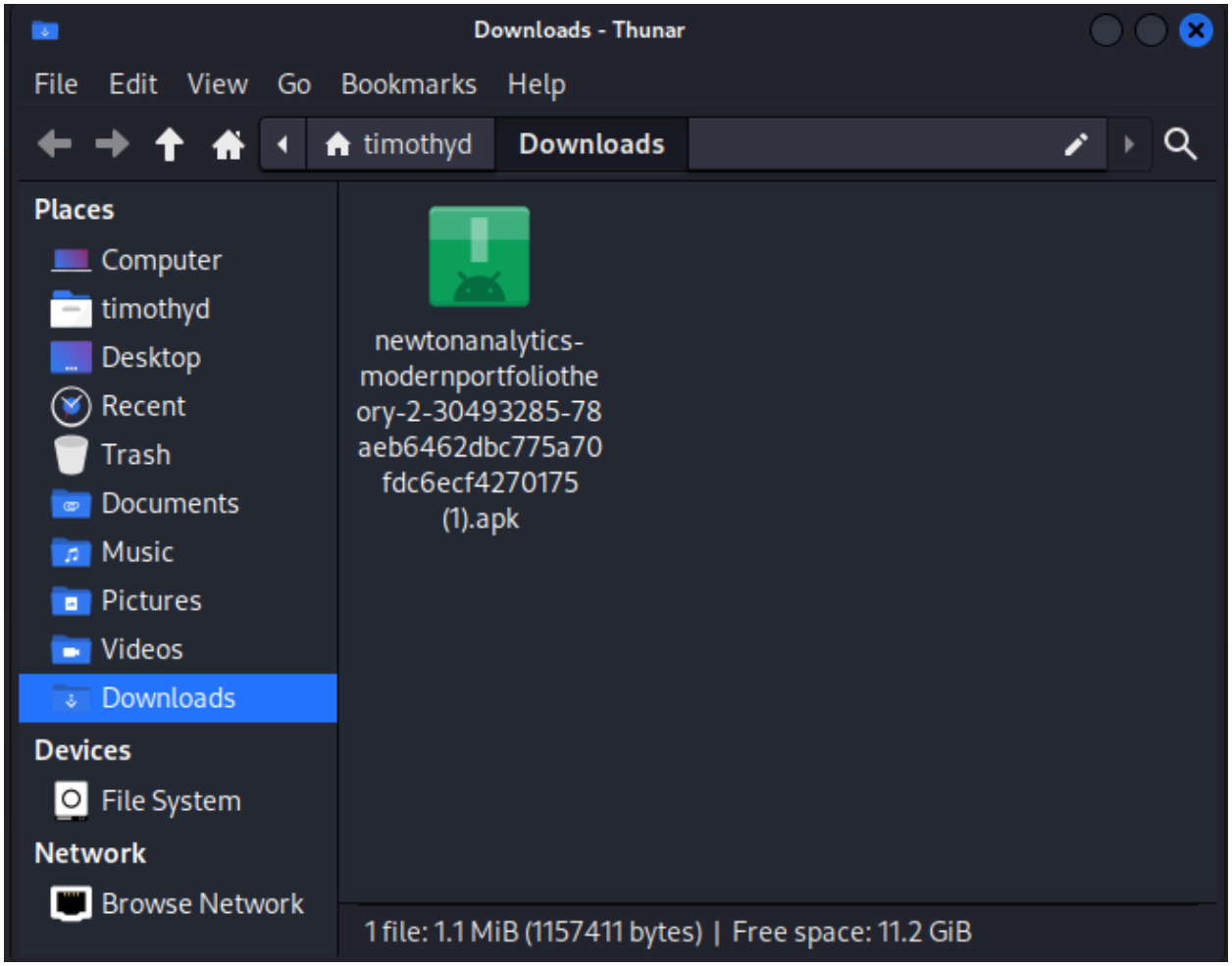
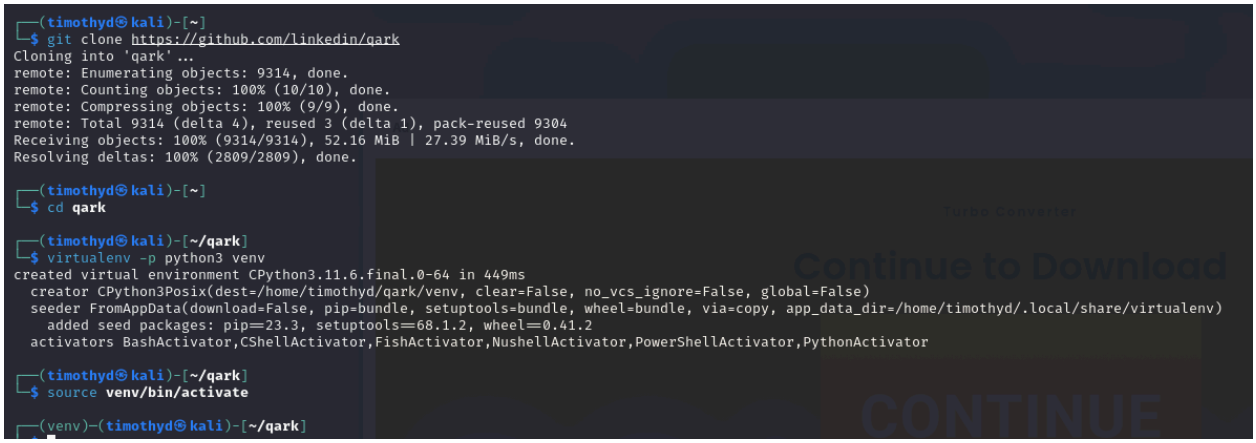
Task 1:



