

Lab – 1

Name: Ishan Aakash Patel

Student ID: 146151238

Course: CYT-230

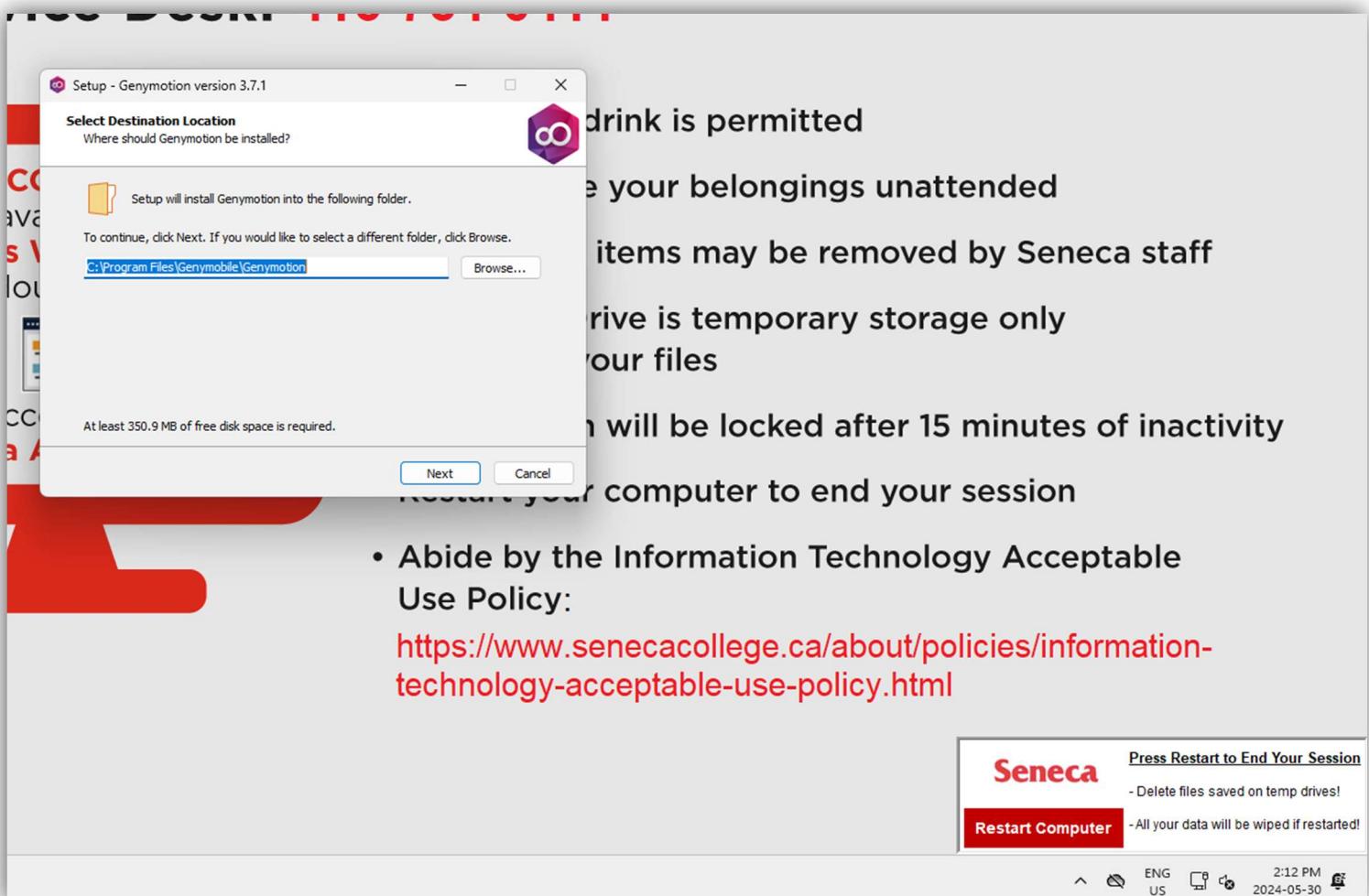
Mobile app Development and Mobile Emulators

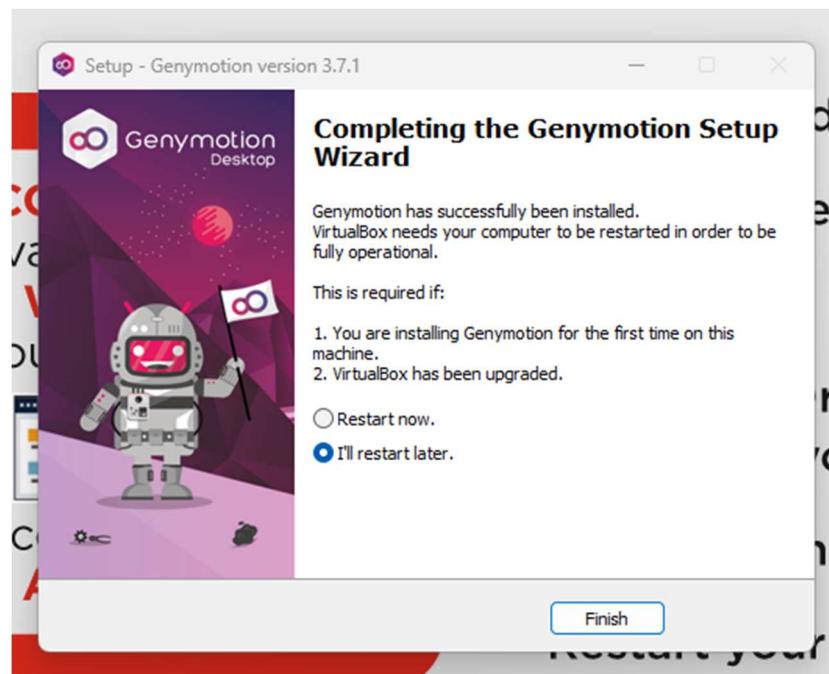
Part – 1

Installing Android Emulators

1) Genymotion

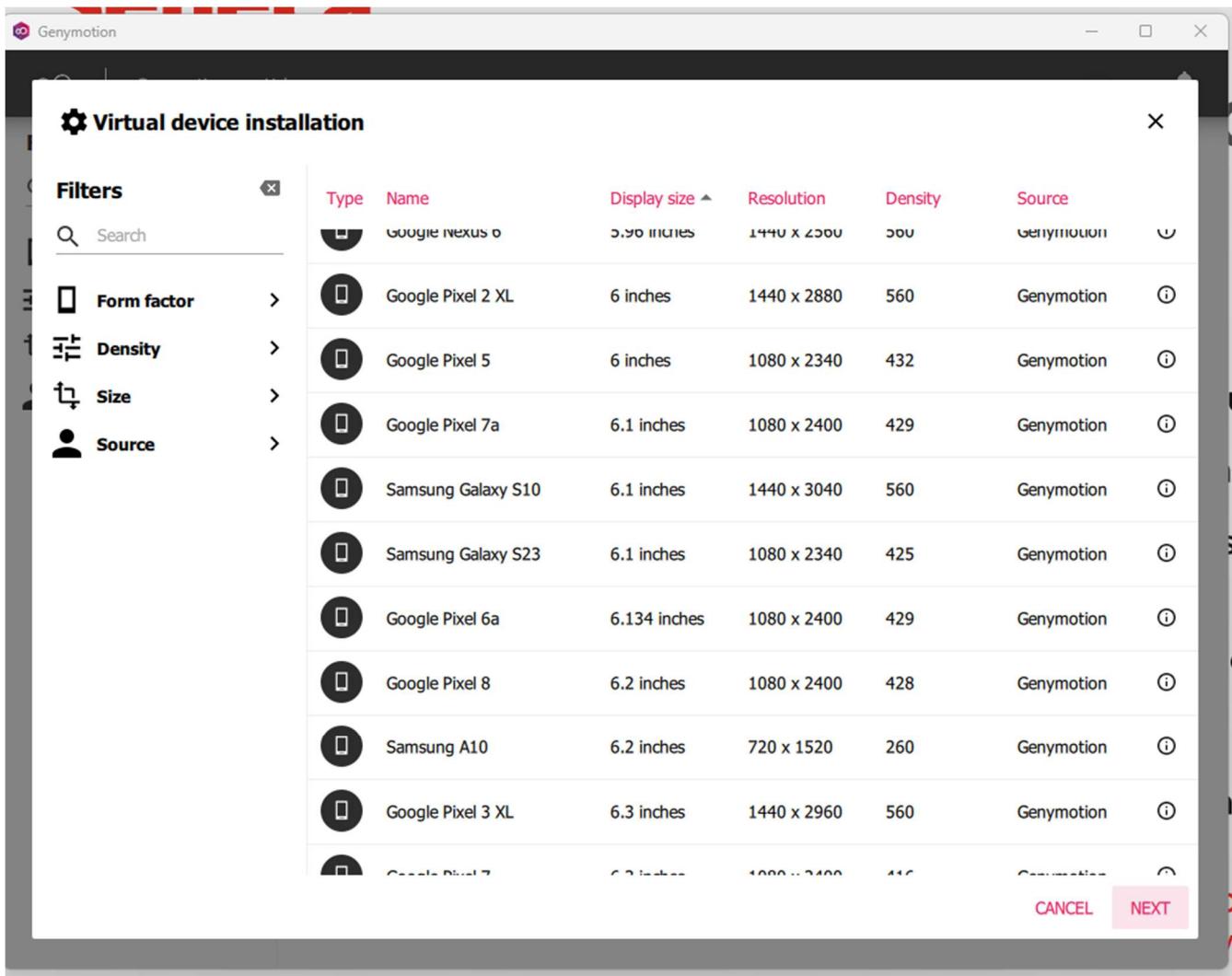
=> The whole part-1, I have completed in Seneca lab computers as my personal laptop was not good enough for part-1.



