
InsecureDataStorage2Activity
InsecureDataStorage3Activity
InsecureDataStorage4Activity
LogActivity
MainActivity
NotesProvider
R
SQLInjectionActivity
Resources
META-INF
res
AndroidManifest.xml
resources.arsc
APK signature
Summary
package jakhar.assem.diva;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
/* Loaded from: classes.dex */
public class InsecureDataStorage2Activity extends AppCompatActivity {
    private SQLiteDatabase dB;
    /* JD-GO INFO: Access modifiers changed from: protected */
    /* JD-GO INFO: to public */
    /* JD-GO INFO: in class: InsecureDataStorage2Activity, android.support.v4.app.FragmentActivity, android.support.v4.app.FragmentManager, android.support.v4.app.FragmentActivityDonut, android.app.Activity */
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        try {
            this.dB = openOrCreateDatabase("ids2", 0, null);
            this.dB.execSQL("CREATE TABLE IF NOT EXISTS myuser(user varchar(50), password varchar(50));");
        } catch (Exception e) {
            Log.d("Div", "Error occurred while creating database: " + e.getMessage());
        }
        setContentView(R.layout.activity_insecure_data_storage2);
    }
    public void saveCredentials(View view) {
        EditText usr = (EditText) findViewById(R.id.ids2Usr);
        EditText pwd = (EditText) findViewById(R.id.ids2Pwd);
        try {
            this.dB.execSQL("INSERT INTO myuser VALUES ('" + usr.getText().toString() + "','" + pwd.getText().toString() + "')");
            this.dB.close();
        } catch (Exception e) {
            Log.d("Div", "Error occurred while inserting into database: " + e.getMessage());
        }
        Toast.makeText(this, "3rd party credentials saved successfully!", 0).show();
    }
}
Database name = ids2
Table name = myuser
Activate Windows
Go to Settings to activate Windows
```

### Screenshot-3:

Go to root using adb shell and go to the path **/data/data/jakhar.aseem.diva/databases #**  
Now, we can see the database file & pull the file **ids2**.

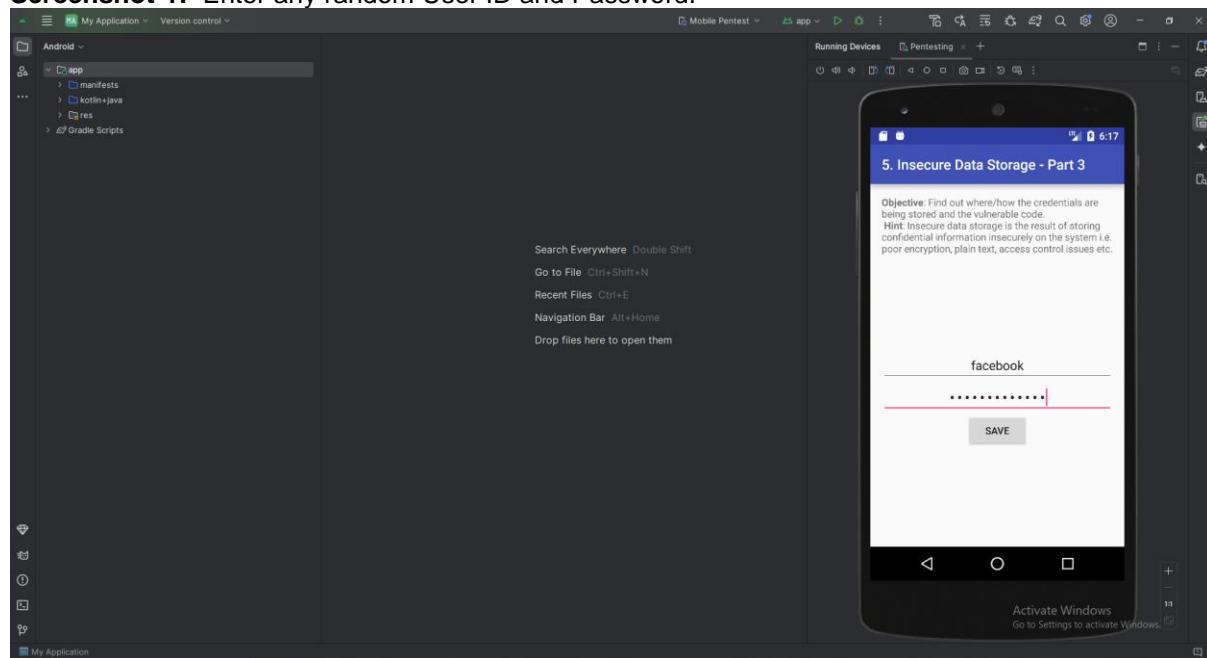
```
root@generic_x86:/data/data/jakhar.aseem.diva/databases# ls
divanotes.db
divanotes.db-journal
ids2 ← .
ids2-journal
root@generic_x86:/data/data/jakhar.aseem.diva/databases# ls -la
-rw-rw---- u0_a61 u0_a61 20480 2024-07-11 23:14 divanotes.db
-rw-rw---- u0_a61 u0_a61 8720 2024-07-11 23:14 divanotes.db-journal
-rw-rw---- u0_a61 u0_a61 16384 2024-08-03 17:33 ids2
root@generic_x86:/data/data/jakhar.aseem.diva/databases# exit
PS C:\Users\Kausik> cd ..
PS D:\> cd ..\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting\.
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> adb pull /data/data/jakhar.aseem.diva/databases/ids2
/data/data/jakhar.aseem.diva/databases/ids2:  File pulled, 0 skipped. 0.9 MB/s (16384 bytes in 0.017s)
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> Database file 'ids2' has pulled.
```

**Screenshot-4:** Open the database file **ids2** using online tool sqlite3 viewer and select the respective table **myuser**. You can find entered User ID and Password.

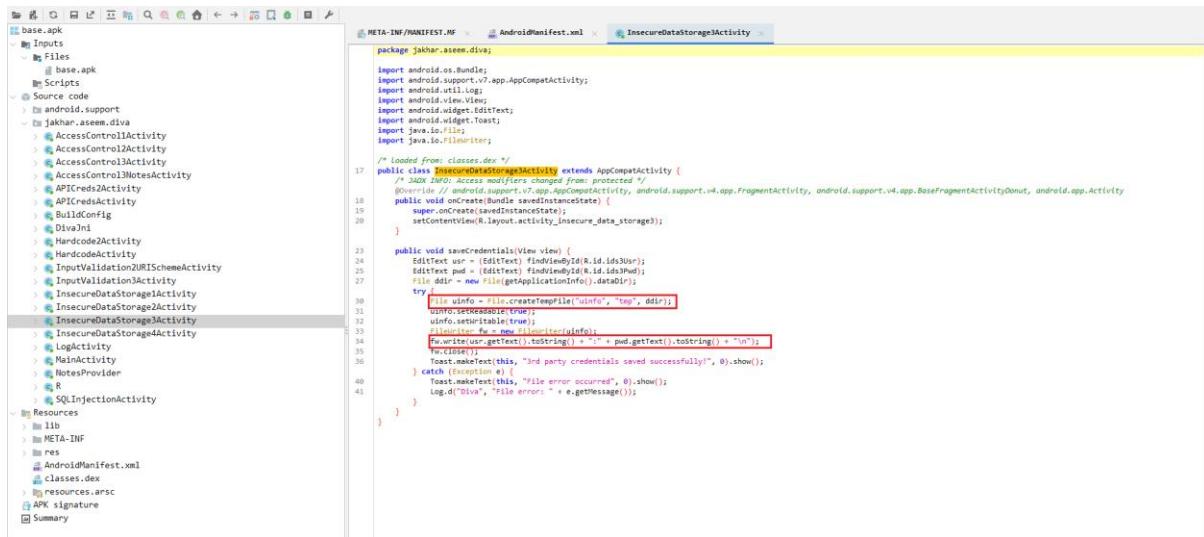
The screenshot shows the SQLite Viewer interface. At the top, there's a URL bar with the address https://inloop.github.io/sqlite-viewer/. Below it, a search bar contains 'myuser'. A query box contains 'SELECT \* FROM `myuser` LIMIT 0,30'. The results table shows one row with columns 'user' and 'password'. The 'user' column value is 'youtube' and the 'password' column value is 'youtube\_pass'. The interface includes a navigation bar at the bottom with icons for back, forward, and search.

## 5. Insecure Data Storage\_Part-3

**Screenshot-1:** Enter any random User ID and Password.



**Screenshot-2:** Respective activity & its source code. We can see that entered credentials are storing in a file name as **uinfo tmp** in **username:password** format.



```
package jakhar.aseem.diva;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
import android.widget.ToggleButton;
import java.io.FileWriter;
import java.io.IOException;
```

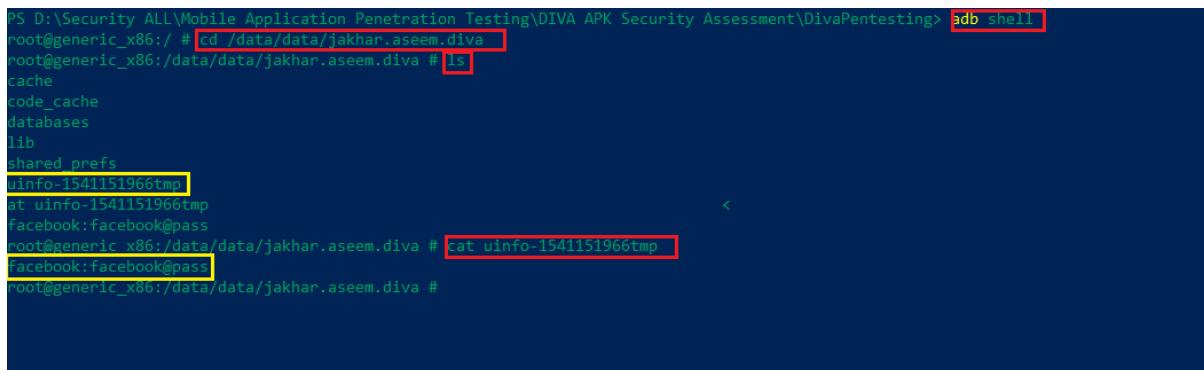
```
/* loaded from: classes.dex */
public class InsecureDataStorage3Activity extends AppCompatActivity {
    /* 3DXDZ IWDI: Access modifiers changed from: protected */
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_insecure_data_storage3);
    }