In this step, I have downloaded the APK file.



In this step, I have cloned the git repo and amd moving to the qark folder. Then I am downloading the python virtual environment and activating it.

```
┌──(venv)─(timothyd㉿kali)-[~/qark]
└─$ sudo qark --apk /home/timothyd/Downloads/newtonanalytics-modernportfoliotheory-2-30493285-78aeb6462dbc775a70fdc6ecf4270175.apk
Decompiling ...
dex2jar /home/timothyd/qark/build/qark/classes.dex → /home/timothyd/qark/build/qark/newtonanalytics-modernportfoliotheory-2-30493285-78aeb6462dbc775a70fdc6ecf4270175.jar
I: Using Apktool 2.3.1 on newtonanalytics-modernportfoliotheory-2-30493285-78aeb6462dbc775a70fdc6ecf4270175.apk
I: Loading resource table ...
I: Decoding AndroidManifest.xml with resources ...
S: WARNING: Could not write to (/root/.local/share/apktool/framework), using /tmp instead ...
S: Please be aware this is a volatile directory and frameworks could go missing, please utilize --frame-path if the default storage directory is unavailable
I: Loading resource table from file: /tmp/1.apk
I: Decoding file-resources ...
I: Decoding values */* XMLs ...
I: Copying raw classes.dex file ...
I: Copying assets and libs ...
I: Copying unknown files ...
I: Copying original files ...
Processing /home/timothyd/qark/build/qark/newtonanalytics-modernportfoliotheory-2-30493285-78aeb6462dbc775a70fdc6ecf4270175.jar (use silent to silence)
Processing android.support.v4.accessibilityservice.AccessibilityServiceInfoCompat
INFO:  Decompiling class android/support/v4/accessibilityservice/AccessibilityServiceInfoCompat
Processing android.support.v4.accessibilityservice.AccessibilityServiceInfoCompatIcs
INFO:   ... done
INFO:  Decompiling class android/support/v4/accessibilityservice/AccessibilityServiceInfoCompatIcs
INFO:   ... done
INFO:  Decompiling class android/support/v4/accessibilityservice/AccessibilityServiceInfoCompatJellyBeanMr2
INFO:   ... done
INFO:  Decompiling class android/support/v4/app/ActionBarDrawerToggleHoneycomb
Processing android.support.v4.accessibilityservice.AccessibilityServiceInfoCompatJellyBeanMr2
Processing android.support.v4.app.ActionBarDrawerToggleHoneycomb
INFO:   ... done
INFO:  Decompiling class android/support/v4/app/ActivityCompatHoneycomb
INFO:   ... done
INFO:  Decompiling class android/support/v4/app/ActivityCompatJB
INFO:   ... done
INFO:  Decompiling class android/support/v4/app/ActivityOptionsCompat
```

In this step, I am running qark while targeting the apk that we downloaded earlier. The path of the report is /usr/local/lib/python3.11/dist-packages/qark-4.0.0-py3.11.egg/qark/report/report.html .

```
┌──(venv)─(timothyd㉿kali)-[~/qark]
└─$ cd build/qark

┌──(venv)─(timothyd㉿kali)-[~/qark/build/qark]
└─$ ls -la
total 1524
drwxr-xr-x  7 root root    4096 May  8 10:56 .
drwxr-xr-x  5 root root    4096 May  8 10:56 ..
-rw-r--r--  1 root root    1365 May  8 10:56 AndroidManifest.xml
drwxr-xr-x  2 root root    4096 May  8 10:56 META-INF
drwxr-xr-x  4 root root    4096 May  8 10:56 cfr
-rw-r--r--  1 root root  652804 May  8 10:56 classes.dex
drwxr-xr-x  2 root root    4096 May  8 10:56 fernflower
-rw-r--r--  1 root root  463307 May  8 10:56 newtonanalytics-modernportfoliotheory-2-30493285-78aeb6462dbc775a70fdc6ecf4270175.jar
drwxr-xr-x  4 root root    4096 May  8 10:56 procyon
drwxr-xr-x 10 root root    4096 May  8 10:56 res
-rw-r--r--  1 root root  404672 May  8 10:56 resources.arsc

┌──(venv)─(timothyd㉿kali)-[~/qark/build/qark]
```