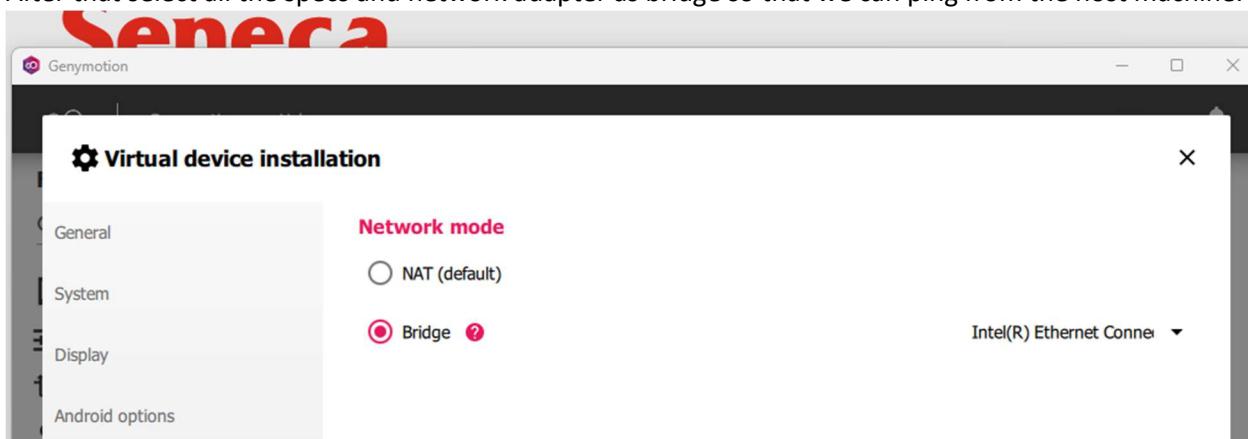


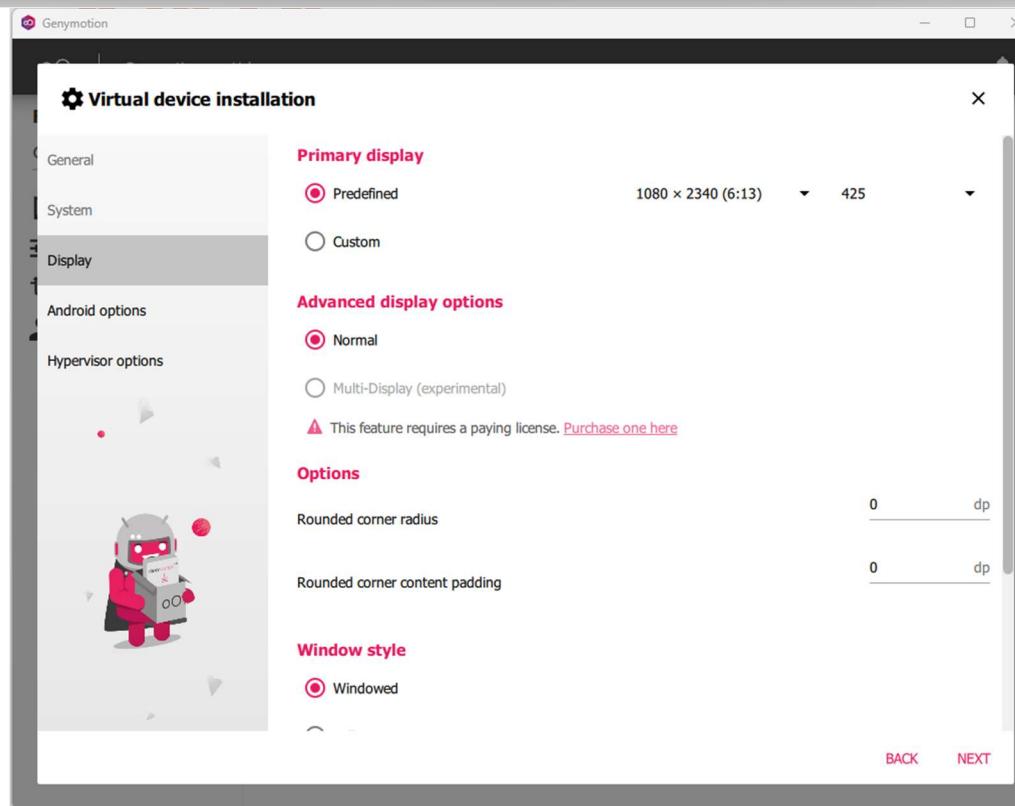
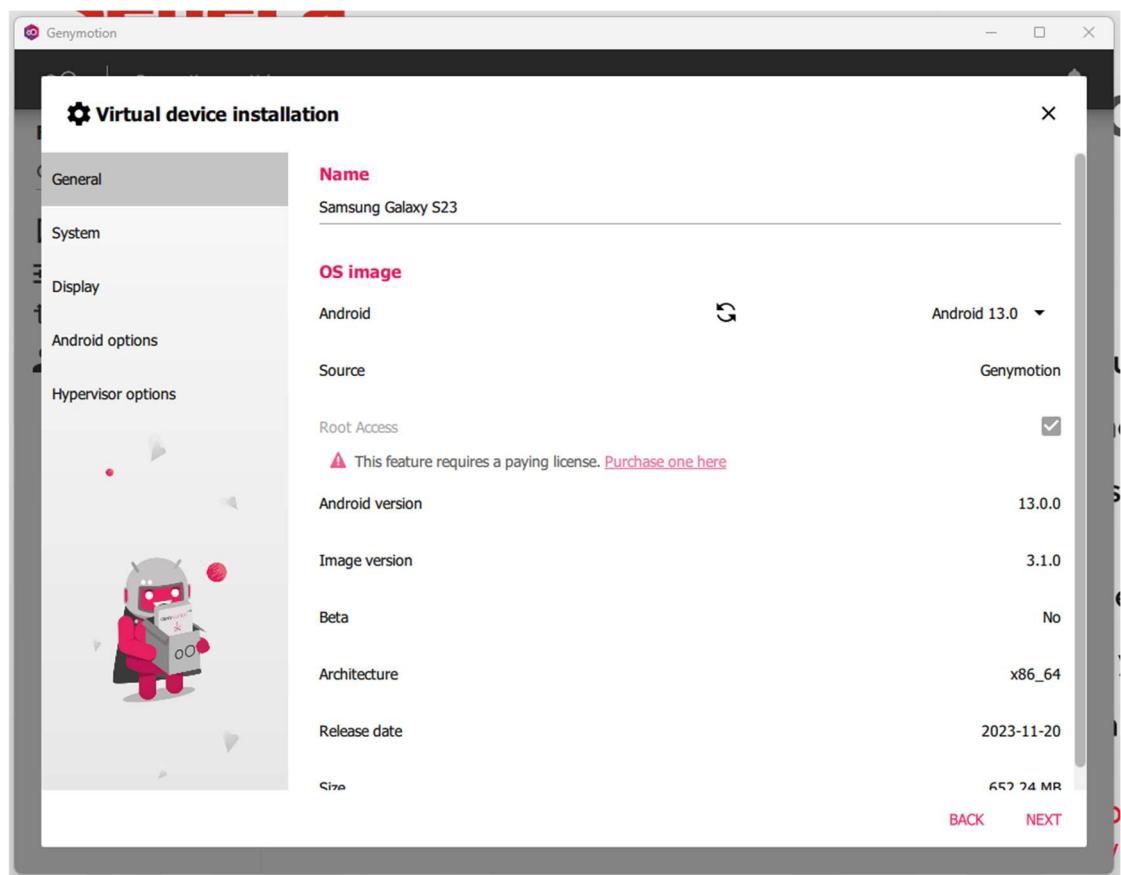
A screenshot of the Seneca Genymotion interface. The top bar includes the Seneca logo, a Genymotion icon, and navigation links for "Genymotion" and "Help". On the left, there's a "Filters" sidebar with "Search" and four filter categories: "Form factor", "Density", "Size", and "Source". The main area features a placeholder message: "You can install a virtual device by u" followed by an icon of two smartphones. Three floating notifications are visible: 1) "News & Updates" about new servers in Southeast Asia for Genymotion SaaS, with a "READ MORE" link. 2) "News & Updates" about the launch of Android 14 beta, with a "READ MORE" link. 3) "What's new in 3.7.1" about a fixed regression in Desktop, with a "READ MORE" link. At the bottom right, a "Press Restart to End Your Session" message is displayed, along with a "Restart Computer" button. The system tray at the very bottom shows icons for file explorer, browser, task manager, and system status, along with the date and time (2024-05-30 2:18 PM).