    public void saveCredentials(View view) {
        EditText usr = (EditText) findViewById(R.id.idsUsr);
        EditText pwd = (EditText) findViewById(R.id.idsPwd);
        File ddir = new File(getApplicationContext().getDatabasePath());
        try {
            uinfo = File.createTempFile("uinfo", "tmp", ddir);
            uinfo.setWritable(true);
            uinfo.setReadable(true);
            FileWriter fw = new FileWriter(uinfo);
            fw.write(usr.getText().toString() + ":" + pwd.getText() + "\n");
            fw.close();
            Toast.makeText(this, "3rd party credentials saved successfully!", 0).show();
        } catch (IOException e) {
            Toast.makeText(this, "File error occurred", 0).show();
            Log.d("Divs", "File error: " + e.getMessage());
        }
    }
}
```

**Screenshot-3:**

Go to root using **adb shell** and go to the path **/data/data/jakhar.aseem.diva/**

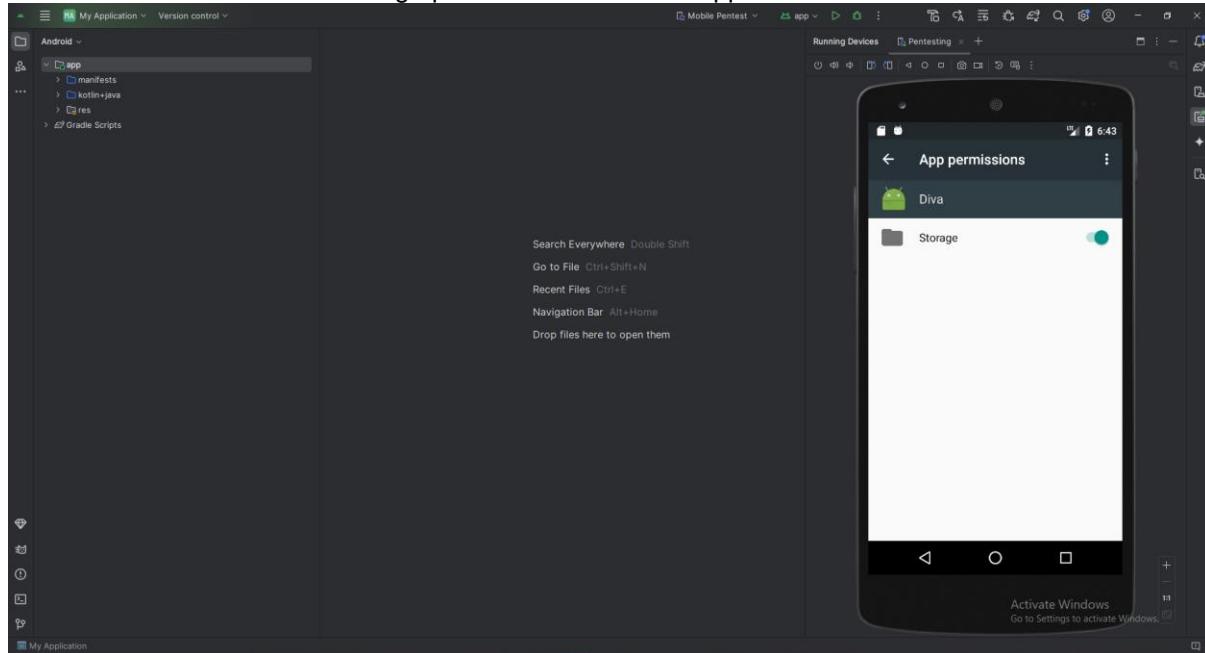
Now, we can find the file **uinfo-1541151966tmp** and can see the credentials as content of the file.



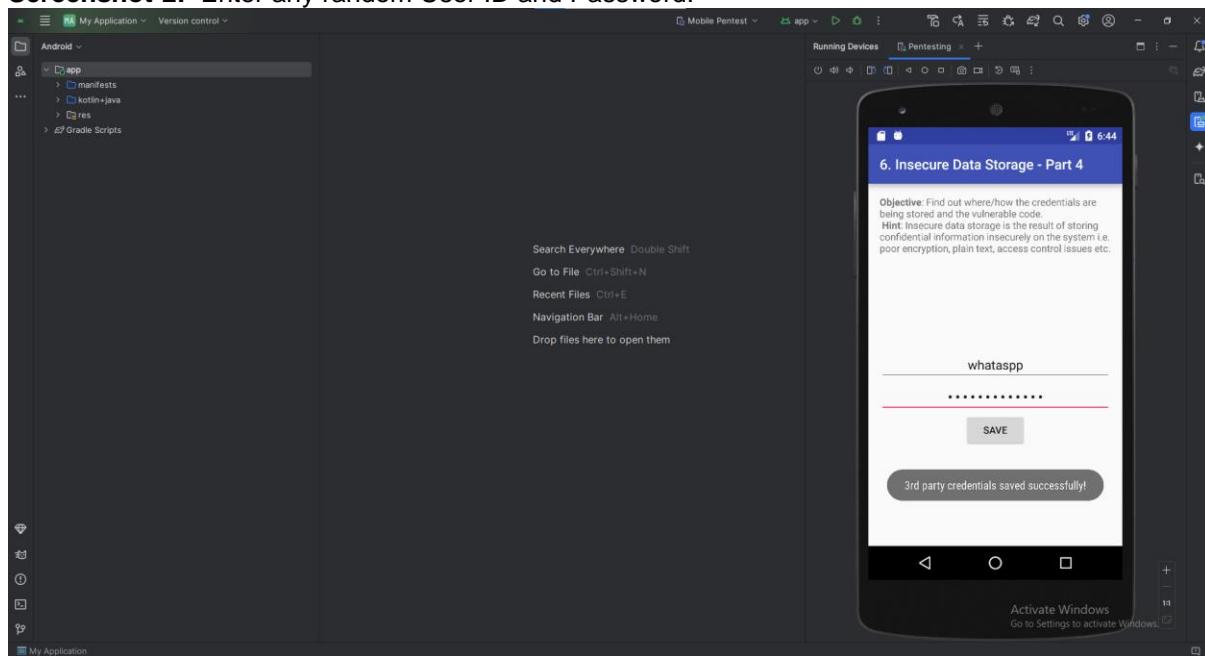
```
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> adb shell
root@generic_x86:/ # cd /data/data/jakhar.aseem.diva
root@generic_x86:/data/data/jakhar.aseem.diva # ls
cache
code_cache
databases
lib
shared_prefs
uinfo-1541151966tmp
at uinfo-1541151966tmp
facebook:facebook@pass
root@generic_x86:/data/data/jakhar.aseem.diva # cat uinfo-1541151966tmp
Facebook:facebook@pass
root@generic_x86:/data/data/jakhar.aseem.diva #
```

## 6. Insecure Data Storage\_Part-4

Screenshot-1: Enable the storage permission of the diva app.



Screenshot-2: Enter any random User ID and Password.



**Screenshot-3:** Respective activity & its source code. We can see that entered credentials are stored in a file name as .uinfo.txt inside of sdcard external storage.

The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The left sidebar shows the project structure with files like base.apk, META-INF/MANIFEST.MF, and various Java and XML files.
- Code Editor:** The main window displays the `InsecureDataStorage4Activity.java` file. The code is as follows:

```
package jahkar.aesem.diva;

import android.os.Bundle;
import android.os.Environment;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
import java.io.File;
import java.io.FileWriter;

/* Loaded from: classes.dex */
public class InsecureDataStorage4Activity extends AppCompatActivity {
    /* Java Dex file modifiers changed from protected */
    @Override // android.support.v7.app.AppCompatActivity, android.support.v4.app.FragmentActivity, android.support.v4.app.BaseFragmentActivityDonut, android.app.Activity
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_insecure_data_storage4);
    }

    public void saveCredentials(View view) {
        EditText usr = findViewById(R.id.ids4usr);
        EditText pwd = findViewById(R.id.ids4pwd);
        String sdri = Environment.getExternalStorageDirectory();
        try {
            File userinfo = new File(sdri.getAbsolutePath() + "/userinfo.txt");
            userinfo.setReadable(true);
            userinfo.createNewFile();
            FileWriter fw = new FileWriter(userinfo);
            fw.write(usr.getText().toString() + ":" + pwd.getText().toString() + "\n");
            fw.close();
            Toast.makeText(this, "3rd party credentials saved successfully!", 0).show();
        } catch (Exception e) {
            Toast.makeText(this, "file error occurred.", 0).show();
            Log.d("Dive", "file error: " + e.getMessage());
        }
    }
}
```

### Screenshot-3:

Go to root using `adb shell` and go to the path `/mnt/sdcard`

Now, we can find the hidden file `.uiinfo.txt` and can see the credentials as content of the file.

```
root@generic_x86:/mnt # cd sdcard
root@generic_x86:/mnt/sdcard # ls -la
drw-rw---- root    sdcard_rw      23 2024-08-03 18:44 .uiinfo.txt
drwxrwxr-x root    sdcard_rw      2024-07-07 20:03 Alarms
drwxrwxr-x root    sdcard_rw      2024-07-07 20:03 Android
drwxrwxr-x root    sdcard_rw      2024-07-07 20:04 DCIM
drwxrwxr-x root    sdcard_rw      2024-07-07 20:03 Download
drwxrwxr-x root    sdcard_rw      2024-07-07 20:03 Movies
drwxrwxr-x root    sdcard_rw      2024-07-07 20:03 Music
drwxrwxr-x root    sdcard_rw      2024-07-07 20:03 Notifications
drwxrwxr-x root    sdcard_rw      2024-07-07 20:03 Pictures
drwxrwxr-x root    sdcard_rw      2024-07-07 20:03 Podcasts
drwxrwxr-x root    sdcard_rw      2024-07-07 20:03 Ringtones
root@generic_x86:/mnt/sdcard # cat .uiinfo.txt
WhatsApp:WhatsApp@1234
root@generic_x86:/mnt/sdcard #
```

## **7. Input Validation Issues \_ Part-1**

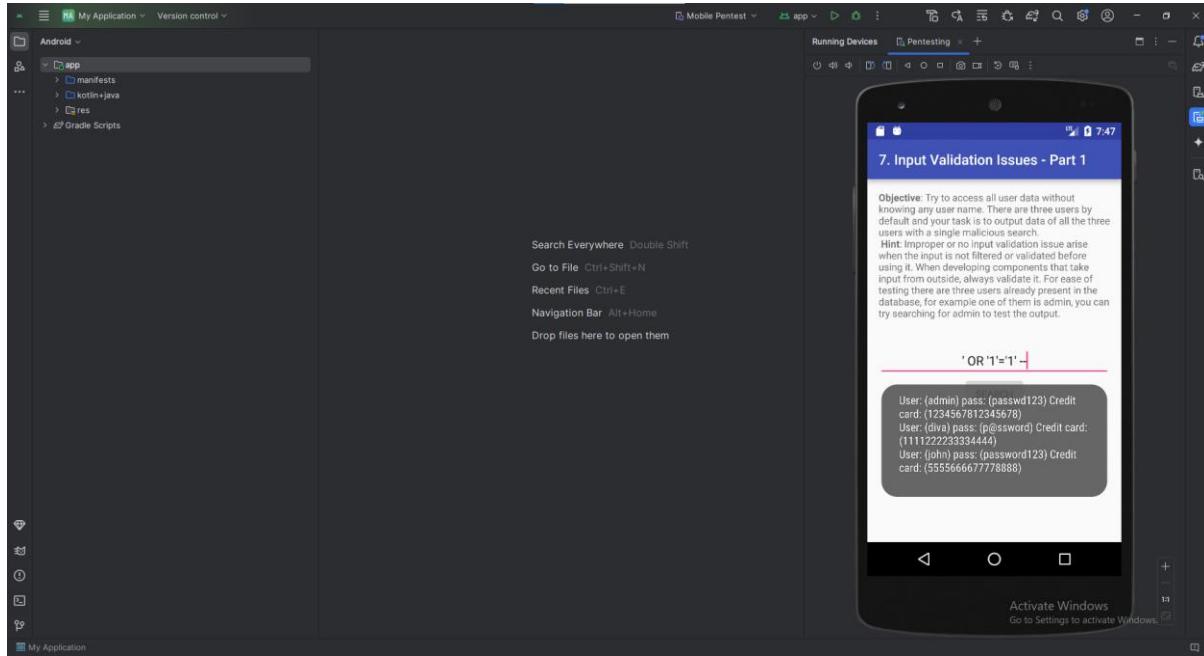
**Screenshot-1:** Check the AndroidManifest.xml file to find out the exact activity.