In this step, I am in the build/qark directory and am listing all of the outputs.

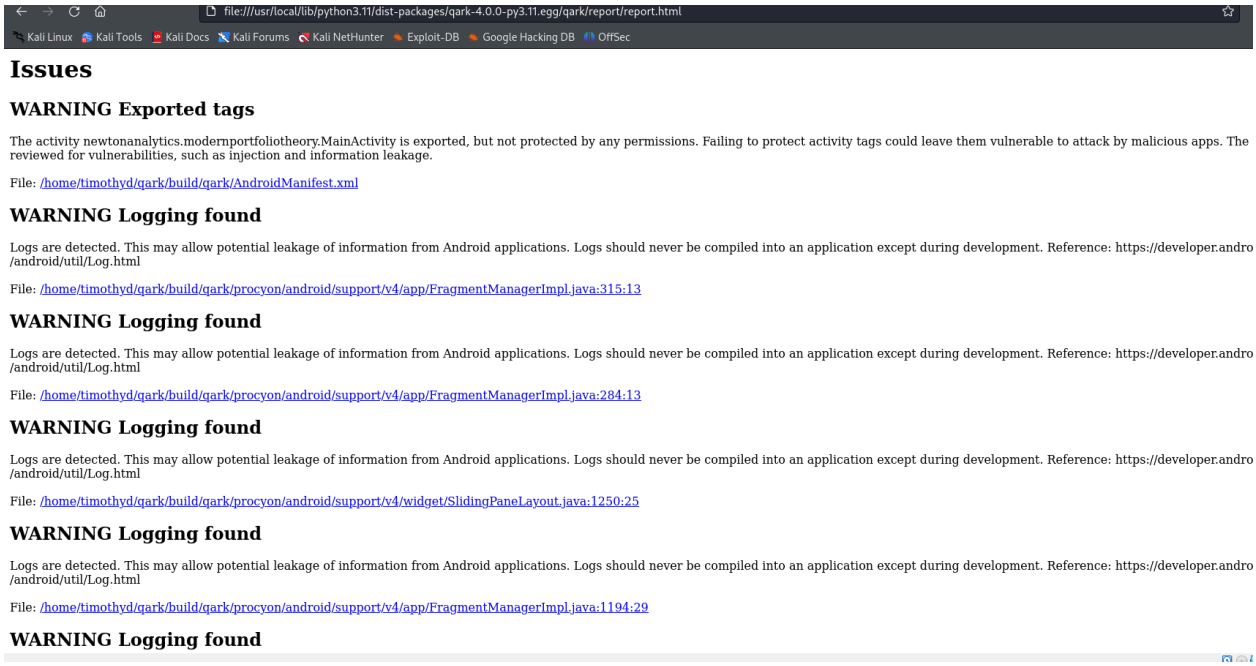
```
┌──(venv)─(timothyd㉿kali)-[~/qark/build/qark]
└─$ cat AndroidManifest.xml
<?xml version="1.0" encoding="utf-8"?>
<manifest android:versionCode="2" android:versionName="1.1" package="newtonanalytics.modernportfoliotheory" platformBuildVersionCode="18" platformBuildVersionName="4.3.1-1425645"
  xmlns:android="http://schemas.android.com/apk/res/android">
    <uses-sdk android:minSdkVersion="14" android:targetSdkVersion="18" />
    <uses-permission android:name="android.permission.INTERNET" />
    <application android:theme="@style/AppTheme" android:label="@string/app_name" android:icon="@drawable/ic_launcher" android:allowBackup="true">
        <activity android:label="@string/app_name" android:name="newtonanalytics.modernportfoliotheory.MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:label="@string/title_activity_display_contact" android:name="newtonanalytics.modernportfoliotheory.DisplayContact" />
        <activity android:label="Your Settings" android:name="newtonanalytics.modernportfoliotheory.UserSettingActivity" />
        <activity android:label="@string/title_activity_run" android:name="newtonanalytics.modernportfoliotheory.Run" android:screenOrientation="portrait" />
    </application>
</manifest>
```