Here are all the android emulators and I selected Samsung Galaxy S23

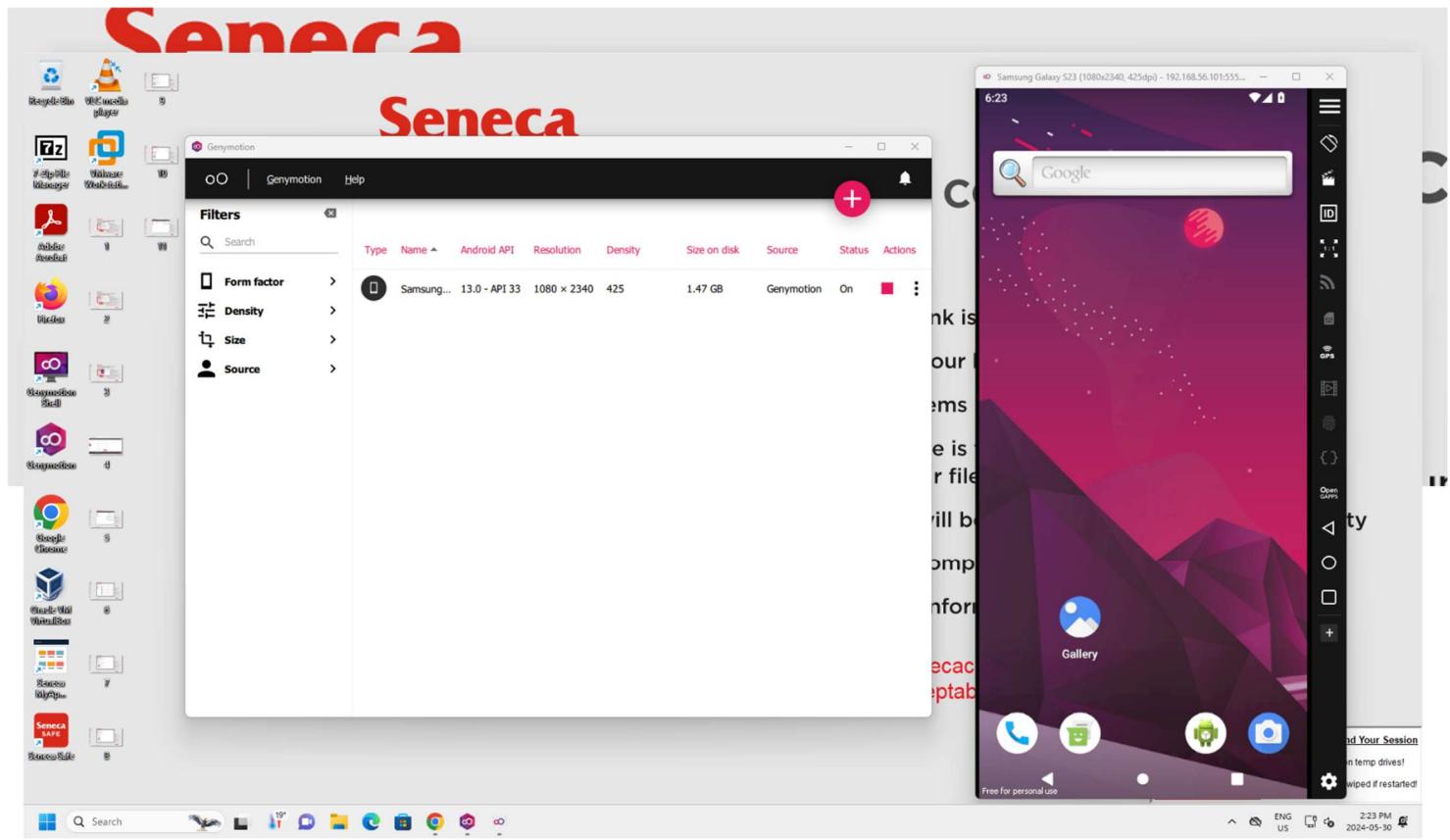


After that select all the specs and network adapter as bridge so that we can ping from the host machine.





Installation done

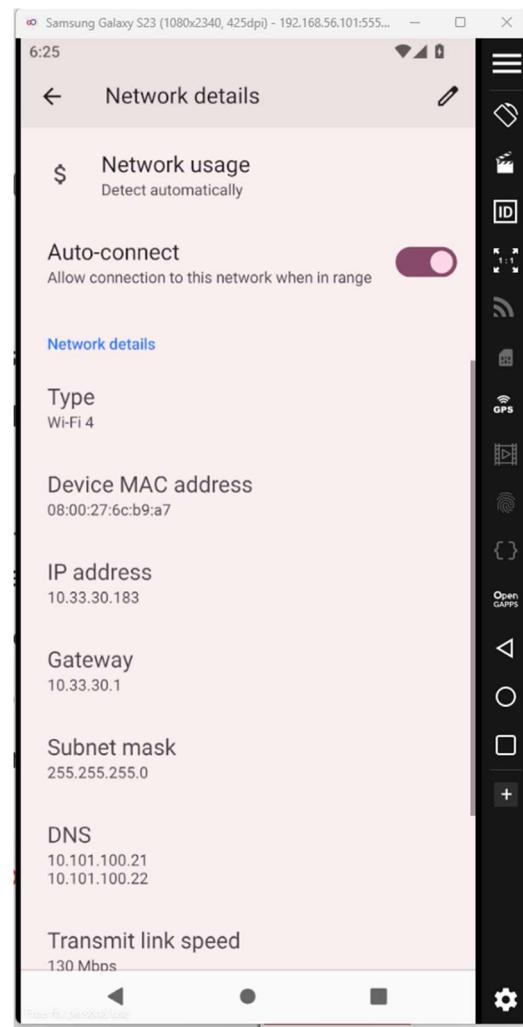


As you can see I can ping from my host machine to the emulator.

A screenshot of a Windows desktop. On the left, a Command Prompt window shows the output of a ping command to 10.33.30.183. The output includes several lines of text about packet transmission and reception times. On the right, a Network details window for the 'Samsung Galaxy S23' virtual device is open. It shows network usage, auto-connect settings, and various connection parameters like IP address (10.33.30.183), gateway (10.33.30.1), subnet mask (255.255.255.0), and DNS servers (10.101.100.21, 10.101.100.22). The transmit link speed is listed as 130 Mbps. The taskbar at the bottom includes the Start button, a search bar, and icons for File Explorer, Task View, and other system tools.

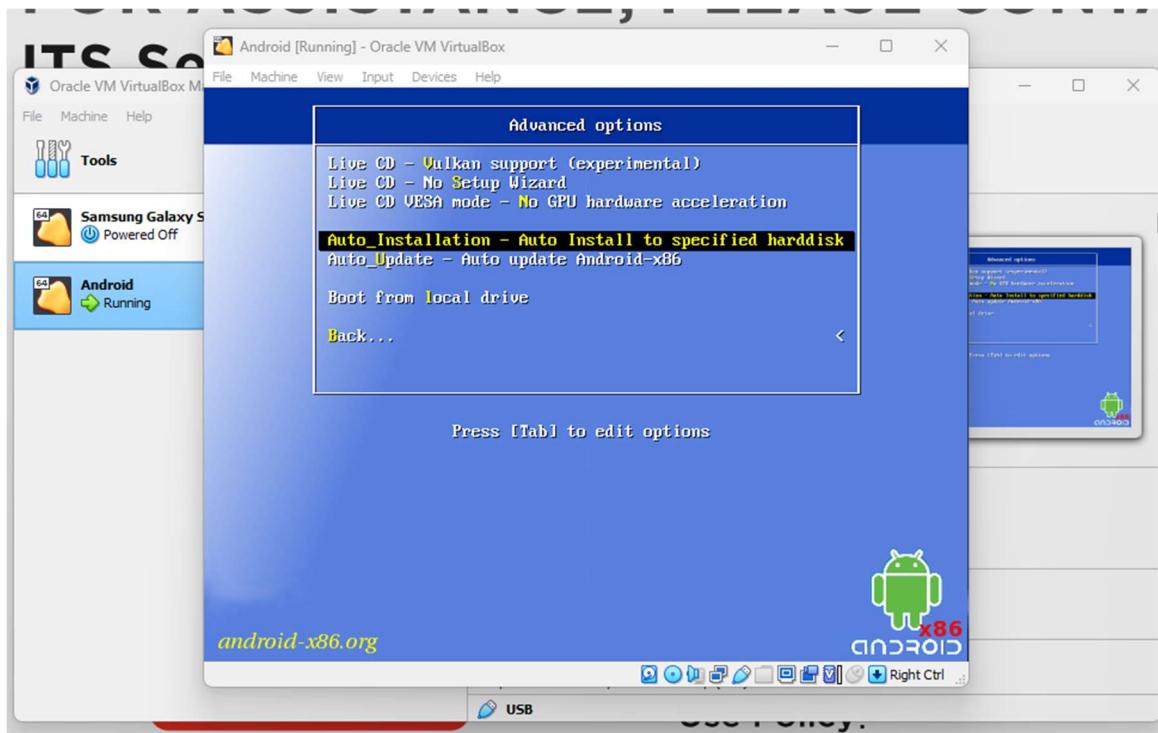
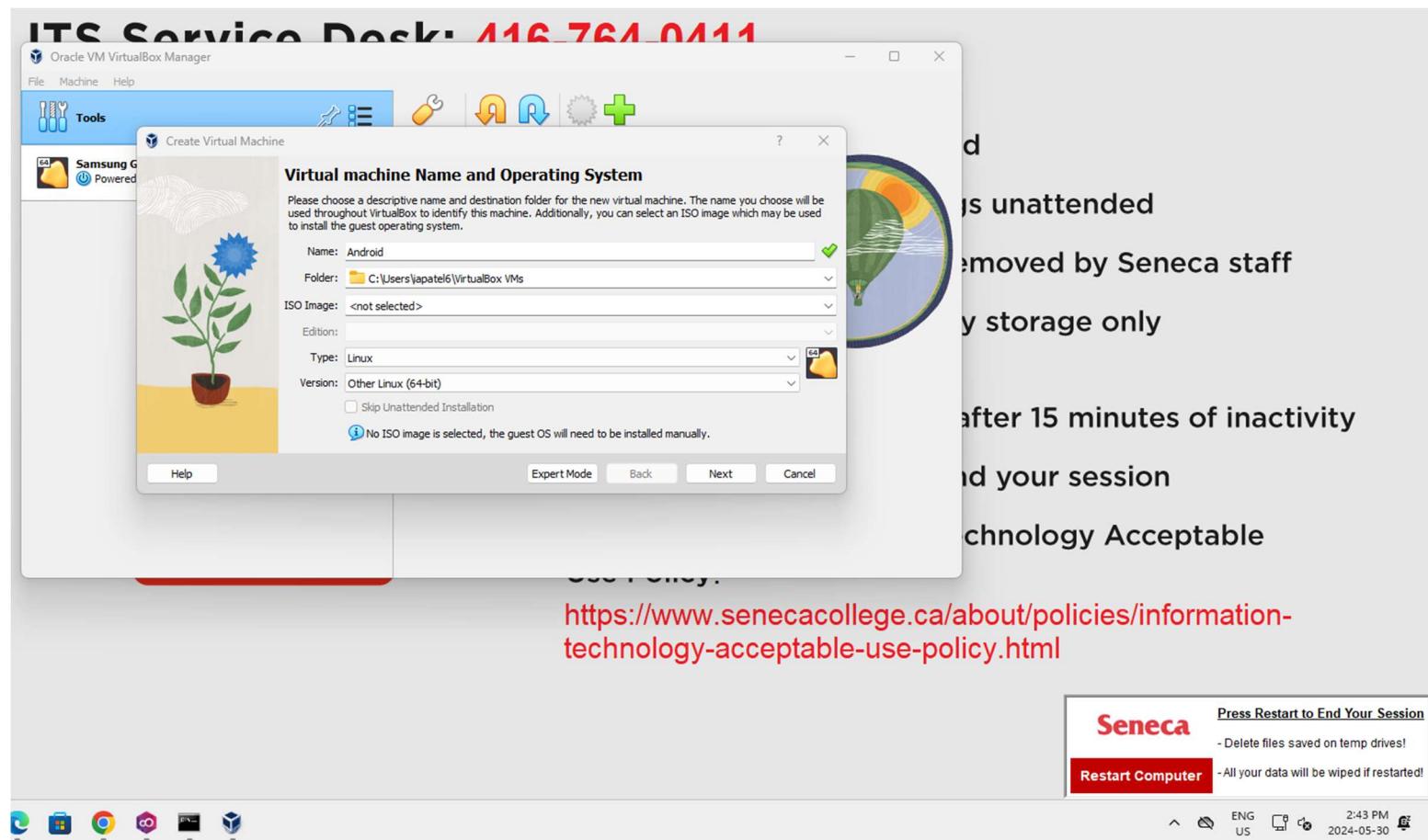
Here is the magnified version of the above screenshot.