```
File View Navigation Tools Plugins Help *base.apk [D:\\Security ALL\\Mobile Application Penetration Testing\\DIVA APK Security Assessment\\DivApenTesting] - jd-gui  
META-INF/MANIFEST.MF AndroidManifest.xml InsecureDataStorage4Activity LogActivity MainActivity SQLInjectionActivity  
base.apk  
Inputs  
Files  
base.apk  
Scripts  
Source code  
android.support  
Jahar..asseen.diva  
AccessControl1Activity  
AccessControl2Activity  
AccessControl3Activity  
AccessControl3NotesActivity  
APICreds2Activity  
APICreds3Activity  
BuildConfig  
DivA1n1  
Hardcode1Activity  
HardcodeActivity  
InputValidation2URISchemeActivity  
InputValidation3Activity  
InsecureDataStorage1Activity  
InsecureDataStorage2Activity  
InsecureDataStorage3Activity  
InsecureDataStorage4Activity  
LogActivity  
MainActivity  
NotesProvider  
R  
SQLInjectionActivity  
Resources  
lib  
META-INF  
res  
AndroidManifest.xml  
classes.dex  
resources.arsc  
APK signature  
Summary  
21    android:label="@string/app_name"  
22    android:icon="@mipmap/ic_launcher"  
23    android:debuggable="true"  
24    android:allowBackup="true"  
25    android:theme="@style/AppTheme"  
26    <activity  
27        android:name=".MainActivity"  
28        android:label="@string/app_name"  
29        android:icon="@mipmap/ic_launcher"  
30        android:launchMode="singleTask"  
31        <intent-filter  
32            <action android:name="android.intent.action.MAIN"/>  
33            <category android:name="android.intent.category.LAUNCHER"/>  
34        </intent-filter>  
35    <activity  
36        android:label="@string/d1"  
37        android:name=".LogActivity"/>  
38    <activity  
39        android:label="@string/d2"  
40        android:name=".HardcodeActivity"/>  
41    <activity  
42        android:label="@string/d3"  
43        android:name=".InsecureDataStorage1Activity"/>  
44    <activity  
45        android:label="@string/d4"  
46        android:name=".InsecureDataStorage2Activity"/>  
47    <activity  
48        android:label="@string/d5"  
49        android:name=".InsecureDataStorage3Activity"/>  
50    <activity  
51        android:label="@string/d6"  
52        android:name=".InsecureDataStorage4Activity"/>  
53    <activity  
54        android:label="@string/d7"  
55        android:name=".SQLInjectionActivity"/>  
56    <activity  
57        android:label="@string/d8"  
58        android:name=".InputValidation2URISchemeActivity"/>  
59    <activity  
60        android:label="@string/d9"  
61        android:name=".AccessControl1Activity"/>  
62    <activity  
63        android:label="@string/d10"  
64        android:name=".APICreds3Activity"/>  
65    <intent-filter  
66        <action android:name="jahar..asseen.diva.action.VIEW_CREDITS"/>  
67        <category android:name="android.intent.category.DEFAULT"/>  
68    </intent-filter>  
69    <activity
```

**Screenshot-2:** Respective activity & its source code. We can see that user information details are stored in SQL database.

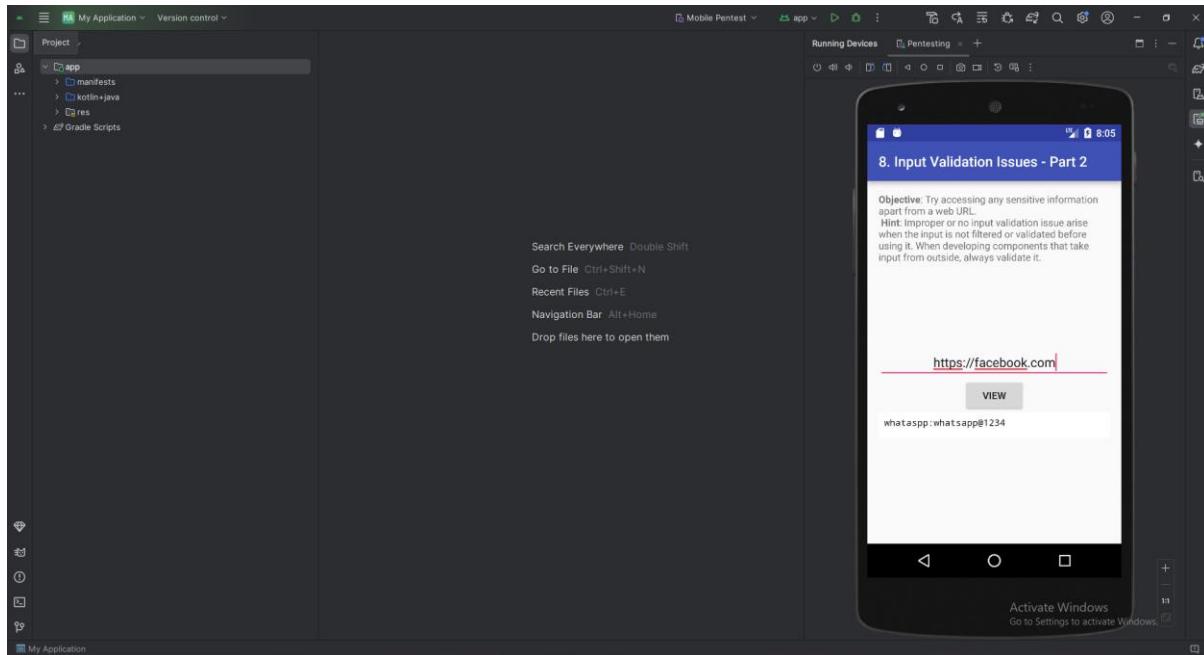
```
File View Navigation Tools Plugins Help  
base.apk [D:\\Security\\ALL\\Mobile Application Penetration Testing\\DIVA APK Security Assessment\\DivA-Pentesting] - jd-gui  
META-INF/MANIFEST.MF AndroidManifest.xml InsecureDataStorage4Activity LogActivity MainActivity SQLInjectionActivity  
base.apk  
Inputs  
Files  
base.apk  
Scripts  
Source code  
android.support  
Jahkar.aesem.diva  
AccessControlActivity  
AccessControlActivity  
AccessControlActivity  
AccessControl3NotesActivity  
APTCreds2Activity  
APTCredsActivity  
BuildConfig  
DivActivity  
HardcodeActivity  
HardcodeActivity  
InputValidation2URISchemeActivity  
InputValidation3Activity  
InsecureDataStorageActivity  
InsecureDataStorage3Activity  
InsecureDataStorage4Activity  
LogActivity  
MainActivity  
NotesProvider  
R  
SQLInjectionActivity  
Resources  
lib  
META-INF  
res  
AndroidManifest.xml  
classes.dex  
resources.arsc  
APK signature  
Summary  
base.apk  
META-INF/MANIFEST.MF  
AndroidManifest.xml  
InsecureDataStorage4Activity  
LogActivity  
MainActivity  
SQLInjectionActivity  
package jahkar.aesem.diva;  
  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
import android.support.v7.app.AppCompatActivity;  
import android.util.Log;  
import android.view.View;  
import android.widget.EditText;  
import android.widget.Toast;  
  
/* Loaded from: C:\Users\dev\ * */  
public class SQLInjectionActivity extends AppCompatActivity {  
    private SQLiteDatabase mDB;  
  
    /* JAD2 user Access modifiers changed from: protected */  
    /* JADX DEBUG: Method arguments not matched by any found parameters */  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        try {  
            this.mDB = openOrCreateDatabase("sql", 0, null);  
            this.mDB.execSQL("CREATE TABLE IF NOT EXISTS sqliuser (user VARCHAR, password VARCHAR, credit_card VARCHAR);");  
            this.mDB.execSQL("INSERT INTO sqliuser VALUES ('admin', 'password123', '12345678901234567890');");  
            this.mDB.execSQL("INSERT INTO sqliuser VALUES ('dive', 'ppassword', '111122233344445555');");  
            this.mDB.execSQL("INSERT INTO sqliuser VALUES ('John', 'password123', '5555666777888899');");  
        } catch (Exception e) {  
            Log.d("Dive-sql", "Error occurred while creating database for SQL: " + e.getMessage());  
        }  
        setContentView(R.layout.activity_sqlinjection);  
    }  
  