Here I am displaying the contents of AdroidManifest.xml

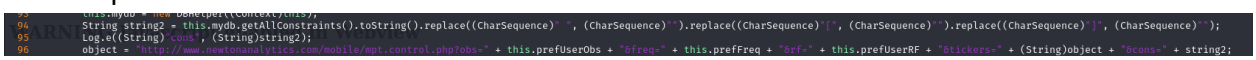
```
┌──(venv)─(timothyd㉿kali)-[~/…/qark/procyon/newtonanalytics/modernportfoliotheory]
└─$ cat DBHelper.java
//
// Decompiled by Procyon v1.0-SNAPSHOT
//

package newtonanalytics.modernportfoliotheory;

import android.database.DatabaseUtils;
import android.database.sqlite.SQLiteDatabase;
import android.content.ContentValues;
import android.database.Cursor;
import java.util.ArrayList;
import android.database.sqlite.SQLiteDatabase$CursorFactory;
import android.content.Context;
import java.util.HashMap;
import android.database.sqlite.SQLiteOpenHelper;

public class DBHelper extends SQLiteOpenHelper
{
    public static final String CONTACTS_COLUMN_CONSTRAIN = "constrain";
    public static final String CONTACTS_COLUMN_ID = "id";
    public static final String CONTACTS_COLUMN_NAME = "name";
    public static final String CONTACTS_TABLE_NAME = "contacts";
    public static final String DATABASE_NAME = "MyDBName.db";
    private HashMap hp;

    public DBHelper(final Context context) {
        super(context, "MyDBName.db", (SQLiteDatabase$CursorFactory)null, 1);
    }

    public boolean checkTicker(final String str) {
        return this.getReadableDatabase().rawQuery("select * from contacts where name='" + str + "'", (String[])null).getCount() > 0;
    }

    public Integer deleteContact(final String s) {
        return this.getWritableDatabase().delete("contacts", "name = ? ", new String[] { s });
    }

    public ArrayList getAllConstraints() {
        final ArrayList<String> list = new ArrayList<String>();
        final Cursor rawQuery = this.getReadableDatabase().rawQuery("select * from contacts", (String[])null);
        rawQuery.moveToFirst();
        while (!rawQuery.isAfterLast()) {
            list.add(rawQuery.getString(rawQuery.getColumnIndex("constrain")));
            rawQuery.moveToNext();
        }
```

In this step, I have navigated to the procyon/newtonanalytics/modernportfoliotheory directory and am outputting the contents of the DBHelper.java file.

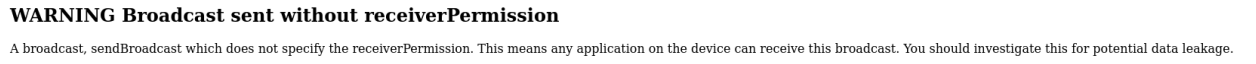
In the checkTicker() and the getData() function, there are sql queries being directly concatenated with user input. This is where the app could be vulnerable to sql injection. To mitigate this vulnerability, we would implement input validation and parameterized queries. This includes creating prepared statements to prevent an attacker from being able to sneak in malicious sql commands..



Here I have opened the report in my browser.