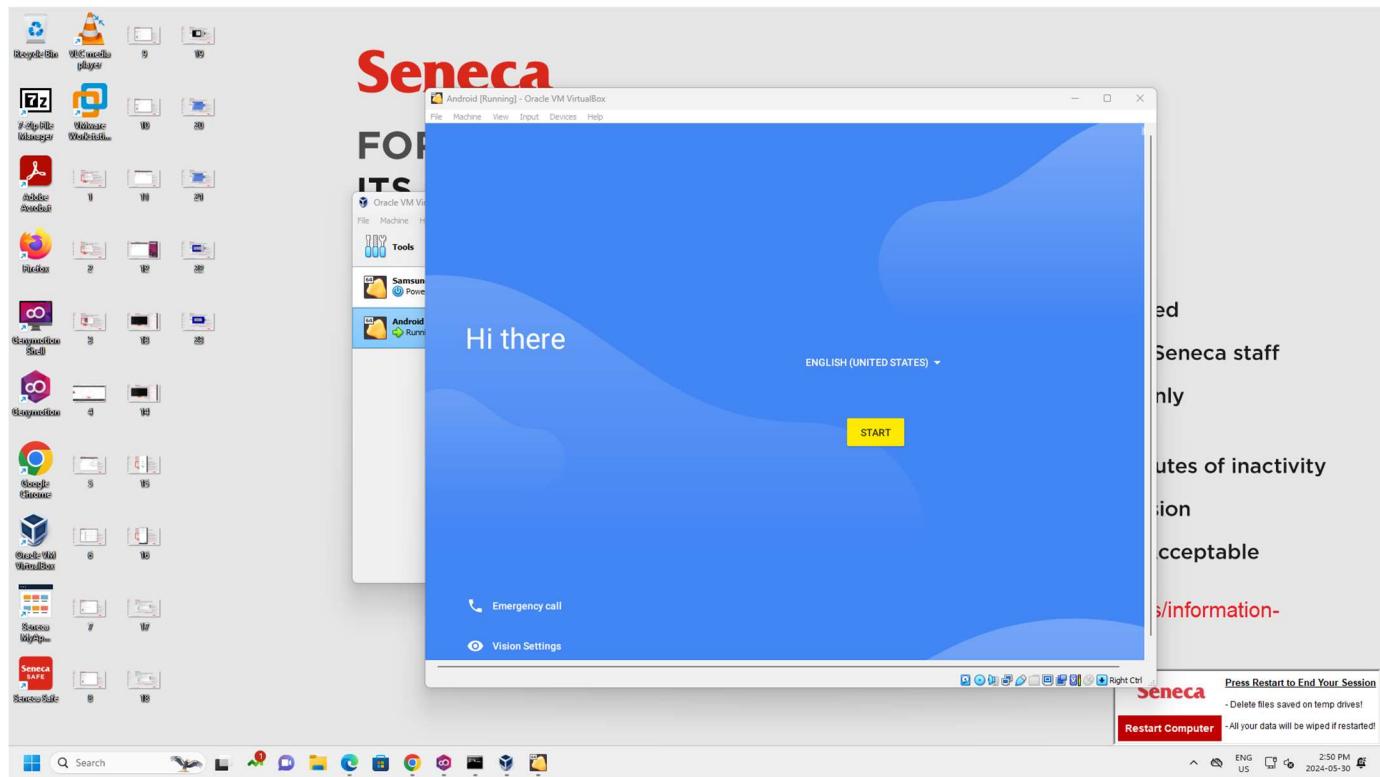
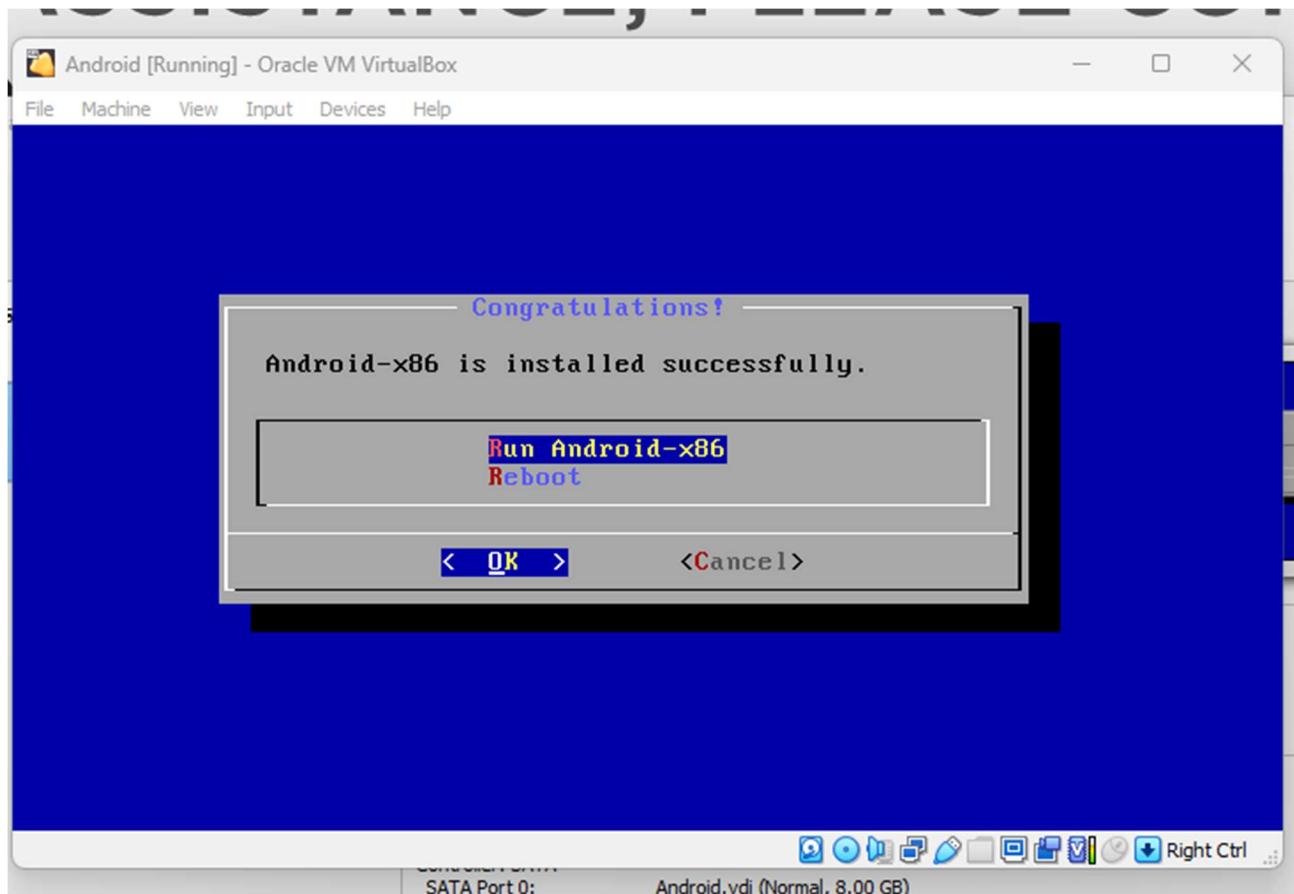
```
Command Prompt  
C:\Users\iapatel6>ping 10.33.30.183  
  
Pinging 10.33.30.183 with 32 bytes of data:  
Request timed out.  
Reply from 10.33.30.183: bytes=32 time=47ms TTL=64  
Reply from 10.33.30.183: bytes=32 time=64ms TTL=64  
Reply from 10.33.30.183: bytes=32 time=86ms TTL=64  
  
Ping statistics for 10.33.30.183:  
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),  
Approximate round trip times in milli-seconds:  
    Minimum = 47ms, Maximum = 86ms, Average = 65ms  
  
C:\Users\iapatel6>ping 10.33.30.183  
  
Pinging 10.33.30.183 with 32 bytes of data:  
Reply from 10.33.30.183: bytes=32 time=14ms TTL=64  
Reply from 10.33.30.183: bytes=32 time<1ms TTL=64  
Reply from 10.33.30.183: bytes=32 time=8ms TTL=64  
Reply from 10.33.30.183: bytes=32 time=71ms TTL=64  
  
Ping statistics for 10.33.30.183:  
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),  
Approximate round trip times in milli-seconds:  
    Minimum = 0ms, Maximum = 71ms, Average = 23ms  
  
C:\Users\iapatel6>
```