    public void search(View view) {  
        EditText srchtxt = (EditText) findViewById(R.id.lvisearch);  
        try {  
            Cursor cr = this.mDB.rawQuery("SELECT * FROM sqliuser WHERE user = " + srchtxt.getText().toString() + " OR ", null);  
            StringBuilder strb = new StringBuilder();  
            if (cr != null & cr.getCount() > 0) {  
                cr.moveToFirst();  
                do {  
                    strb.append("User: (" + cr.getString(0) + ") pass: (" + cr.getString(1) + ") Credit card: (" + cr.getString(2) + ")\n");  
                } while (cr.moveToNext());  
            } else {  
                strb.append("User: (" + srchtxt.getText().toString() + ") not found");  
            }  
            Toast.makeText(this, strb.toString(), 0).show();  
        } catch (Exception e) {  
            Log.d("Dive-sql", "Error occurred while searching in database: " + e.getMessage());  
        }  
    }  
}
```

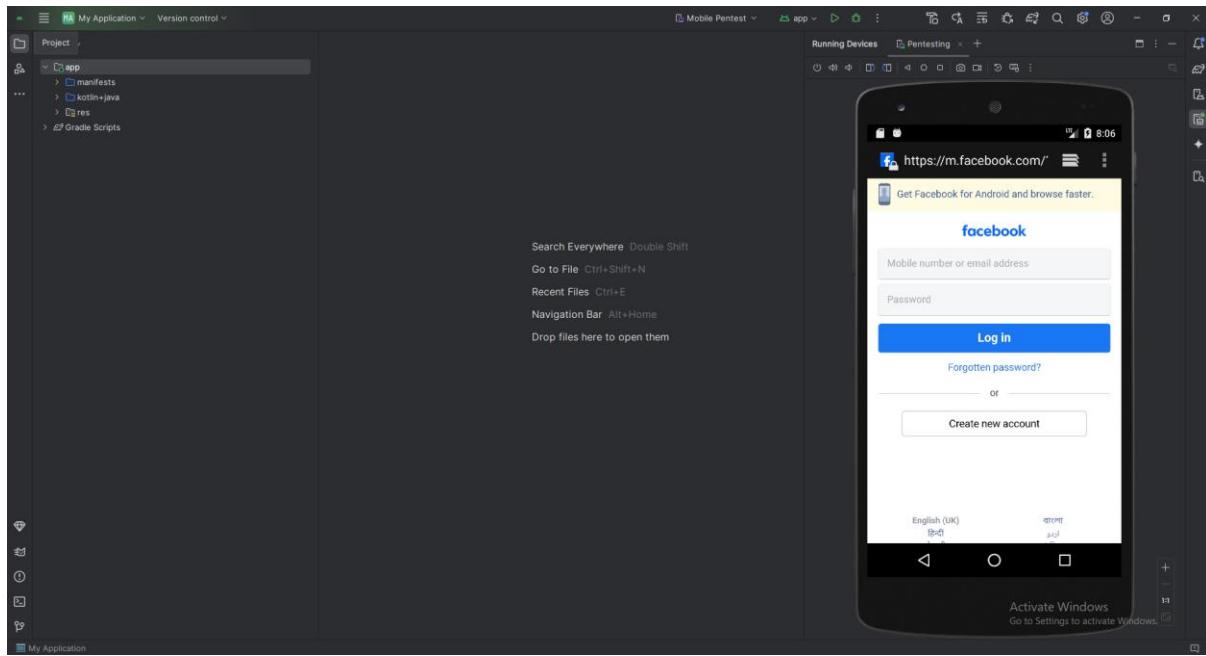
**Screenshot-3:** We can perform SQL Injection with single malicious payload ‘ OR ‘1’=’1’ – and can see all user’s details.



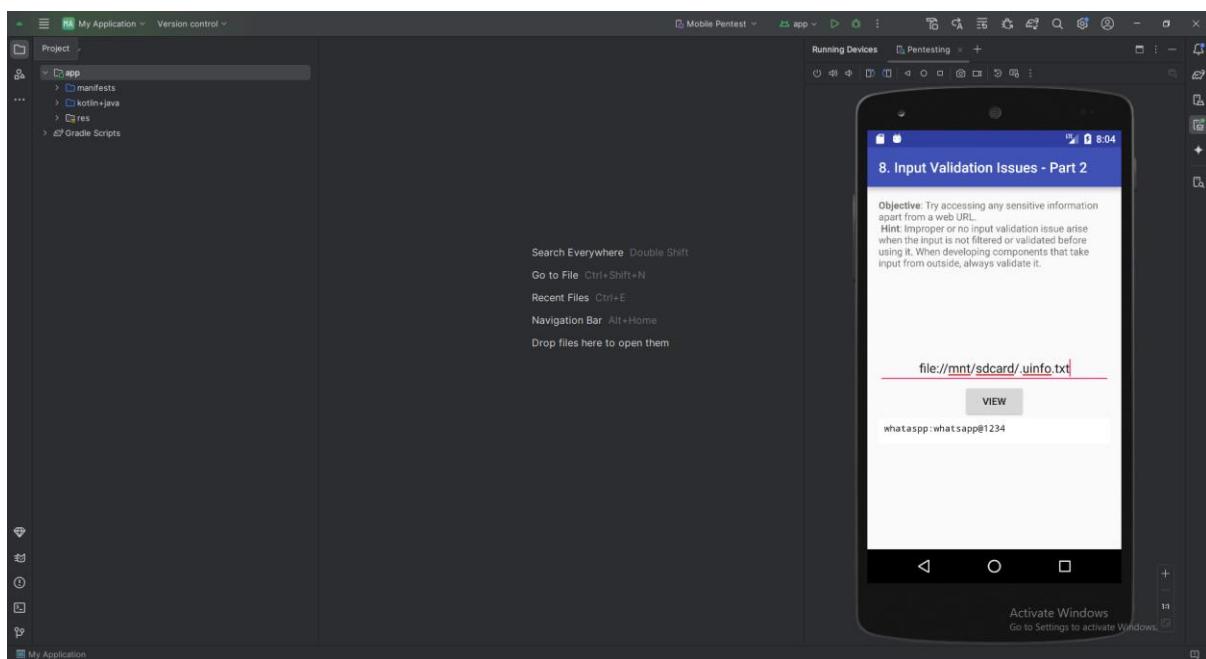
## 8. Input Validation Issues \_Part-2

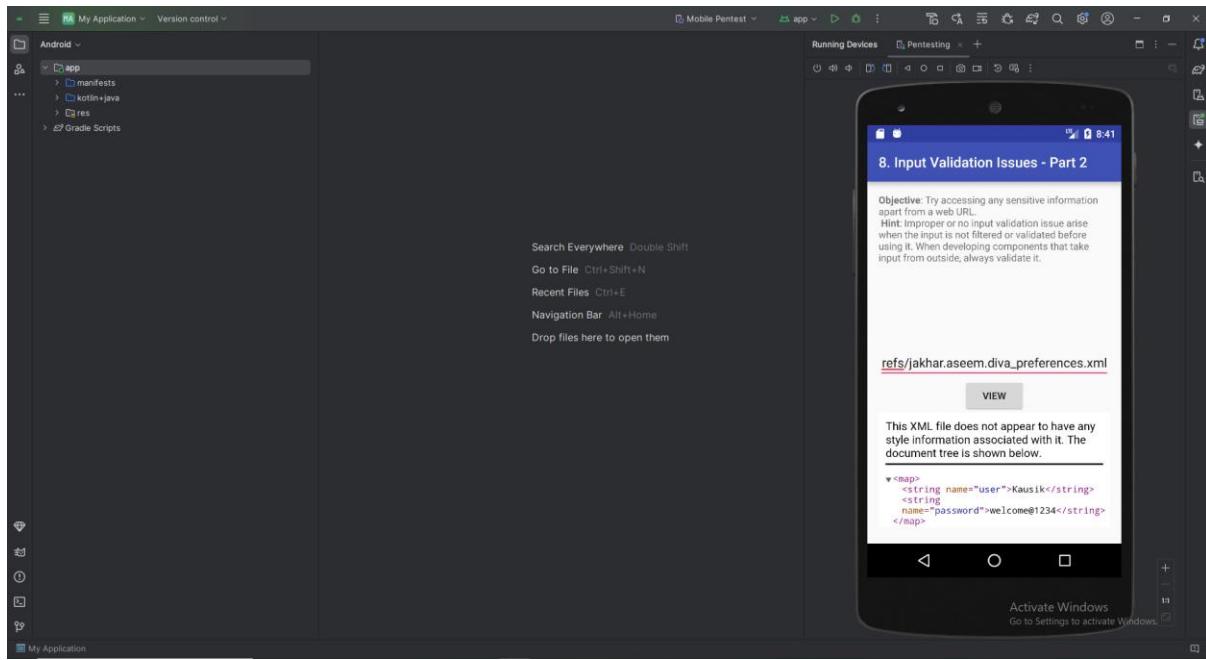
**Screenshot-1:** Type any URL and see that webpage is working as normal.





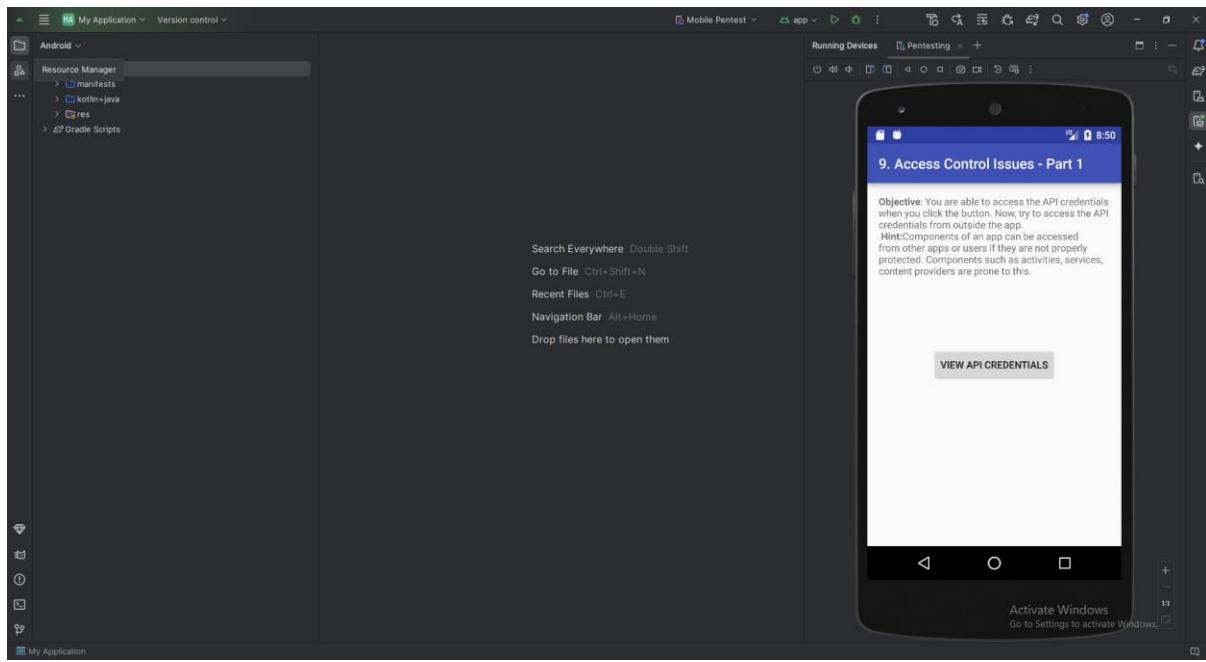
**Screenshot-2:** Enter any file path `file:///mnt/sdcard/.uiinfo.txt` and see the content. This is called Local File Inclusion (LFI).



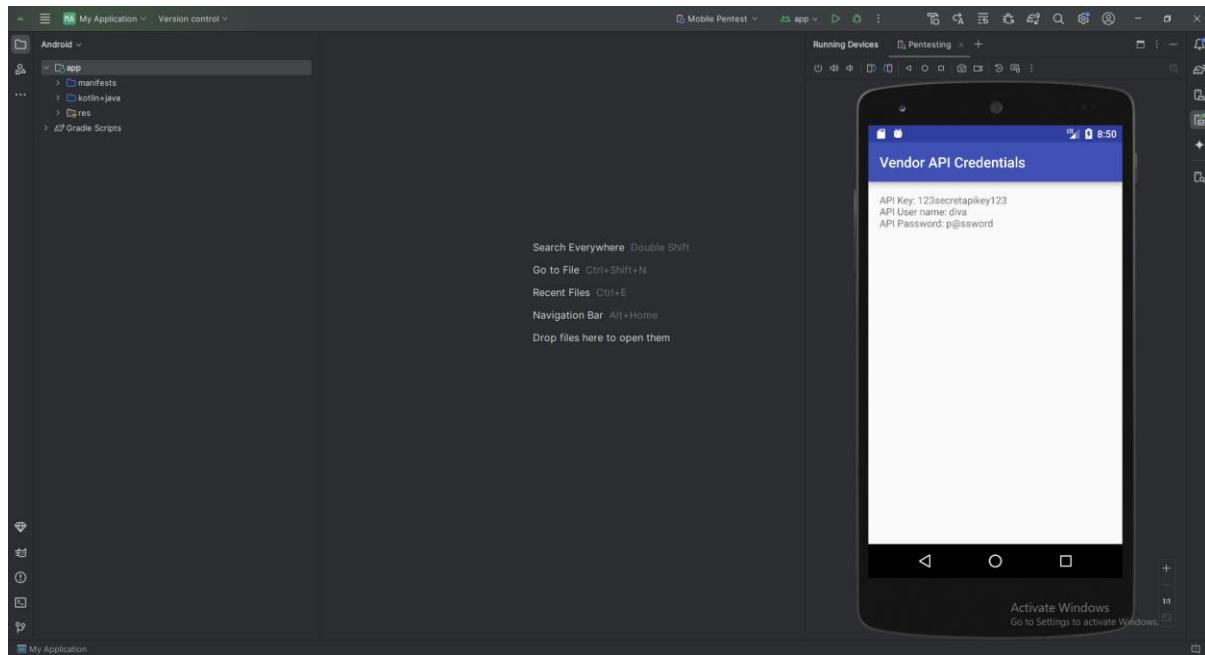


## 9. Access Control Issues \_ Part-1

**Screenshot-1:** Until we are clicking the button, API credentials will not be shown.



**Screenshot-2:** Once we are clicking the button, API credentials will be shown.



**Screenshot-3:** Respective activity & its source code. We can see the activity name by starting which we can see the API credentials on the devices without open the app explicitly.

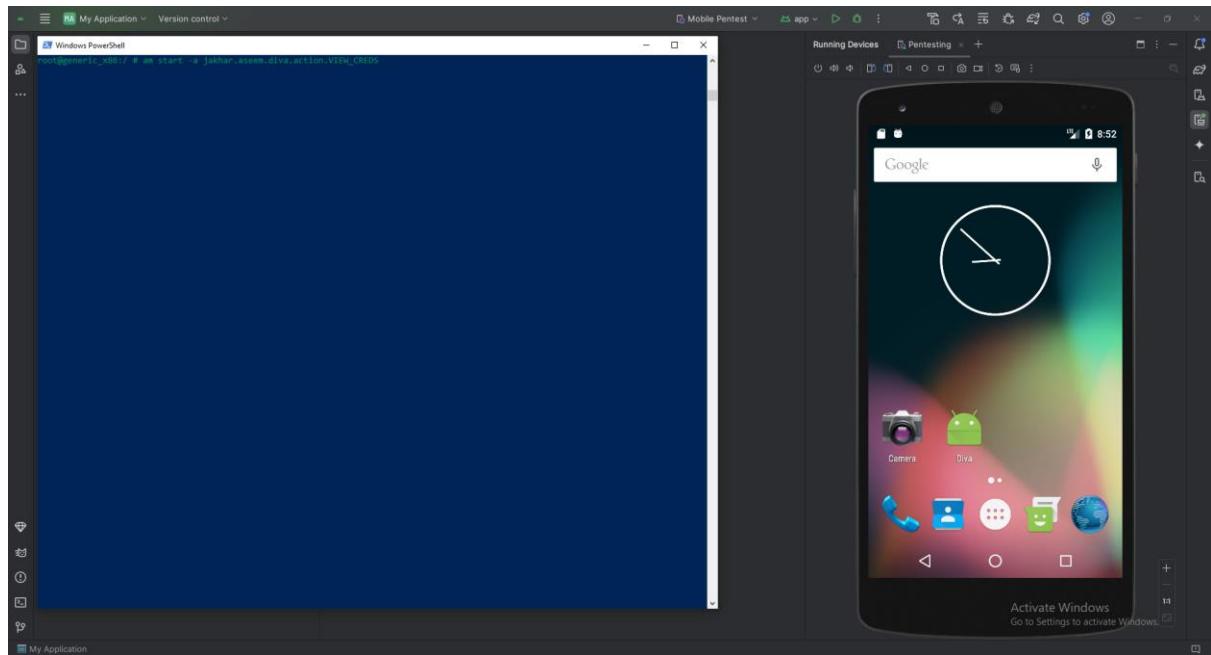
The screenshot shows the JD-GUI Java decompiler interface. The left sidebar shows the file structure of the APK, including 'base.apk' and various Java files like 'AccessControl1Activity', 'InsecureDataStorage4Activity', etc. The main pane displays the source code of 'AccessControl1Activity'. A red box highlights the following code snippet:

```

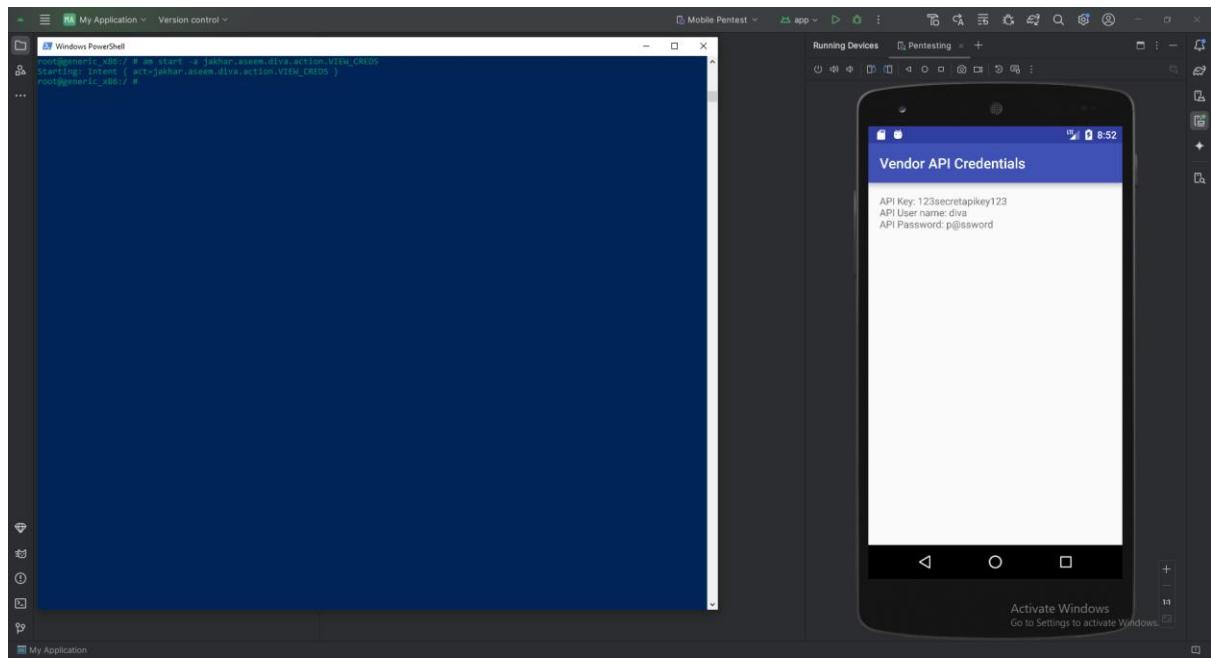
12     public class AccessControl1Activity extends AppCompatActivity {
13         @Override // From android.support.v7.app.AppCompatActivity
14         protected void onCreate(Bundle savedInstanceState) {
15             super.onCreate(savedInstanceState);
16             setContentView(R.layout.activity_access_control);
17         }
18
19         public void viewAPICredentials(View view) {
20             Intent i = new Intent();
21             i.setComponent(new ComponentName("com.diva.accesscontrol", "com.diva.accesscontrol.ACCESS_CREDENTIALS"));
22             if (i.resolveActivity(getApplicationContext()) != null) {
23                 startActivity(i);
24             } else {
25                 Toast.makeText(this, "Error while getting API details", 0).show();
26                 Log.e("Diva-acct", "Couldn't resolve the Intent VIEW_CREDITS to our activity");
27             }
28         }
29     }

```

**Screenshot-4:** Now the DIVA app is closed.

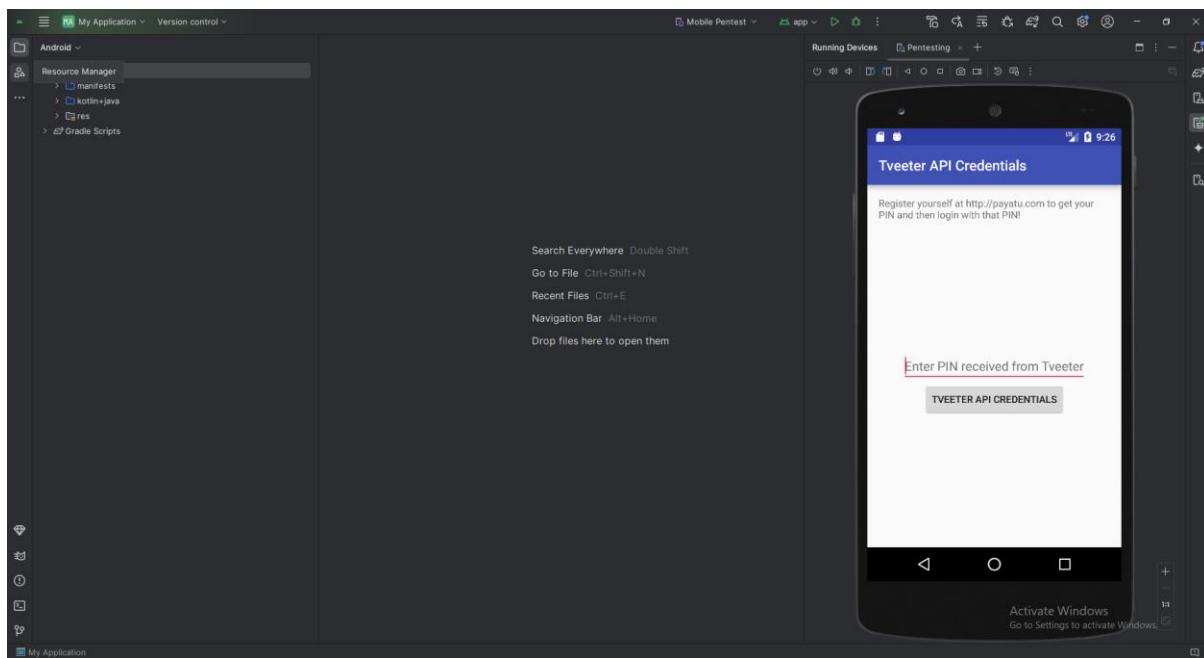
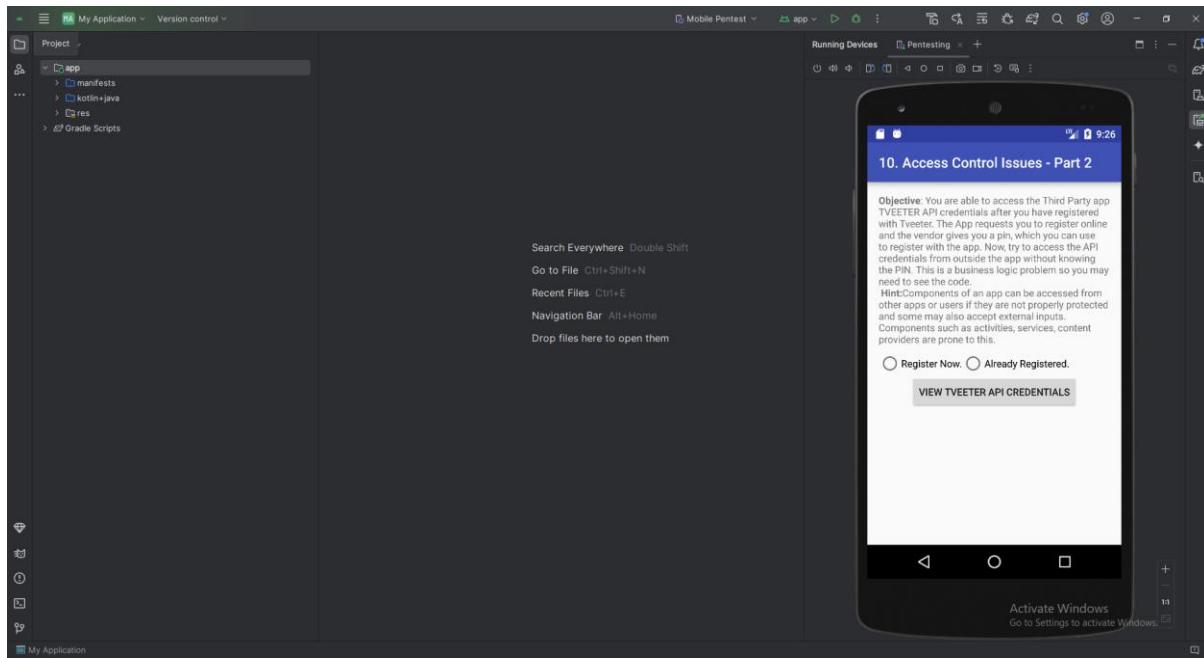


**Screenshot-5:** Now from adb shell try to start the activity by using the command `am start - a < jakhar.aseem.diva.action.VIEW_CREDS >` and after entering hit API Credentials are exposing on the emulator without validating any authorization.

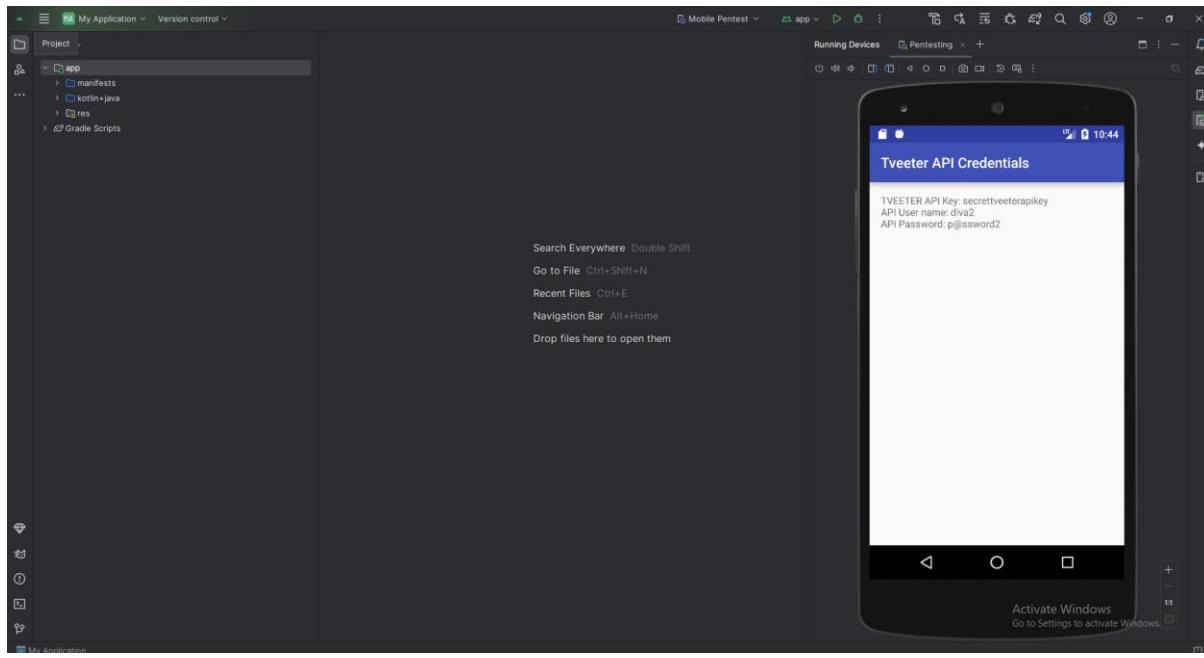


## 10. Access Control Issues \_ Part-2

**Screenshot-1:** The 1<sup>st</sup> Activity is `jakhar.aseem.diva.APICreds2Activity` with chk pin verification step.



**Screenshot-2:** The 2nd Activity is `jakhar.aseem.diva.action.VIEW_CREDS2` which will show API Credentials.



**Screenshot-3:** 1<sup>st</sup> and 2<sup>nd</sup> activity name details from AndroidManifest.xml file.

The screenshot shows the JD-GUI interface with the AndroidManifest.xml file open. The manifest contains numerous activities, many of which are highlighted in red, likely indicating they are part of the attack vector or have been identified by the tool. The activities include:

- InsecureDataStorageActivity
- LogActivity
- MainActivity
- SQLInjectionActivity
- AccessControl1Activity
- AccessControl2Activity
- AccessControl3Activity
- AccessControl3NotesActivity
- APICreds2Activity
- APICreds3Activity
- BuildConfig
- DivaJni
- HandcodeActivity
- InputValidationActivity
- InputValidation2URISchemeActivity
- InputValidation3Activity
- InsecureDataStorage1Activity
- InsecureDataStorage2Activity
- InsecureDataStorage3Activity
- InsecureDataStorage4Activity
- LogActivity
- MainActivity
- NotesProvider
- R
- SQLInjectionActivity

The manifest also includes provider declarations for NotesProvider and AccessControl3Activity.

**Screenshot-4:** Respective activity & its source code. We can see that before start 2<sup>nd</sup> Activity (VIEW\_CRED\$2) one chk pin validation point is there.

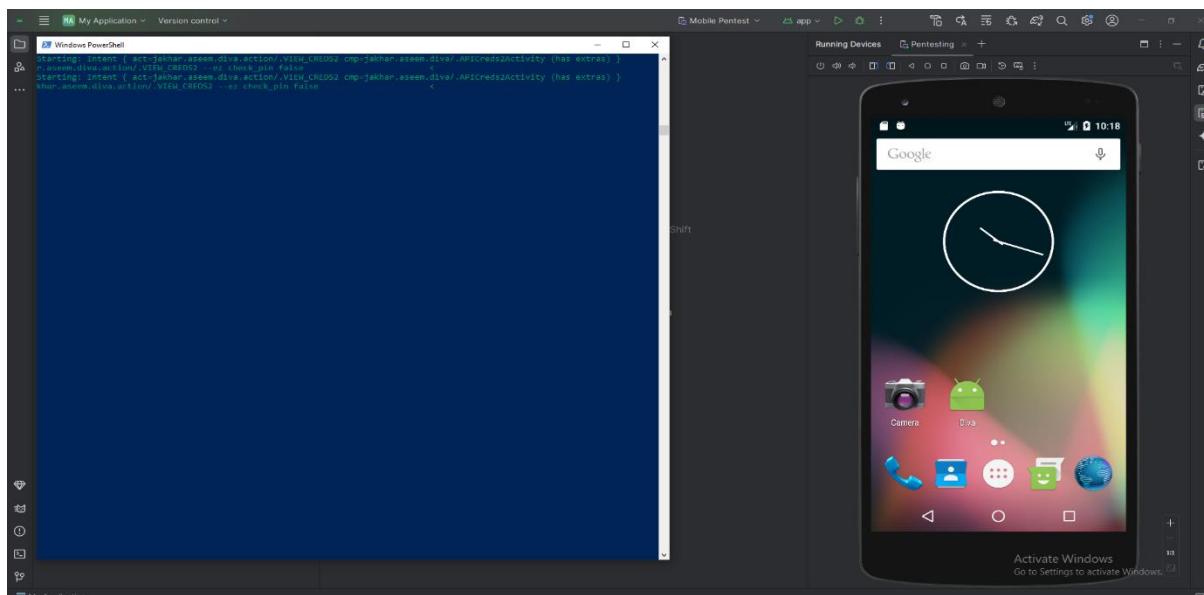
The screenshot shows the Android Studio interface with the project navigation bar at the top. The project structure on the left includes folders for mips, mips64, x86, x86\_64, META-INF, CERT.RSA, CERT\_SF, NAMEIFEST.MF, res (AndroidManifest.xml, classes.dex), resources.arsc, and resources.arsc (res folder). The strings.xml file is selected in the resources.arsc folder.

The code editor displays the contents of strings.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="abc_action_menu_overflow_description">More options</string>
    <string name="abc_action_mode_done">Done</string>
    <string name="abc_activity_chrome_view_see_all">See all</string>
    <string name="abc_applications_selection_application">Choose an app</string>
    <string name="abc_capital_on_0K">Off</string>
    <string name="abc_share_on_0K">On</string>
    <string name="abc_search_bar_search">Search</string>
    <string name="abc_searchview_description_clear">Clear query</string>
    <string name="abc_searchview_description_query">Search query</string>
    <string name="abc_searchview_description_search">Search</string>
    <string name="abc_searchview_description_voice">Voice search</string>
    <string name="abc_shareactionprovider_share_with">Share with</string>
    <string name="abc_shareactionprovider_share_with_acitvity">Share with %s</string>
    <string name="abc_shareactionprovider_share_with_collection">Share with Collection</string>
    <string name="status_bar_notification_info_overflow">999+</string>
    <string name="wail_intro">Objective: You are able to access the API credentials when you click the button. Now, try to access the API credentials from outside the app.  
In Hint: Component</string>
    <string name="wail_already_registered">Already Registered.</string>
    <string name="wail_intro">Objective: You are able to access the Third Party app TWEETER API credentials after you have registered with Tweeter. The App requests you to register online</string>
    <string name="wail_register">Register Now</string>
    <string name="wail_accessories">ACCESS PRIVATE NOTES</string>
    <string name="wail_createpin">CREATE/CHANGE PIN</string>
    <string name="wail_enterpin">Enter PIN</string>
    <string name="wail_pin">GO TO PRIVATE NOTES</string>
    <string name="wail_intro">Objective: This is a private notes application. You can create a PIN once and access your notes after entering the correct pin. Now, try to access the private notes</string>
    <string name="wail_settings">Settings</string>
    <string name="wail_tweeter">TWEETER API Credentials</string>
    <string name="wail_vendor">Vendor API Credentials</string>
    <string name="wail_diva">Diva</string>
    <string name="character_counter_gatten">%s/ds/Xsd</string>
    <string name="check_pin">Check pin</string>
    <string name="d10">Access Control Issues - Part 1</string>
    <string name="d10_10">Access Control Issues - Part 2</string>
    <string name="d11">Access Control Issues - Part 3</string>
    <string name="d12">Hardcoding Issues - Part 2</string>
    <string name="d12_10">Hardcoding Issues - Part 1</string>
    <string name="d12_20">Hardcoding Issues - Part 2</string>
    <string name="d13">Insecure Data Storage - Part 1</string>
    <string name="d13_10">Insecure Data Storage - Part 2</string>
    <string name="d13_20">Insecure Data Storage - Part 3</string>
    <string name="d14">Input Validation Issues - Part 1</string>
    <string name="d14_10">Input Validation Issues - Part 2</string>
    <string name="d15">Insufficient Encryption - Part 1</string>
    <string name="d15_10">Insufficient Encryption - Part 2</string>
    <string name="d16">Diva (Open insecure and vulnerable App) is an App intentionally designed to be insecure. The aim of the App is to teach developers/QA/security professionals, how to identify security issues in an application</string>
    <string name="welcome">Welcome to DIVA!</string>
    <string name="hardcode_error">Enter the vendor key</string>
    <string name="hardcode_intro">Objective: Find out what is hardcoded and where.\nHint: Developers sometimes will hardcode sensitive information for ease.</string>
    <string name="hardcode_error2">Enter the vendor key</string>
    <string name="hardcode_intro2">Objective: Find out where the credentials are being stored and the vulnerable code.\nHint: Insecure data storage is the result of storing confidential information in plain text</string>
    <string name="l1d_password">Enter 3rd party service password</string>
    <string name="l1d_save">SAVE</string>
    <string name="l1d_user">Enter 3rd party service user name</string>
    <string name="l1l_enter">Enter user name to search</string>

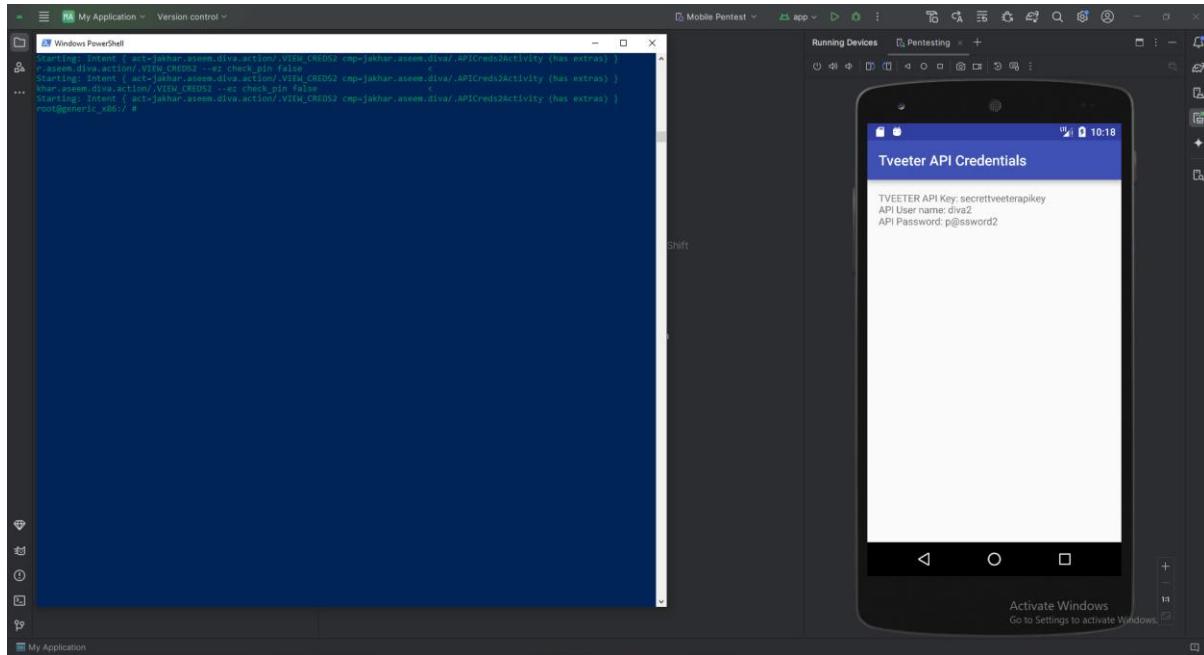
```