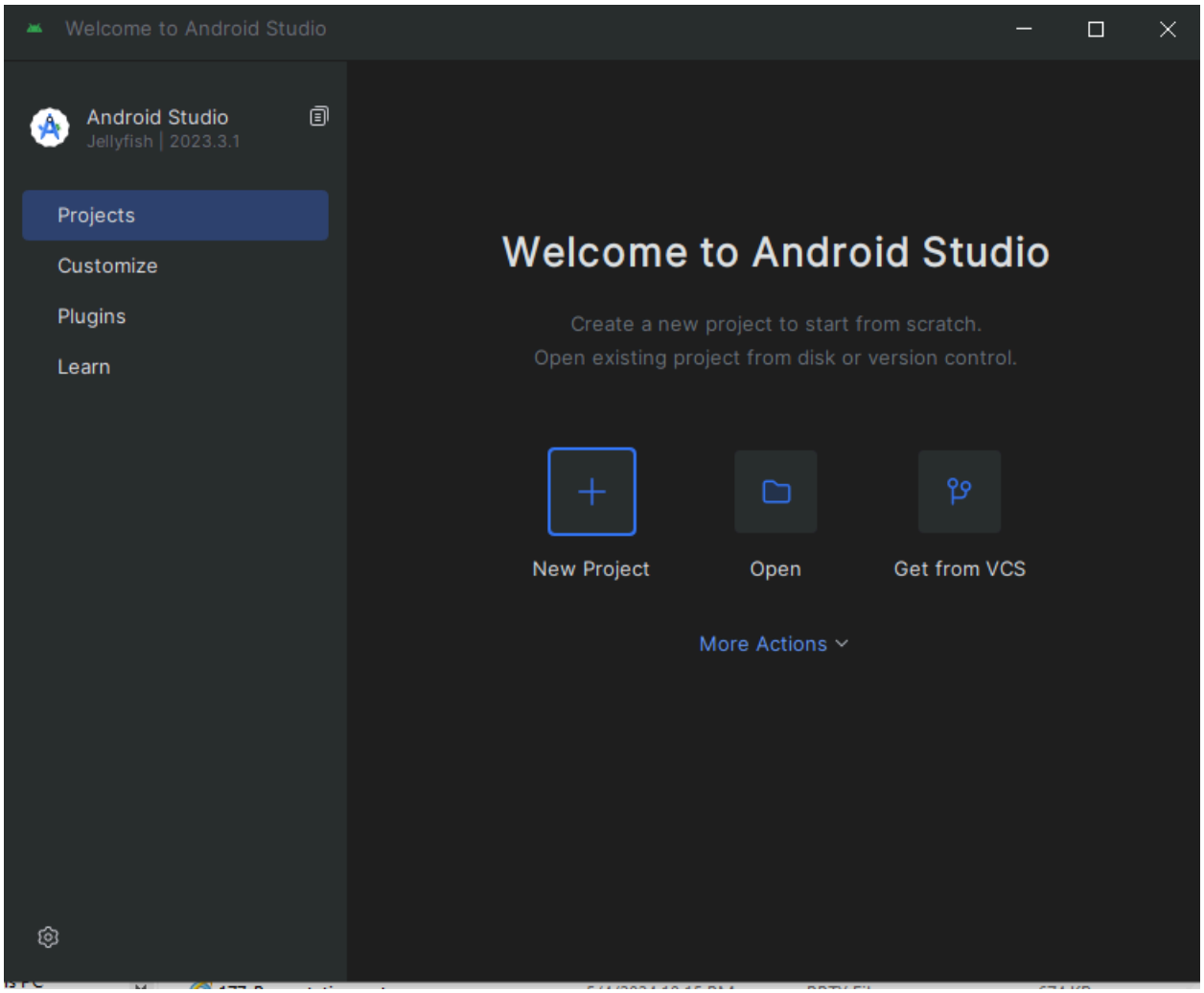
One vulnerability that I have found is “Hardcoded HTTP url found”. The image below is where it is located in the source code. It is located in the Run.java file. The severity of this vulnerability is pretty high due to the fact that the data that is being transmitted between the app and server is not encrypted. An attacker could be able to read and manipulate the data that is being transferred. A way to mitigate this is to enforce HSTS, which is HTTP String Transport Security. This will force the web browser to only communicate with the server over HTTPS, even if the initial request was made over HTTP.



Another vulnerability that I found was “Broadcast sent without receiverPermission”. The image below is where it is located in the source code. It is located in the LocalBroadcastManager.java file. This vulnerability means that a broadcast is being sent without specifying a receiver permission. Without specifying receiver permissions, any application installed on the device can intercept and receive the broadcast. The severity of this vulnerability depends on how sensitive the information that the broadcast contains is. If the information is non sensitive then the severity is low, but if the information is very sensitive then the severity is high. The impact of this vulnerability is that unauthorized apps can intercept the broadcast and view the information that it holds. The way to mitigate this is to implement receiver permissions. This will ensure that broadcasts that hold sensitive information are only accessible to trusted receivers with the necessary permissions. This will mitigate the risk of unauthorized access to these broadcasts being sent.