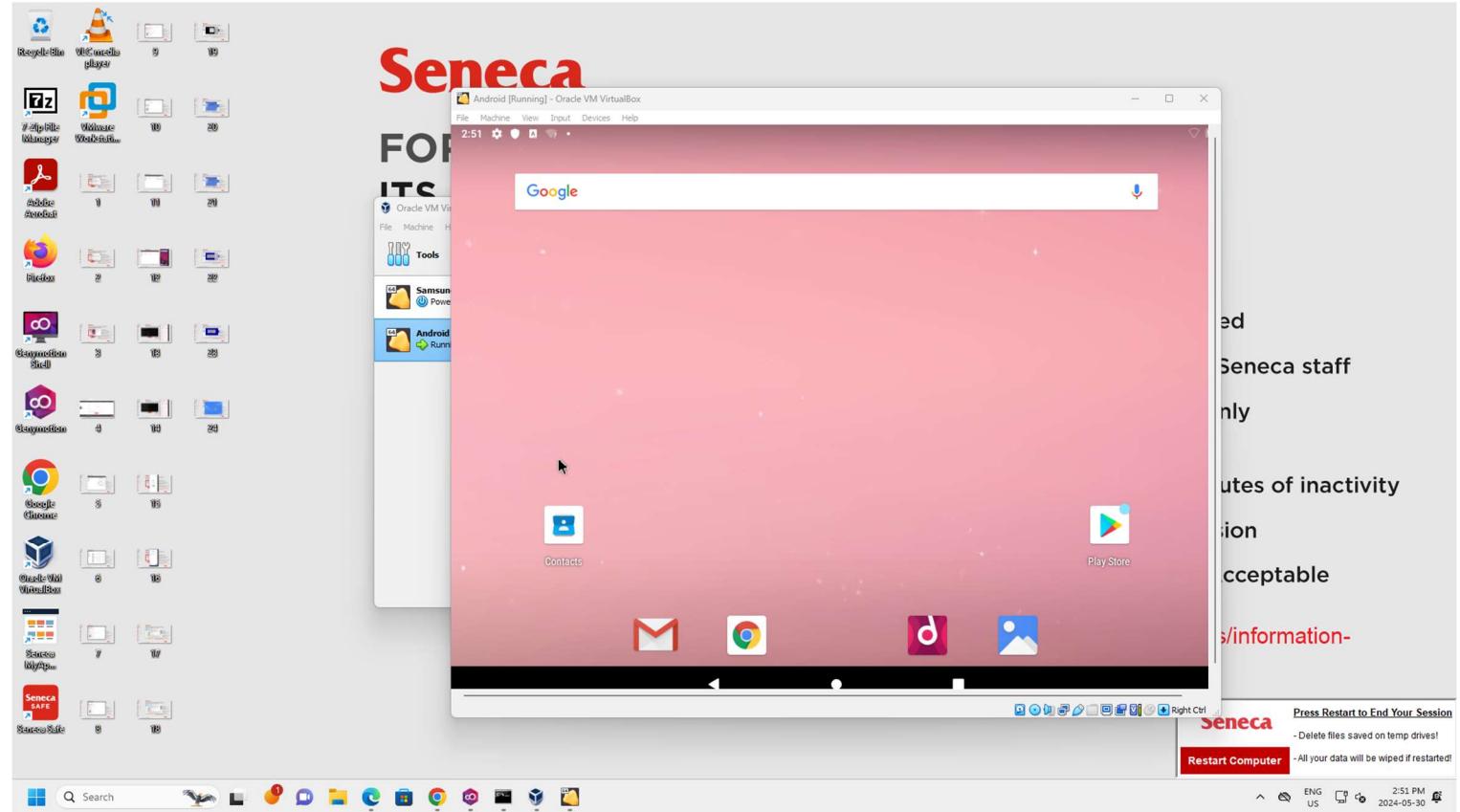
This was all for the genymotion and we will install the second emulator in the virtualbox name – android x86