**Screenshot-5:** Now the DIVA app is closed.



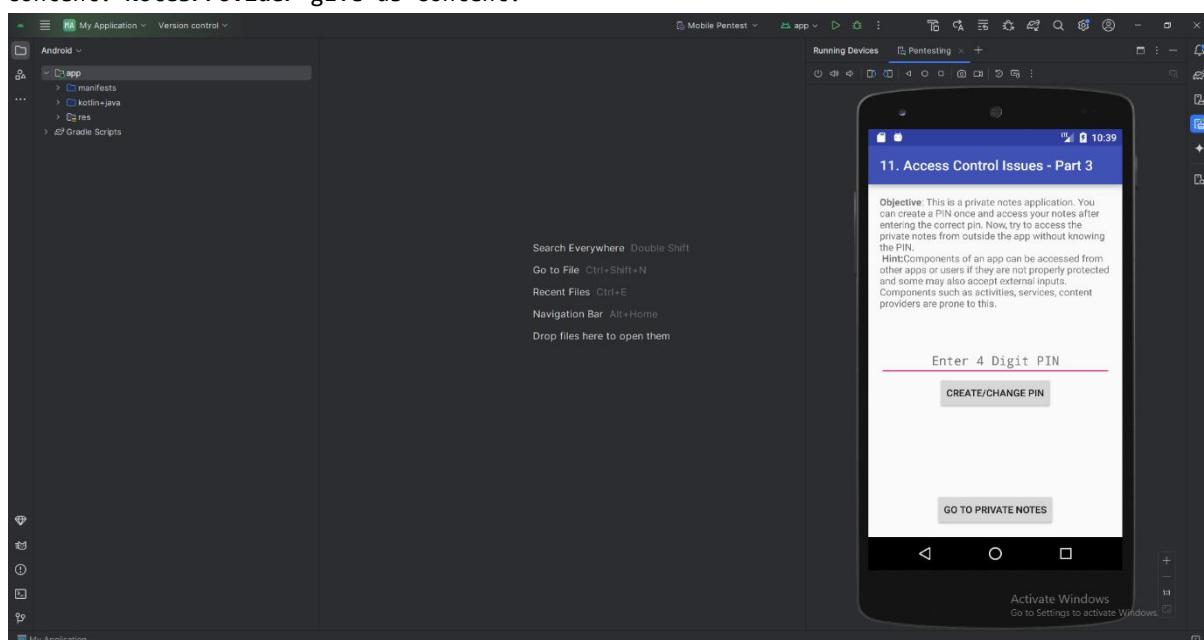
**Screenshot-6:** Now from adb shell try to execute the command  
`am start -n jakhar.aseem.diva/.APICreds2Activity -a jakhar.aseem.diva.action/.VIEW_CREDS2 --ez check_pin false`  
and after entering hit API Credentials are exposing on the emulator without validating any authorization.

`am` = Activity Manager  
`-n` = Do not check whether I am verified or not.  
`-a` = Start an activity  
`--ez check_pin false` = Value of the extra variable (check\_pin) make as false as we don't know the pin.



## 11. Access Control Issues \_ Part-3

**Screenshot-1:** The 1<sup>st</sup> Activity is **AccessControl3Activity** which is responsible for creation of the PIN. The 2<sup>nd</sup> Activity is **AccessControl3NotesActivity** which is responsible to store the content. **NotesProvider** give us content.



File View Navigation Tools Plugins Help

base.apk [D:\Security\All\Mobile Application Penetration Testing\DIVA APK Security Assessment\DivA Pентesting] - jadx-gui

base.apk

- Inputs
- Files
- Source code
- android.support
- Jakhar.assem.diva
  - AccessControl1Activity
  - AccessControl2Activity
  - AccessControl3Activity**
  - AccessControl3NotesActivity
  - APICreds2Activity
  - APICredsActivity
  - DivaJni
  - Hardcode2Activity
  - HardcodeActivity
  - InputValidationURISchemeActivity
  - InputValidation3Activity
  - InsecureDataStorage1Activity
  - InsecureDataStorage2Activity
  - InsecureDataStorage3Activity
  - InsecureDataStorage4Activity
  - LogActivity
  - MainActivity
  - NotesProvider
  - R
  - SQLInjectionActivity
- Resources
- lib
- META-INF
- res
- AndroidManifest.xml
- classes.dex
- resources.arsc
- APK signature
- Summary

META-INF/MANIFEST.MF    AndroidManifest.xml    AccessControl3Activity    AccessControl3NotesActivity    NotesProvider    R

```

package jakhar.assem.diva;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.preference.PreferenceManager;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;

/* Loaded from: classes.dex */
16 public class AccessControl3Activity extends AppCompatActivity {
17     /* [ADK INFO] Access modifiers changed from: protected */
18     /* [ADK INFO] Access modifiers changed from: protected */
19     public void onCreate(Bundle savedInstanceState) {
20         super.onCreate(savedInstanceState);
21         setContentView(R.layout.activity_access_control);
22         SharedPreferences spref = PreferenceManager.getDefaultSharedPreferences(this);
23         String pin = spref.getString(R.string.pkey, "");
24         if (!pin.isEmpty()) {
25             Button button = (Button) findViewById(R.id.aci3viewbutton);
26             button.setVisibility(8);
27         }
28     }

29     public void addPin(View view) {
30         SharedPreferences spref = PreferenceManager.getDefaultSharedPreferences(this);
31         EditText editText = (EditText) findViewById(R.id.aci3pin);
32         EditText pinTxt = (EditText) findViewById(R.id.aci3pin);
33         String pin = pinTxt.getText().toString();
34         if (pin == null || pin.isEmpty()) {
35             Toast.makeText(this, "Please Enter a valid pin!", 0).show();
36             return;
37         }
38         Button button = (Button) findViewById(R.id.aci3viewbutton);
39         spref.putString(getString(R.string.pkey), pin);
40         spref.commit();
41         if (button.getVisibility() != 8) {
42             button.setVisibility(8);
43         }
44     }

45     public void goToNotes(View view) {
46         Intent i = new Intent(this, AccessControl3NotesActivity.class);
47         startActivity(i);
48     }
}

```

Activate Windows  
Go to Settings to activate Windows.

File View Navigation Tools Plugins Help

base.apk [D:\Security\All\Mobile Application Penetration Testing\DIVA APK Security Assessment\DivA Pентesting] - jadx-gui

base.apk

- Inputs
- Files
- Source code
- android.support
- Jakhar.assem.diva
  - AccessControl1Activity
  - AccessControl2Activity
  - AccessControl3Activity**
  - AccessControl3NotesActivity**
  - APICreds2Activity
  - APICredsActivity
  - BuildConfig
  - DivaJni
  - Hardcode2Activity
  - HardcodeActivity
  - InputValidationURISchemeActivity
  - InputValidation3Activity
  - InsecureDataStorage1Activity
  - InsecureDataStorage2Activity
  - InsecureDataStorage3Activity
  - InsecureDataStorage4Activity
  - LogActivity
  - MainActivity
  - NotesProvider
  - R
  - SQLInjectionActivity
- Resources
- lib
- META-INF
- res
- AndroidManifest.xml
- classes.dex
- resources.arsc
- APK signature
- Summary

META-INF/MANIFEST.MF    AndroidManifest.xml    AccessControl3Activity    AccessControl3NotesActivity    NotesProvider    R

```

package jakhar.assem.diva;
import android.content.SharedPreferences;
import android.database.Cursor;
import android.os.Bundle;
import android.preference.PreferenceManager;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.SimpleCursorAdapter;
import android.widget.Toast;

/* Loaded from: classes.dex */
19 public class AccessControl3NotesActivity extends AppCompatActivity {
20     /* [ADK INFO] Access modifiers changed from: protected */
21     /* [ADK INFO] Access modifiers changed from: protected */
22     public void onCreate(Bundle savedInstanceState) {
23         super.onCreate(savedInstanceState);
24         setContentView(R.layout.activity_access_control_notes);
25     }

26     public void accessNotes(View view) {
27         EditText pinTxt = (EditText) findViewById(R.id.aci3notespinText);
28         Button button = (Button) findViewById(R.id.aci3inaccessbutton);
29         SharedPreferences spref = PreferenceManager.getDefaultSharedPreferences(this);
30         String pin = spref.getString(getString(R.string.pkey), "");
31         String userpin = pinTxt.getText().toString();
32         if (userpin.equals(pin)) {
33             ListView listView = (ListView) findViewById(R.id.aci3nlistView);
34             Cursor cr = getContentResolver().query(Uri.parse("content://"+NotesProvider.CONTENT_URI), new String[]{"_id", "title", "note"}, null, null, null);
35             String[] columns = {"title", "note"};
36             int[] ids = {R.id.title_entry, R.id.note_entry};
37             SimpleCursorAdapter adapter = new SimpleCursorAdapter(this, R.layout.notes_entry, cr, columns, ids, 0);
38             listView.setAdapter(adapter);
39             pinTxt.setvisibility(4);
40             button.setvisibility(4);
41             pinTxt.setvisibility(4);
42             button.setvisibility(4);
43         }
44     }

45     public void goToNotes(View view) {
46         Intent i = new Intent(this, AccessControl3NotesActivity.class);
47         startActivity(i);
48     }
}

```

Activate Windows  
Go to Settings to activate Windows.