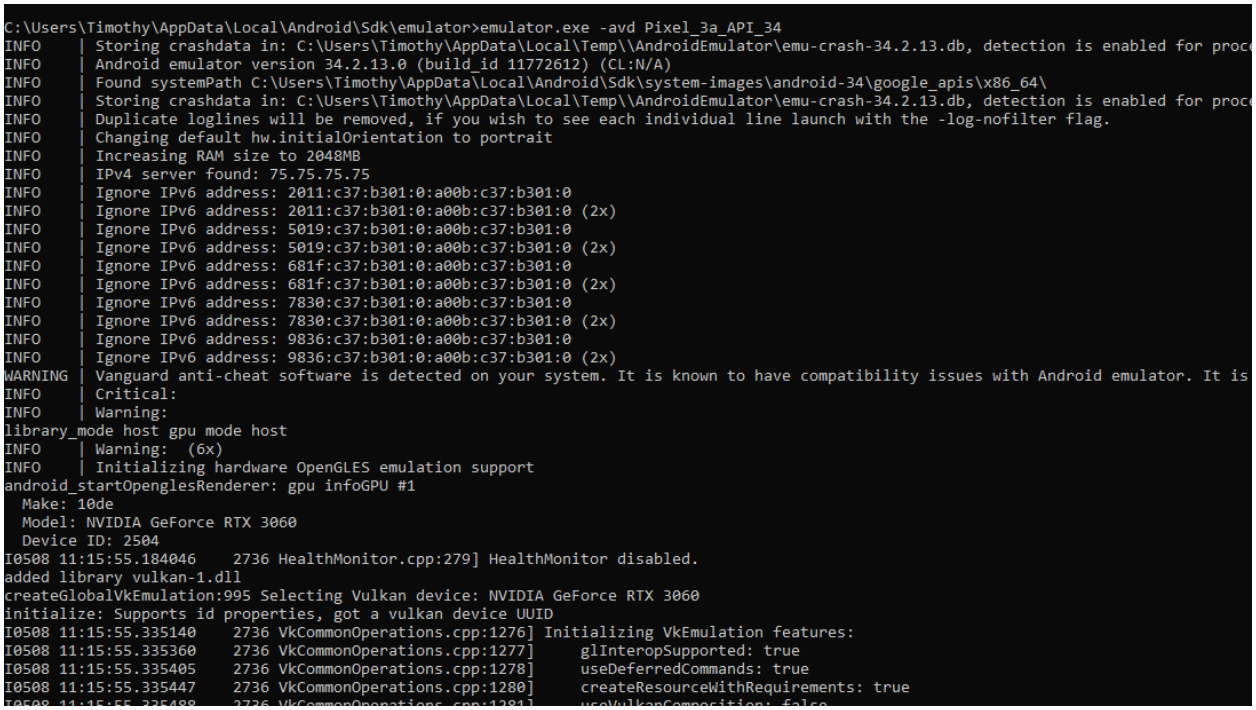
```
368     public void sendBroadcastSync(final Intent intent) {
369         if (this.sendBroadcast(intent)) {
370             this.executePendingBroadcasts();
371         }
372     }
```

Task 2:

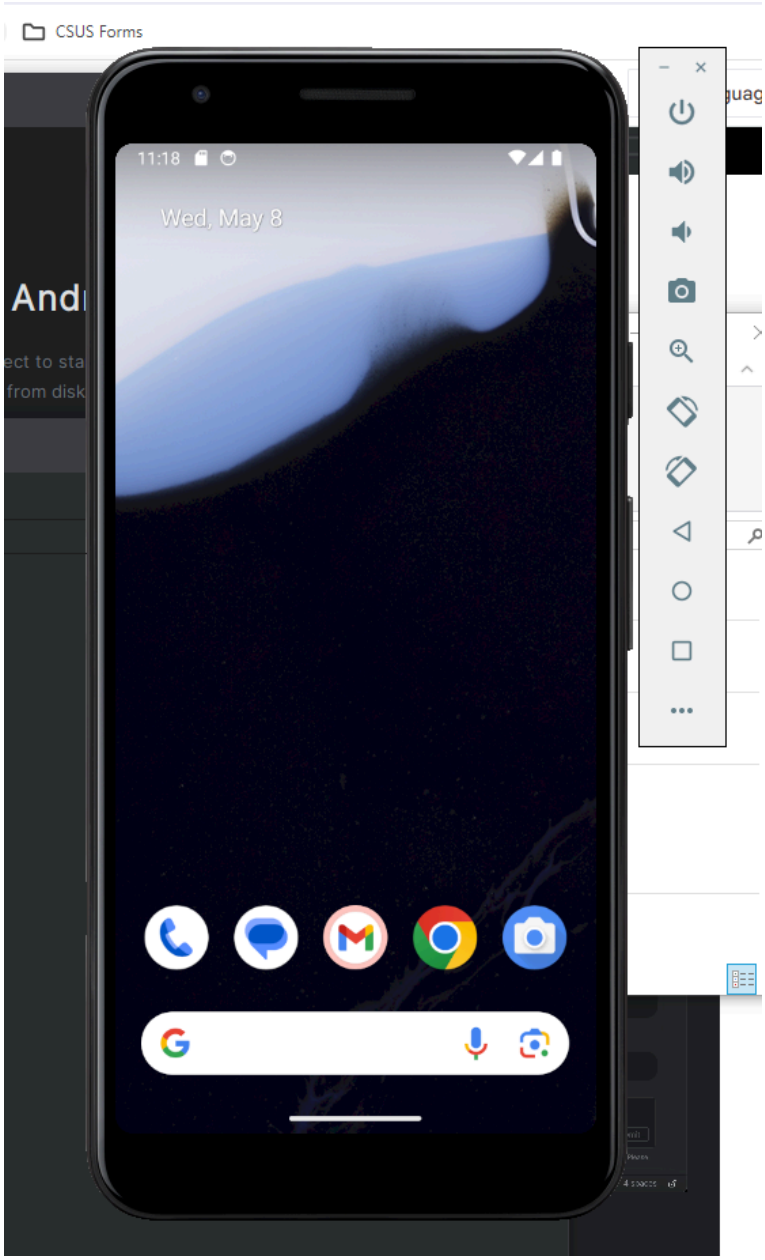


In this step, I have downloaded android studios and the necessary SDK's.

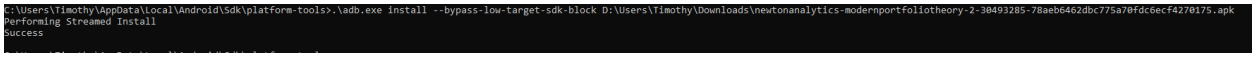
```
C:\Users\Timothy\AppData\Local\Android\Sdk\emulator>emulator.exe -list-avds
INFO    | Storing crashdata in: C:\Users\Timothy\AppData\Local\Temp\\AndroidEmulator\emu-crash-34.2.13.db, detection is enabled for process: 13088
Pixel_3a_API_34
```



In this step, I have moved to the Android SDK emulator folder. I then list the AVD. Then I am starting the device emulator.



Here we can see the phone emulator.



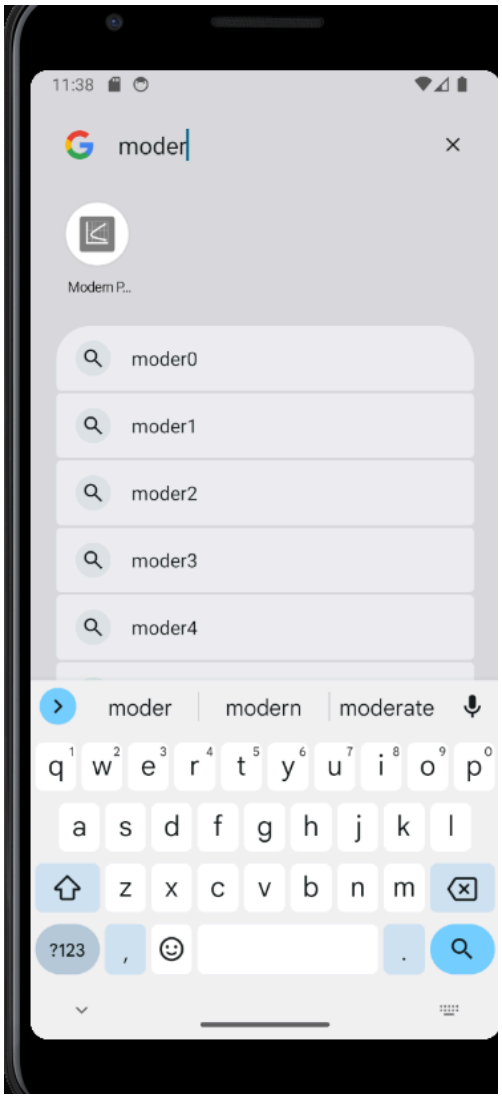
In this step, I downloaded the APK and then moved to the AppData\Local\Android\Sdk\platform-tools directory. Then I am installing the APK.

```
C:\Users\Timothy\AppData\Local\Android\Sdk\platform-tools>.\adb.exe shell
emu64xa:/ $ pm list packages
package:com.android.systemui.auto_generated_rro_vendor__
package:com.google.android.providers.media.module
package:com.google.android.overlay.permissioncontroller
package:com.google.android.overlay.googlewebview
package:com.android.calllogbackup
package:com.android.carrierconfig.auto_generated_rro_vendor__
package:com.android.systemui.accessibility.accessibilitymenu
package:com.android.internal.emulation.pixel_3_xl
package:com.android.providers.contacts
package:com.android.internal.emulation.pixel_4a
package:com.android.dreams.basic
package:com.android.companiondevicemanager
package:com.android.cts.priv.ctsshim
package:com.google.android.calendar
package:com.google.android.networkstack.tethering.emulator
package:com.google.android.contacts
package:com.android.mms.service
package:com.google.android.cellbroadcastreceiver
package:com.android.providers.downloads
package:com.android.bluetoothmidiservice
package:com.android.credentialmanager
package:com.google.android.printservice.recommendation
package:com.google.android.captiveportallogin
package:com.android.storagemanager.auto_generated_rro_product__
package:com.google.android.networkstack
package:com.google.android.overlay.googleconfig
package:com.android.keychain
package:com.google.android.tag
package:com.android.internal.emulation.pixel_2_xl
package:com.android.internal.emulation.pixel_7a
package:android.auto_generated_rro_vendor__
package:com.google.android.apps.wellbeing
package:com.android.nfc.auto_generated_rro_product__
package:com.android.virtualmachine.res
package:com.android.emulator.multidisplay
package:com.android.shell
package:com.google.android.adservices.api
package:com.google.android.wifi.dialog
package:com.google.android.apps.wallpaper.nexus
package:com.android.inputdevices
package:com.google.android.ondevicepersonalization.services
```