2) Android x86

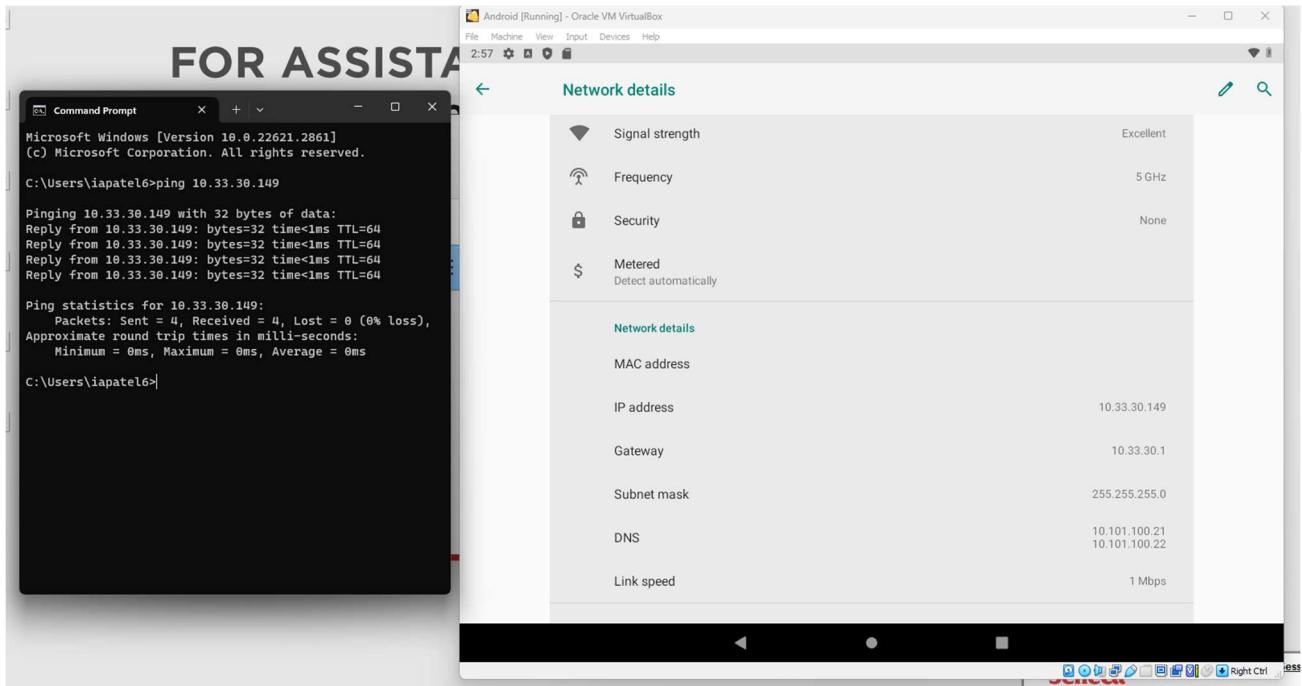




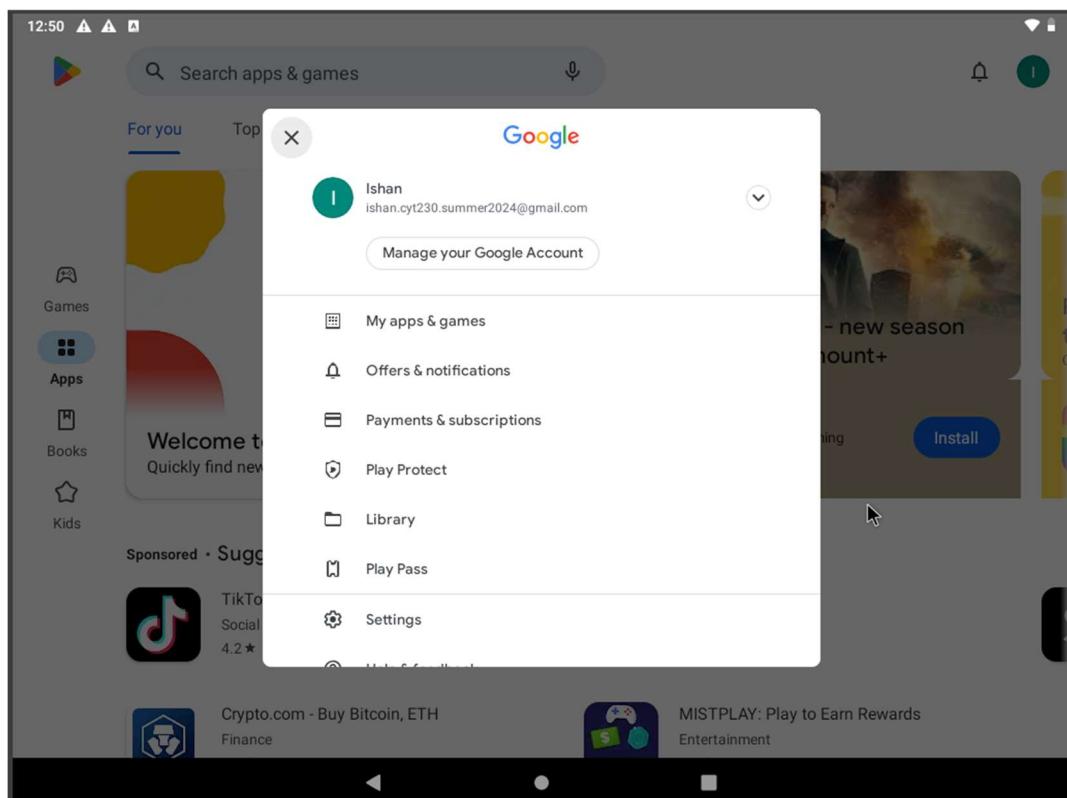
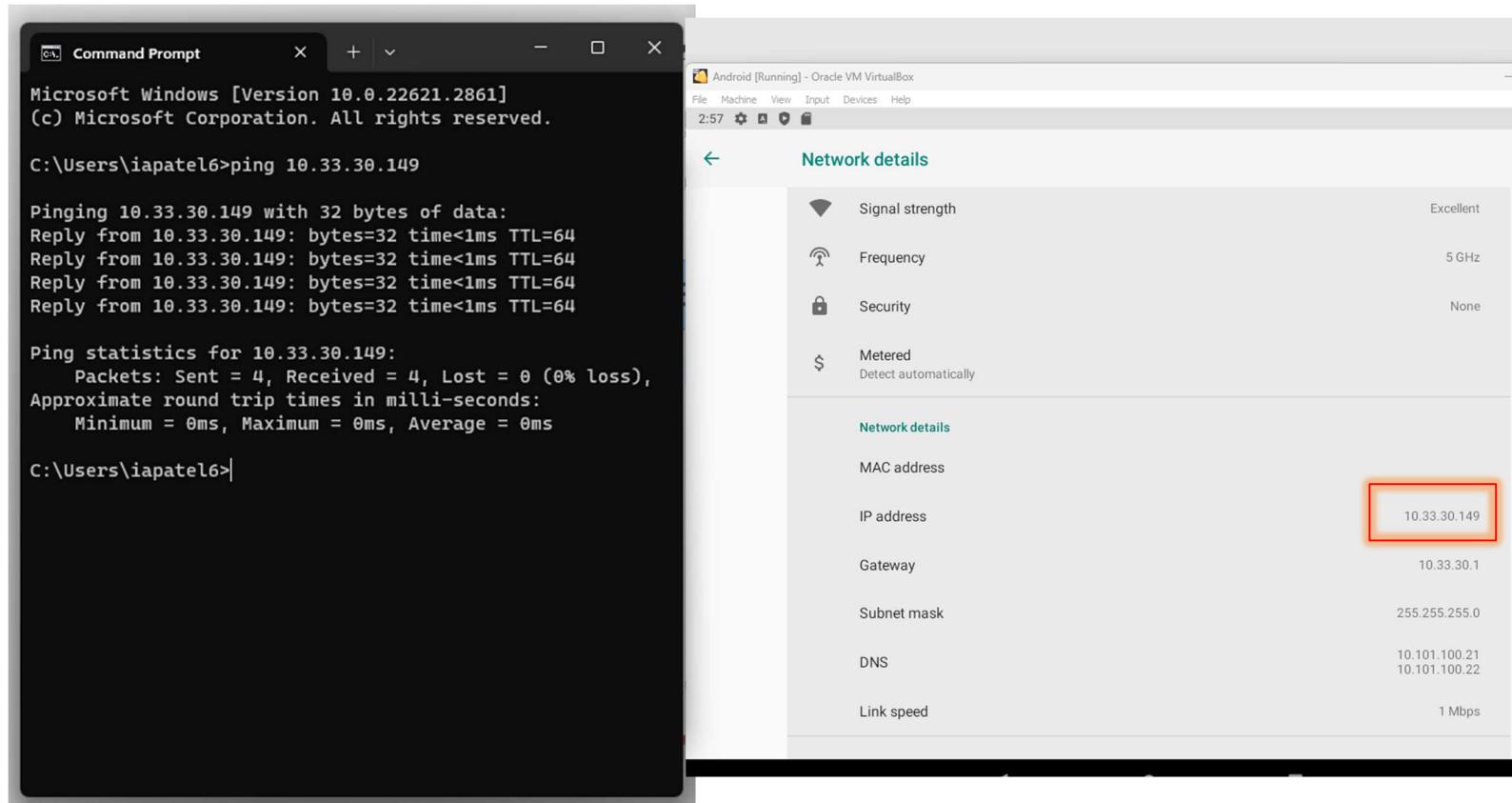
As you can see the VM is perfectly running.



We can also ping the android emulator from the host machine.



Here is the magnified version of the above screenshot.

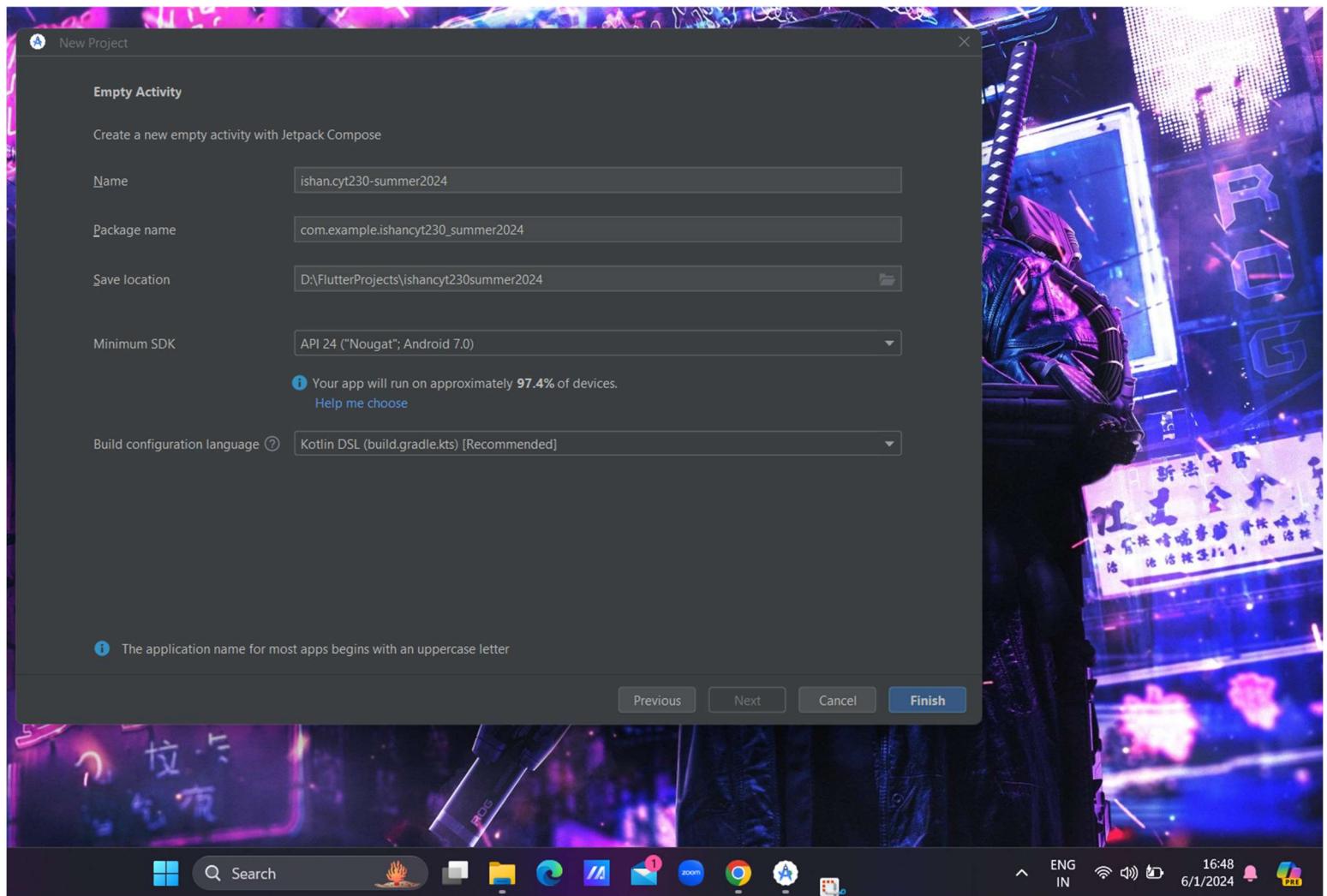


Here is the google play store account with the proper naming convention.