Screenshot-2: NotesProvider give us content. Now copy the Content URI from here.

base.apk [D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapenting] - jadx-gui

META-INF/MANIFEST.MF    AndroidManifest.xml    AccessControl3Activity    AccessControlNotesActivity    NotesProvider    R

```

package jakhar.aseem.diva;
import android.content.ContentProvider;
import android.net.Uri;
import android.os.Bundle;
import android.content.Context;
import android.content.UriParser;
import android.database.Cursor;
import android.util.Log;
import android.util.Exception;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.net.Uri;
import android.text.TextUtils;

/* Loaded from: classes.dex */
public class NotesProvider extends ContentProvider {
    static final String AUTHORITY = "jakhar.aseem.diva.provider.notesprovider";
    static final String CREATE_TABLE = "CREATE TABLE notes (_id INTEGER PRIMARY KEY AUTOINCREMENT, title TEXT NOT NULL, note TEXT NOT NULL);";
    static final String C_ID = "_id";
    static final String C_NOTE = "note";
    static final String C_TITLE = "title";
    static final String DBNAME = "divanotes.db";
    static final int DBVERSION = 1;
    static final String DROP_TBL_QRY = "DROP TABLE IF EXISTS notes";
    static final int PATH_TABLE = 11;
    static final String TABLE = "notes";
    static final Uri CONTENT_URI = Uri.parse("content://jakhar.aseem.diva.provider.notesprovider/notes");
    static final UriMatcher urimatcher = new UriMatcher(-1);

    static {
        urimatcher.addURI(AUTHORITY, TABLE, 1);
        urimatcher.addURI(AUTHORITY, "notes/#", 2);
    }

    /* loaded from: classes.dex */
    private static class DBHelper extends SQLiteOpenHelper {
        public DBHelper(Context context) {
            super(context, NotesProvider.DBNAME, null, 1);
        }

        @Override // android.database.sqlite.SQLiteOpenHelper
        public void onCreate(SQLiteDatabase db) {
            db.execSQL(NotesProvider.DROP_TBL_QRY);
            db.execSQL(CREATE_TABLE);
            db.execSQL("INSERT INTO notes(title,note) VALUES ('office', '10 Meetings. 5 Calls. Lunch with CEO')");
            db.execSQL("INSERT INTO notes(title,note) VALUES ('home', 'Buy toys for baby, Order dinner')");
            db.execSQL("INSERT INTO notes(title,note) VALUES ('holiday', 'Either Goa or Amsterdam')");
            db.execSQL("INSERT INTO notes(title,note) VALUES ('work', 'Spent too much on home theater')");
            db.execSQL("INSERT INTO notes(title,note) VALUES ('Exercise', 'Alternate days running')");
            db.execSQL("INSERT INTO notes(title,note) VALUES ('Weekend', 'b333333333333333r')");
        }
    }

    @Override // android.database.sqlite.SQLiteOpenHelper
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    }
}

```

Activate Windows  
Go to Settings to activate Windows.

**Screenshot-3:** Now from adb shell try to execute the below command and fetch all details.

content query --uri content://jakhar.aseem.diva.provider.notesprovider/notes

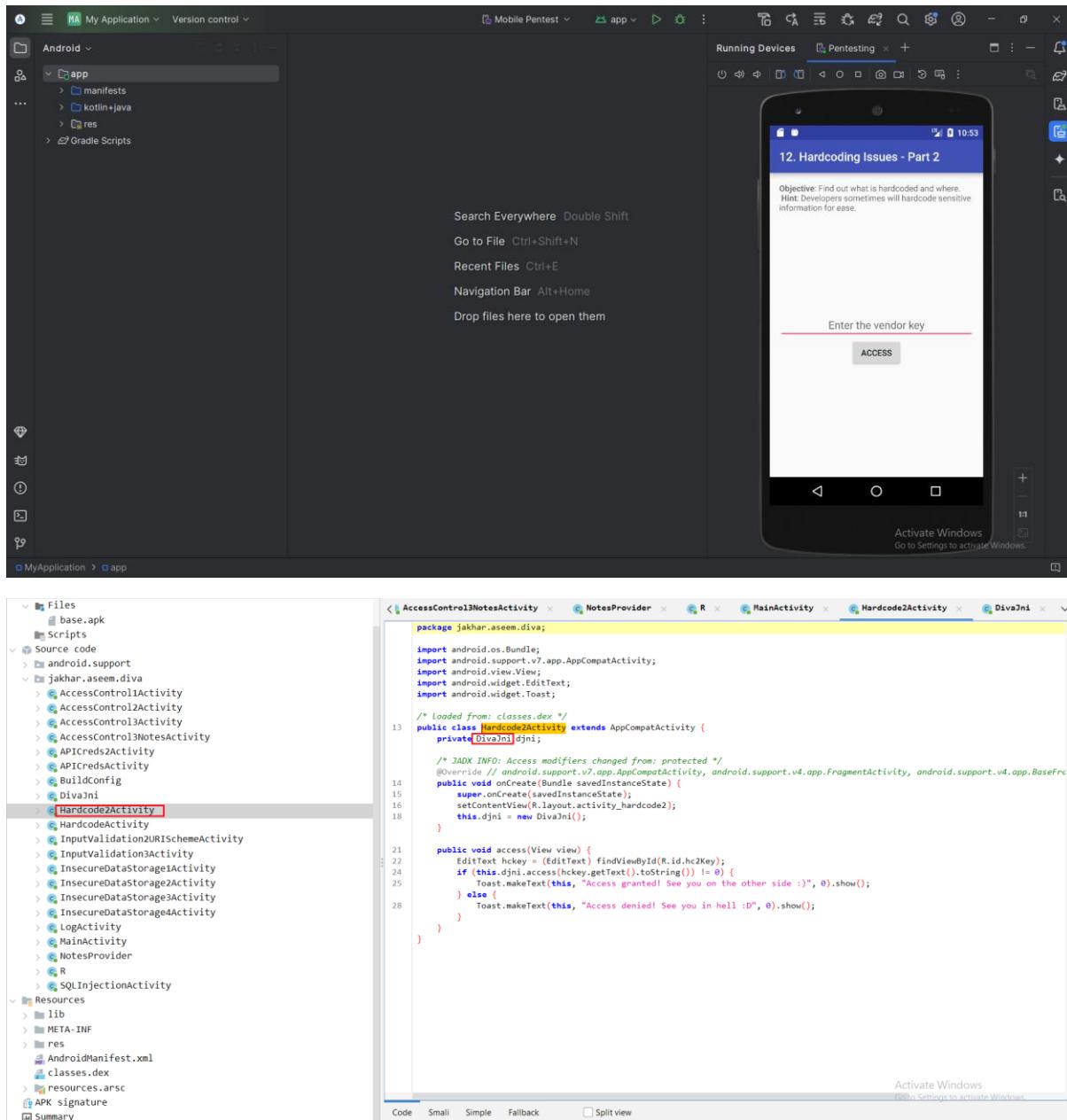
```

root@generic_x86:/ # content query --uri content://jakhar.aseem.diva.provider.notesprovider/notes
Row: 0 _id=5, title=Exercise, note=Alternate days running
Row: 1 _id=4, title=Expense, note=Spent too much on home theater
Row: 2 _id=6, title=Weekend, note=b333333333333333r
Row: 3 _id=3, title=holiday, note=Either Goa or Amsterdam
Row: 4 _id=2, title=home, note=Buy toys for baby, Order dinner
Row: 5 _id=1, title=office, note=10 Meetings. 5 Calls. Lunch with CEO
root@generic_x86:/ #

```

## 12. Hardcoding Issues \_ Part-2

**Screenshot-1:** The 1<sup>st</sup> activity **Hardcode2Activity** is responsible for entering the vendor key. We are getting 2<sup>nd</sup> activity name ‘DivaJni’ from the source code of the **Hardcode2Activity**.



**Screenshot-2:** The source code of the 2<sup>nd</sup> activity `DivAjni` provide the string value `soName = "divajjni"`. We navigate the file `libdivajjni.so` and found as compiled content.

package jakhar.asseem.diva;

/\* Loaded from: classes.dex \*/

public class Divajni {

    private static final String soName = "divajni";

    public native int access(String str);

    public native int initiateLaunchSequence(String str);

    static {

        System.loadLibrary(soName);

    }

}

1 00 45 4C 00 01 01 01 00 00 00 00 00 00 00 00

2 02 00 00 00 00 01 00 00 00 00 00 00 34 00 00

3 EF BF BD 10 00 00 00 00 00 00 34 00 20 00 07

4 28 00 15 00 14 00 06 00 00 00 00 34 00 00 00 04

5 00 00 34 00 00 00 EF BF BD 00 00 EF BF BD 00

6 00 00 04 00 00 00 00 00 00 00 01 00 00 00 00

7 00 00 00 00 00 00 00 00 00 00 EF BF BD 00 00

8 EF BF BD 00 00 00 05 00 00 00 00 10 00 00 01 00

9 00 00 EF BF BD 0E 00 00 00 EF BF BD 1E 00 00 EF BF

10 1E 00 00 00 24 00 00 00 00 24 00 00 00 00 00

11 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

12 BD 1E 00 00 EF BF BD 1E 00 00 EF BF BD 00 00

13 EF BD 00 00 00 06 00 00 00 00 04 00 00 00 50 EF

14 BF BD 74 64 5C 06 00 00 00 5C 06 00 00 5C 06 00

15 54 00 00 00 54 00 00 00 00 00 00 00 00 00 00 00

16 51 EF BF BD 74 64 00 00 00 00 00 00 00 00 00 00

17 00 00 00 00 00 00 00 00 00 00 07 00 00 00 00 00

18 00 00 00 52 EF BF BD 74 64 EF BF BD 00 00 00 EF BF

19 5C 1E 00 00 00 EF BF BD 1E 00 00 28 01 00 00 20 01

20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

21 00 00 00 00 00 00 00 00 00 00 01 00 00 00 00 00

22 00 00 00 00 00 00 00 00 00 00 10 00 00 00 00 00

23 00 00 00 00 00 00 00 00 00 00 12 00 00 00 10 00

24 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

25 00 00 00 4E 00 00 00 00 12 00 07 00 54 00 00 EF BF

26 BD 00 00 00 6F 00 00 00 12 00 07 00 EF BF BD 00

27 00 00 00 00 00 00 00 00 00 00 12 00 00 00 EF BF

28 BD 00 00 00 00 EF BF BD 04 00 00 00 06 00 00 00 12 00

29 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

30 10 00 00 00 00 EF BF BD 0F 00 00 00 00 00 00 00 04 20

31 00 00 00 00 00 00 00 10 00 EF BF BD 0F 00 00 00 EF BF

32 BD 00 00 00 04 28 00 00 00 00 00 00 10 00 EF BE

33 BD EF BF BD 00 5F 63 78 61 5F 66 69 66 61 6C

34 69 7A 65 00 5F 5F 63 78 61 5F 61 74 65 78 69 74

35 00 5F 5F 73 74 61 63 68 63 68 5F 66 5F 61 69

36 6C 00 4A 61 76 61 5F 6A 61 68 61 72 5F 61 73

37 65 65 6D 5F 64 69 76 61 5F 44 69 76 61 4A 66

38 6B 65 6D 65 73 73 00 44 79 61 5F 44 69 61 6B

39 61 62 63 65 72 72 00 44 79 61 5F 44 69 61 64

40 69 76 61 64 6A 69 5F 69 66 69 74 61 74 65 4C

41 61 75 63 68 53 65 71 75 65 63 65 00 73 74

42 72 63 70 79 00 4A 4E 49 5F 4F 6E 4C 6F 61 64 00

43 5F 65 64 61 74 61 00 5F 5F 62 73 73 5F 73 74 61

**Screenshot-3:** Decompile the **base.apk** file by using below command and found **base** directory.

```
apktool d .\base.apk
```