In this step, I have entered the debugger shell and then listed the packages installed.

```
package:com.android.theme.font.notoserifsource
package:com.android.traceur.auto_generated_rro_product__
package:com.google.android.health.connect.backuprestore
package:com.android.systemui.emulation.pixel_8_pro
package:com.google.android.settings.intelligence
package:newtonanalytics.modernportfoliotheory
package:com.android.systemui.emulation.pixel_3
package:com.android.systemui
package:com.android.wallpapercropper
package:com.android.internal.emulation.pixel_4
package:com.android.systemui.emulation.pixel_7
package:com.android.internal.emulation.pixel_fold
package:com.android.internal.emulation.pixel_5
package:com.android.systemui.emulation.pixel_6
package:com.android.providers.contacts.auto_generated_rro_product__
```

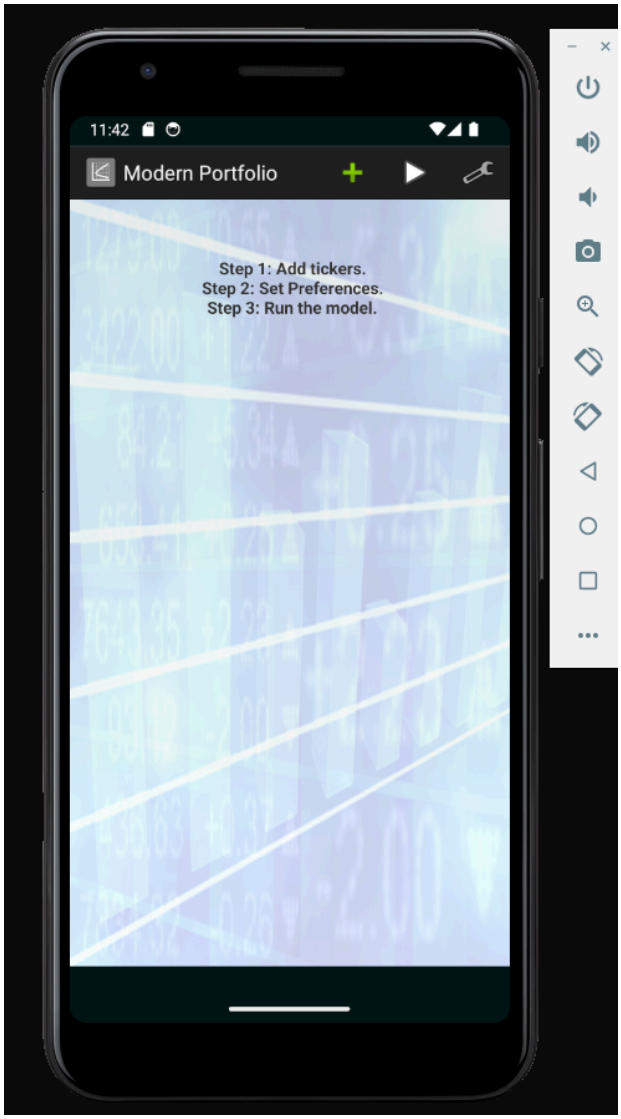
We can see that newtonanalytics is installed.



Here we can see that modern portfolio is installed onto the emulator.

```
emu64xa:/ $ am start -n newtonanalytics.modernportfoliotheory/.MainActivity
Starting: Intent { cmp=newtonanalytics.modernportfoliotheory/.MainActivity }
emu64xa:/ $
```

In this step, I have sent an intent from the debugger.



Here we can see that modern portfolio has been launched. I went through the permissions screen to get to this page.