Part – 2

Android Studio

I installed android studio and created the project with the proper naming convention.



I followed all the steps which were presented in the link one and below is the results.

The screenshot shows the Android Studio interface with the code editor open to `MainActivity.kt`. The code defines a `MainActivity` class that overrides `onCreate` and sets its content to a `Greeting` composable with the name "Ishan". A preview window on the right shows the resulting UI with the text "Hello Android!".

```
1 package com.example.ishancyt230_summer2024
2
3 import ...
4
5 class MainActivity : ComponentActivity() {
6     override fun onCreate(savedInstanceState: Bundle?) {
7         super.onCreate(savedInstanceState)
8         enableEdgeToEdge()
9         setContent {
10             Ishancyt230summer2024Theme {
11                 Scaffold(modifier = Modifier.fillMaxSize()) { innerPadding ->
12                     Greeting(
13                         name = "Ishan",
14                         modifier = Modifier.padding(innerPadding)
15                     )
16                 }
17             }
18         }
19     }
20
21     @Composable
22     fun Greeting(name: String, modifier: Modifier = Modifier) {
23         Text(
24             text = "Hello $name!",
25             modifier = modifier
26         )
27     }
28
29     @Preview(showBackground = true)
30     @Composable
31 }
32
33 @Composable
34 fun Greeting(name: String, modifier: Modifier = Modifier) {
35     Text(
36         text = "Hello $name!",
37         modifier = modifier
38     )
39 }
40
41 @Preview(showBackground = true)
42 @Composable
```

First, I replaced the text with my name.

The screenshot shows the Android Studio interface with the code editor open to `MainActivity.kt`. The code has been modified to change the greeting text from "Hello" to "Hi, my name is Ishan". The preview window on the right shows the updated UI with the text "Hi, my name is Ishan!".

```
1 package com.example.ishancyt230_summer2024
2
3 import androidx.compose.ui.tooling.preview.Preview
4 import com.example.ishancyt230_summer2024.ui.theme.Ishancyt230summer2024Theme
5
6 class MainActivity : ComponentActivity() {
7     override fun onCreate(savedInstanceState: Bundle?) {
8         super.onCreate(savedInstanceState)
9         setContent {
10             Ishancyt230summer2024Theme {
11                 // A surface container using the 'background' color from the theme
12                 Surface(
13                     modifier = Modifier.fillMaxSize(),
14                     color = MaterialTheme.colorScheme.background
15                 ) {
16                     Greeting( name: "Ishan")
17                 }
18             }
19         }
20     }
21
22     @Composable
23     fun Greeting(name: String, modifier: Modifier = Modifier) {
24         Text(
25             text = "Hi, my name is $name!",
26             modifier = modifier
27         )
28     }
29
30     @Preview(showBackground = true)
31     @Composable
32 }
```

Then I changed the color and also added some padding.

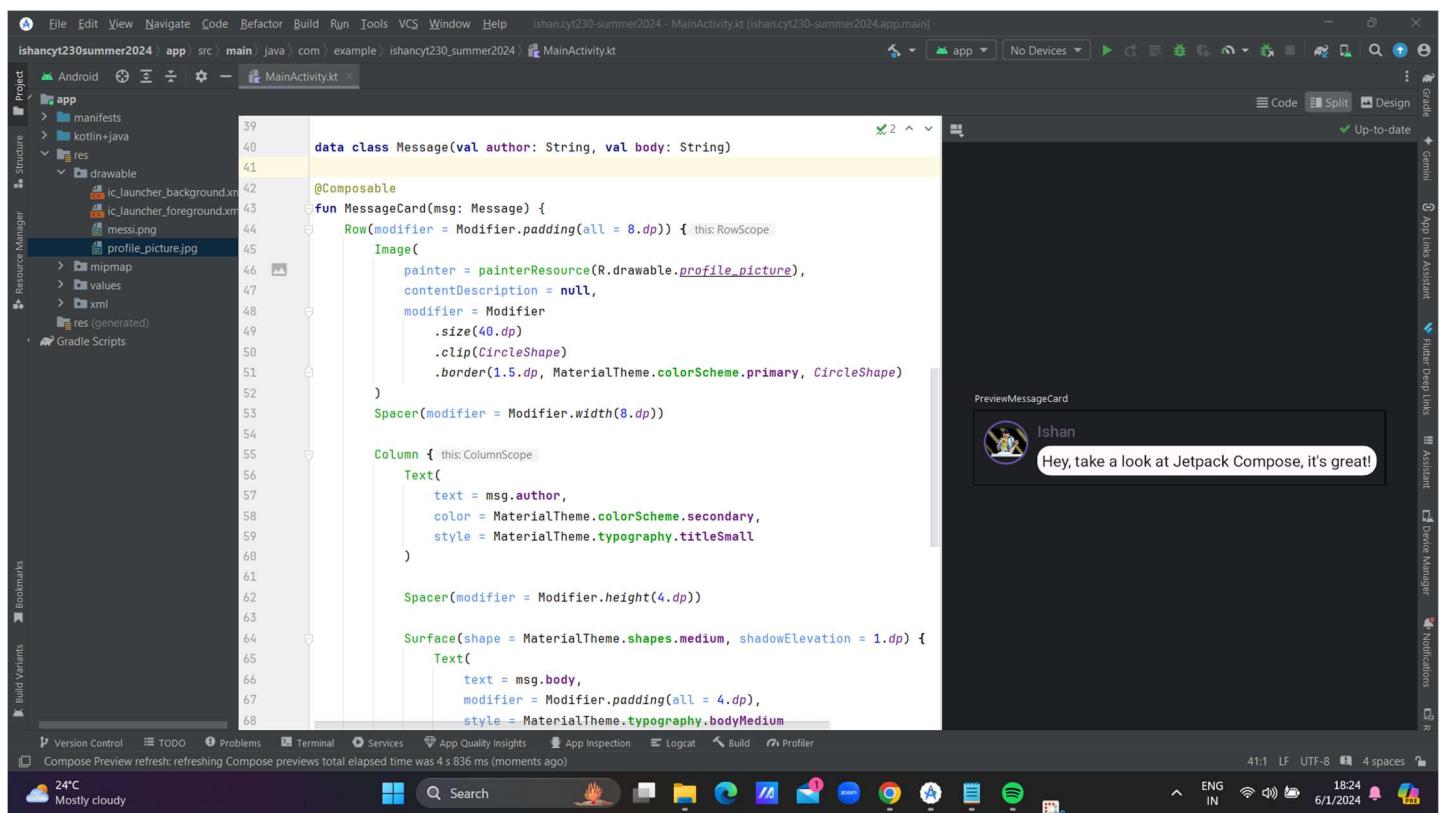
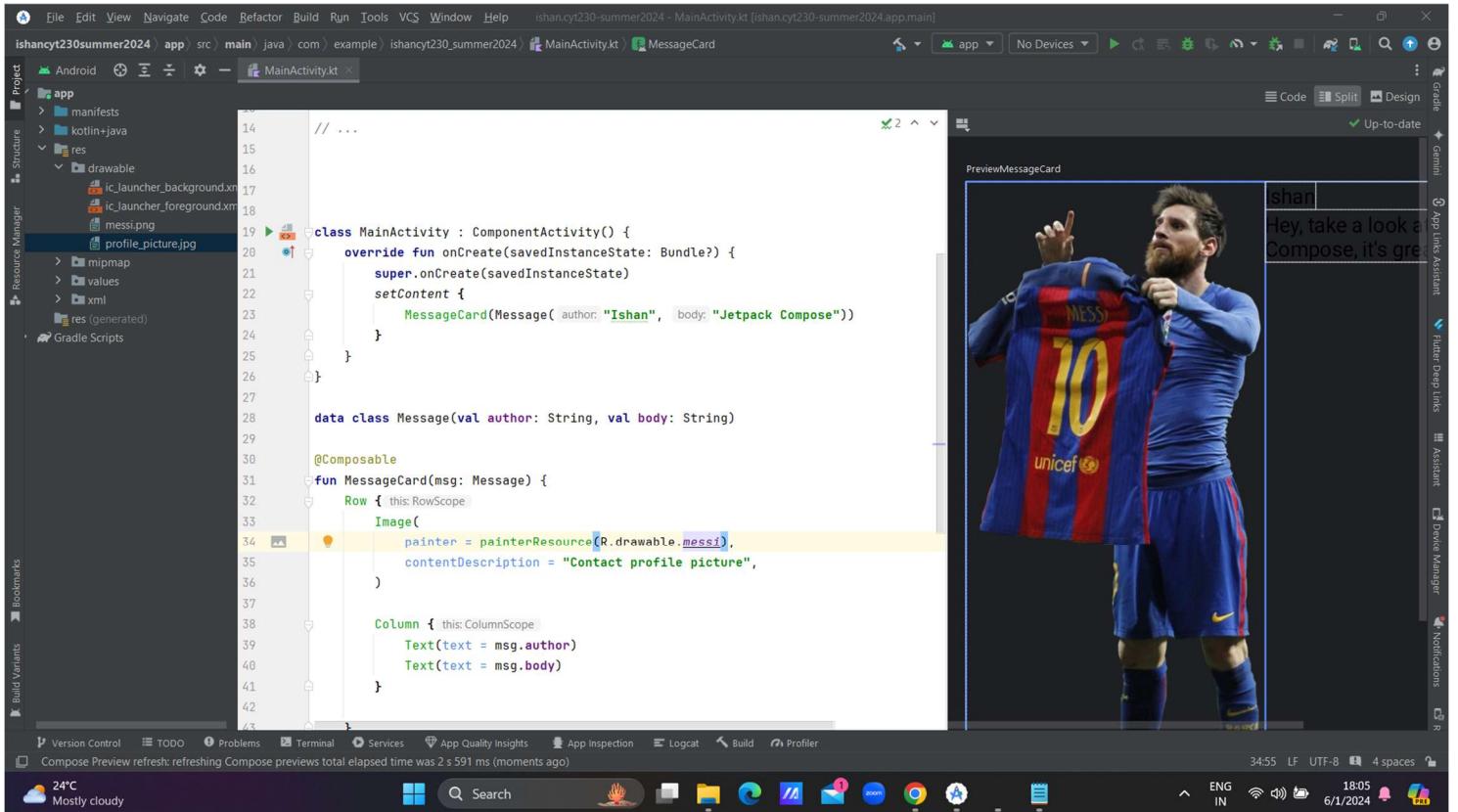
The screenshot shows the Android Studio interface with the code editor open to a Kotlin file named `MainActivity.kt`. The code defines a `Greeting` composable function and a `GreetingPreview` preview function. The `Greeting` function uses a `Surface` component with a magenta background color. The `GreetingPreview` function uses the same `Surface` component. A yellow highlight is on the `color = Color.Magenta` line. To the right, a preview window titled "GreetingPreview" shows the text "Hi, my name is Ishan!" on a magenta background.

```
21 // A surface container using the 'background' color from the theme
22 Surface(
23     modifier = Modifier.fillMaxSize(),
24     color = MaterialTheme.colorScheme.background
25 ) {
26     Greeting(name: "Ishan")
27 }
28
29
30 }
31
32
33 @Composable
34 fun Greeting(name: String, modifier: Modifier = Modifier) {
35     Surface(color = Color.Magenta) {
36         Text(
37             text = "Hi, my name is $name!",
38             modifier = modifier
39         )
40     }
41 }
42
43 @Preview(showBackground = true)
44 @Composable
45 fun GreetingPreview() {
46     Ishancyt230summer2024Theme {
47         Greeting(name: "Ishan")
48     }
49 }
50
51
52 }
```