Now navigate the below path:

```
base\lib\x86\libdivajni.so
```

Now see the all vendor key value from **libdivajni.so** by using bellow command:

```
strings .\libdivajni.so
```

```
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> ls

Directory: D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting

Mode                LastWriteTime         Length Name
----                -----          ---- 
-a----   29-07-2024 10:35 PM      1502294 base.apk
-a----   29-07-2024 11:08 PM        1124 base.apk.jadx
-a----   23-07-2024 10:54 PM      528094 Diva Scoreboard Mobile PT.pdf
-a----   23-07-2024 10:56 PM      152285 Diva Static Report Mobile PT.pdf
-a----   03-08-2024 05:48 PM        16384 ids2
-a----   18-08-2024 10:54 AM      8673261 StaticAnalysis.docx

PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> apktool d .\base.apk
I: Using Apktool 2.9.3 on base.apk
I: Loading resource table...
I: Decoding file-resources...
I: Loading resource table from file: C:\Users\Kausik\AppData\Local\apktool\framework\1.apk
I: Decoding values /* XMLS...
I: Decoding AndroidManifest.xml with resources...
I: Regular manifest package...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
Press any key to continue . .
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> ls

Directory: D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting

Mode                LastWriteTime         Length Name
----                -----          ---- 
d----   18-08-2024 11:03 AM           base
-a----   29-07-2024 10:35 PM      1502294 base.apk
-a----   29-07-2024 11:08 PM        1124 base.apk.jadx
-a----   23-07-2024 10:54 PM      528094 Diva Scoreboard Mobile PT.pdf
-a----   23-07-2024 10:56 PM      152285 Diva Static Report Mobile PT.pdf
-a----   03-08-2024 05:48 PM        16384 ids2
-a----   18-08-2024 10:54 AM      8673261 StaticAnalysis.docx
```

```

PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> cd ..\base
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting\base> ls
Directory: D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting\base

Mode                LastWriteTime         Length Name
----                -----        ----
d-----       18-08-2024 11:03 AM          1 lib
d-----       18-08-2024 11:03 AM      100 original
d-----       18-08-2024 11:03 AM        1 res
d-----       18-08-2024 11:03 AM        1 small
-a---- 18-08-2024 11:03 AM      3390 AndroidManifest.xml
-a---- 18-08-2024 11:03 AM      400 apktool.yml

PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> cd ..\lib
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting\lib> ls
Directory: D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting\lib

Mode                LastWriteTime         Length Name
----                -----        ----
d-----       18-08-2024 11:03 AM        1 arm64-v8a
d-----       18-08-2024 11:03 AM        1 armeabi
d-----       18-08-2024 11:03 AM        1 armeabi-v7a
d-----       18-08-2024 11:03 AM        1 mips
d-----       18-08-2024 11:03 AM        1 mips64
d-----       18-08-2024 11:03 AM          1 x86
d-----       18-08-2024 11:03 AM        1 x86_64

PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting> cd x86
PS D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting\lib\x86> ls
Directory: D:\Security ALL\Mobile Application Penetration Testing\DIVA APK Security Assessment\Divapentesting\lib\x86

Mode                LastWriteTime         Length Name
----                -----        ----
-a---- 18-08-2024 11:03 AM          5164 libdivajni.so

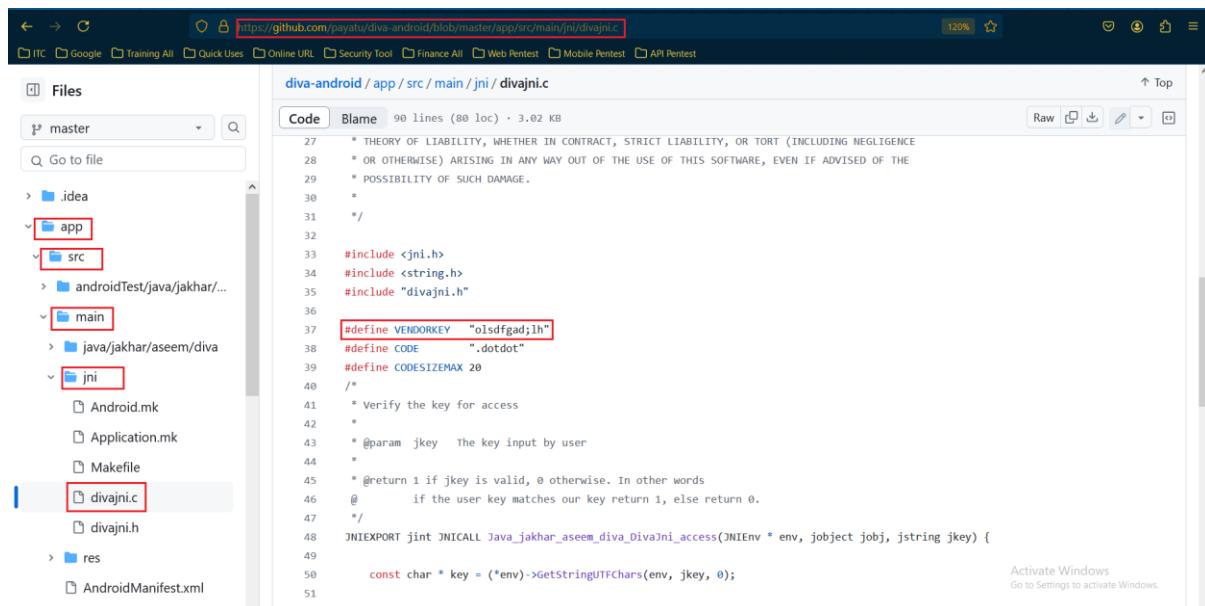
```

```

PS D:\divapentesting\lib\x86> strings .\libdivajni.so
__cxa_finalize
__cxa_atexit
__stack_chk_fail
Java_jakhar_aseem_diva_DivaJni_access
Java_jakhar_aseem_diva_DivaJni_initiateLaunchSequence
strcpy
JNI_OnLoad
edata
_bss_start
_end
libstdc++.so
libm.so
libc.so
libdl.so
libdivajni.so
d$0[
olsdfgad;lh
.dotdot
;*2$"
GCC: (GNU) 4.8
gold 1.11
.shstrtab
.dynsym
.dynstr
.hash
.rel.dyn
.rel.plt
.text

```

**Screenshot-4:** Either brute force all Vendor Key value or recon the same and then use it.



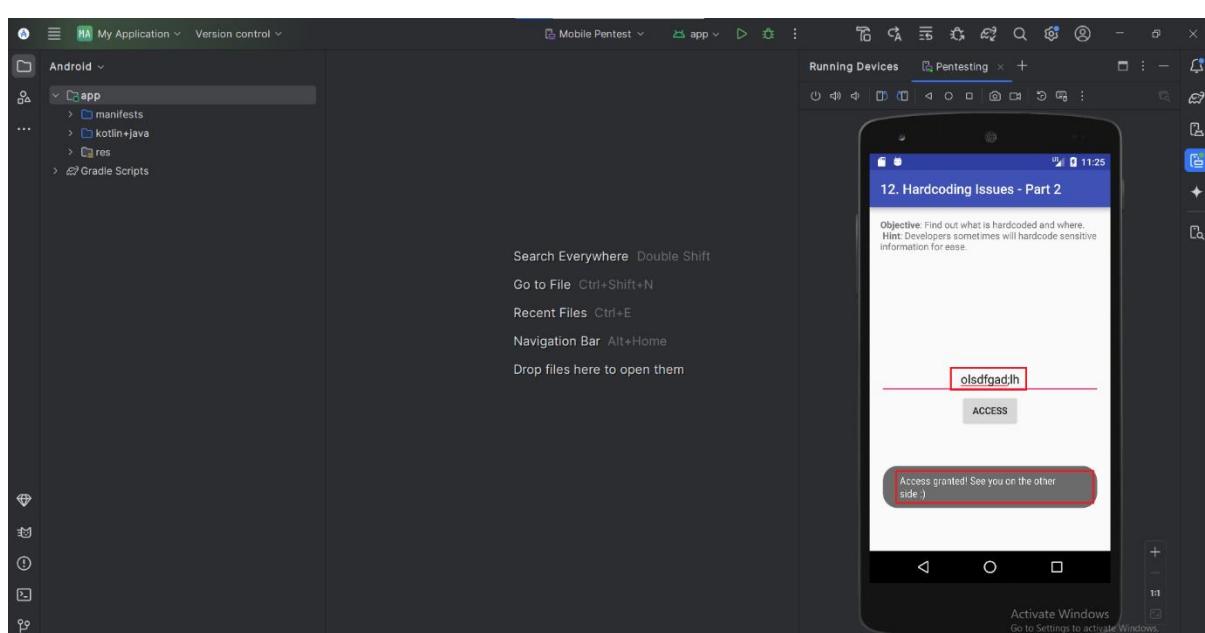
The screenshot shows a GitHub file viewer for 'divjni.c'. The file path is 'div-android / app / src / main / jni / divjni.c'. The code contains a hard-coded vendor key '#define VENDORKEY "olsdfgad;lh"'. The file is 90 lines long and 3.02 KB. A red box highlights the 'divjni.c' file in the sidebar and the vendor key line in the code editor.

```

27 * THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE
28 * OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
29 * POSSIBILITY OF SUCH DAMAGE.
30 */
31
32 #include <jni.h>
33 #include <string.h>
34 #include "divjni.h"
35
36 #define VENDORKEY "olsdfgad;lh"
37 #define CODE ".dotdot"
38 #define CODESIZEMAX 20
39
40 /*
41 * Verify the key for access
42 *
43 * @param jkey The key input by user
44 *
45 * @return 1 if jkey is valid, 0 otherwise. In other words
46 * @ 1 if the user key matches our key return 1, else return 0.
47 */
48 JNIEXPORT jint JNICALL Java_jakhar_aseem_diva_DivaJni_access(JNIEnv * env, jobject job, jstring jkey) {
49     const char * key = (*env)->GetStringUTFChars(env, jkey, 0);
50
51     return ...
52 }

```

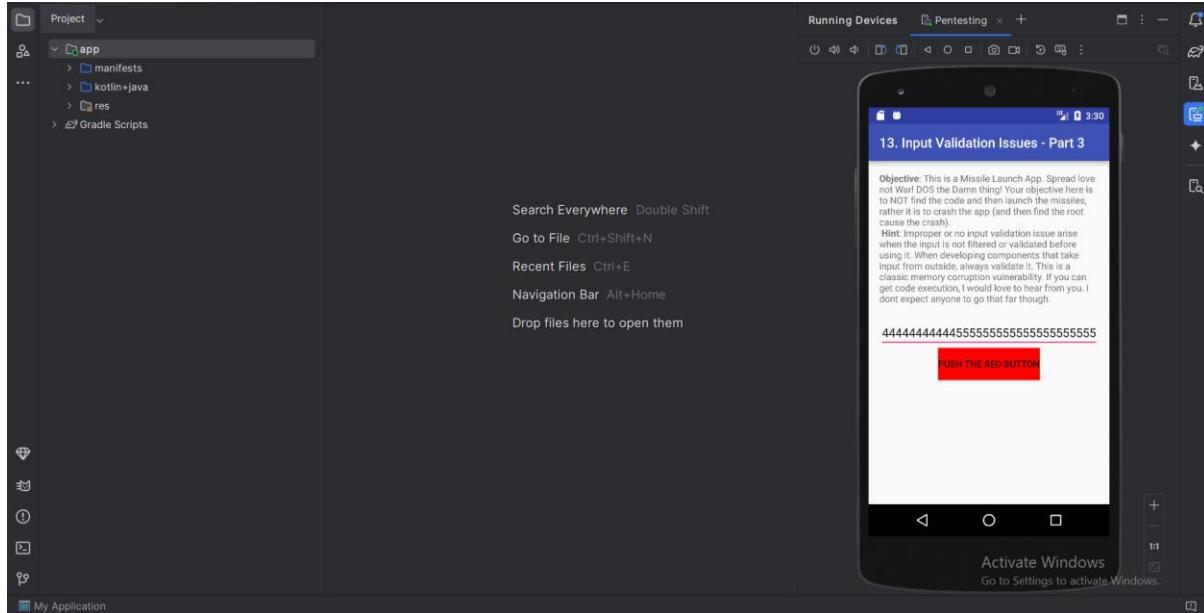
Activate Windows  
Go to Settings to activate Windows.



The screenshot shows the Android Studio interface with the project 'My Application'. The file structure on the left includes 'app', 'manifests', 'kotlin+java', and 'res'. On the right, a 'Running Devices' window shows a virtual device with the title '12. Hardcoding Issues - Part 2'. The device screen displays the text 'olsdfgad;lh' in a red-bordered box and an 'ACCESS' button below it. A red box highlights the text 'olsdfgad;lh' on the device screen.

## 13. Input Validation Issues \_ Part-3

**Screenshot-1:** Here it could observe that improper input validation as there have no limit of input value. We are entering a large no of input and hit the red button.



**Screenshot-2:** After entering the large no of input the DIVA app has stopped.