The screenshot shows the same Android Studio interface with the code editor open to the same `MainActivity.kt` file. The `Greeting` function now includes a `padding(24.dp)` modifier in the `Text` component. A yellow highlight is on the `padding(24.dp)` line. The preview window titled "GreetingPreview" shows the text "Hi, my name is Ishan!" with a 24dp padding around it, displayed on a magenta background.

```
26     color = MaterialTheme.colorScheme.background
27 ) {
28     Greeting(name: "Ishan")
29 }
30
31
32 }
33
34 @Composable
35 fun Greeting(name: String, modifier: Modifier = Modifier) {
36     Surface(color = Color.Magenta) {
37         Text(
38             text = "Hi, my name is $name!",
39             modifier = modifier.padding(24.dp)
40         )
41     }
42 }
43
44 @Preview(showBackground = true)
45 @Composable
46 fun GreetingPreview() {
47     Ishancyt230summer2024Theme {
48         Greeting(name: "Ishan")
49     }
50 }
51
52 }
```

Next, I tried to play around with the UI and here are few results.



Here is the badge I got when I completed all the steps.

The screenshot shows a Microsoft Edge browser window with multiple tabs open. The active tab is 'developers.google.com/profile/u/105919475233926224239'. The page displays a 'Google for Developers' profile with various sections like 'Products', 'Solutions', 'Events', 'Learn', 'Community', 'Developer Program' (which is selected), and 'Blog'. A search bar and language dropdown ('English') are also present. On the left, there's a 'STATS' section showing '2 Badges earned'. The main content area features a large 'Learning' badge card. The badge icon is a circular graphic with a blue and white geometric pattern. Below it, the text reads 'EARNED JUN 1, 2024' and 'Learning'. The description states 'Completed Learning Activities across Google's developer ecosystem'. There are 'Badge details' and 'Share' buttons. To the right of the badge card, there's a sidebar with 'Page info' and links for 'Info', 'Chat', and 'API'. A decorative blue bird illustration is visible on the right side of the page. At the bottom, there's a cookie consent banner from developers.google.com with 'Agree' and 'No thanks' buttons, along with a system tray showing weather, date, and time information.

Part – 3

App Manifest

Step 1: Install ADB Tool on the Host Machine

First, you need to install the Android Debug Bridge (ADB) tool on your host machine.

Note: Don't forget to turn on USB debugging in the emulator.

Step 2: Connect to the Android Emulator

Next, you need to connect to your Android emulator. Ensure your emulator is running.

Open a terminal or command prompt.

Use the following command to list all connected devices:

Command: adb devices

Step 3: Locate the base.apk File

Now, locate the base.apk file within the emulator. The base.apk file is usually found in the /data/app/ directory of the emulator.

Start a shell session on the emulator:

Command: adb shell

Navigate to the directory containing the APK files. Typically, this is in /data/app/:

Command: cd /data/app/ ls

This will list the directories of installed applications.

Find the directory corresponding to your app. It will typically be in the form of package.name-1.

Step 4: Pull the base.apk File to the Host Machine

Use the adb pull command to transfer the APK file to your host machine.

Exit the shell session by typing exit.

Pull the APK file to your host machine:

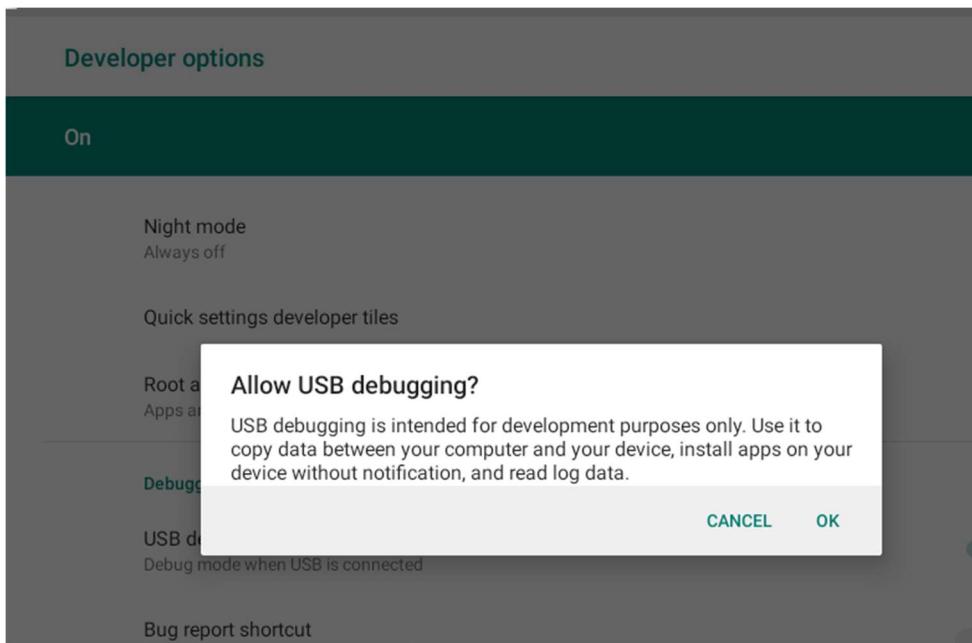
Command: adb pull /data/app/package.name-1/base.apk /path/to/save/base.apk

Step 5: Decompile the APK File

Decompile the APK file to access its contents, including the AndroidManifest.xml file. You can use an online APK decompiler or a tool like JADX.

I used an Online Decompiler

USB debugger turned on



Now connect to the android VM with cmd in your host machine.

```
Microsoft Windows [Version 10.0.22631.3593]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ASUS>adb --version
Android Debug Bridge version 1.0.41
Version 35.0.1-11580240
Installed as D:\ADB\platform-tools\adb.exe
Running on Windows 10.0.22631

C:\Users\ASUS>adb connect 10.242.19.108:5555
* daemon not running; starting now at tcp:5037
* daemon started successfully
connected to 10.242.19.108:5555

C:\Users\ASUS>adb devices
List of devices attached
10.242.19.108:5555    device
```

```
Command Prompt - adb shel + ^ Microsoft Windows [Version 10.0.22631.3593] (c) Microsoft Corporation. All rights reserved. C:\Users\ASUS>adb connect 10.242.19.108:5555 already connected to 10.242.19.108:5555 C:\Users\ASUS>adb devices List of devices attached 10.242.19.108:5555 device C:\Users\ASUS>adb shell x86_64:/ $ cd /data/app/ x86_64:/data/app $ ls ls: .: Permission denied 1|x86_64:/data/app $ su :/ # ls acct init.superuser.rc sbin bin init.usb.configfs.rc sdcard bugreports init.usb.rc sepolicy cache init.zygote32.rc storage charger init.zygote64_32.rc sys config lib system d mnt ueventd.android_x86_64.rc data odm ueventd.rc default.prop oem vendor dev plat_file_contexts vendor_file_contexts etc plat_hwservice_contexts vendor_hwservice_contexts fstab.android_x86_64 plat_property_contexts vendor_property_contexts init plat_seapp_contexts vendor_seapp_contexts
```

```
:/ # cd data/app
:/data/app # ls
com.android.vending-8l_5ynPhIm69qFf1wajziw==
com.google.android.gms-uf9p25ZHhElhjvK3KewHQ==
com.google.android.googlequicksearchbox-xZtEJI_0x6grQI0lcT5cQQ==
com.uptodown-VfJbLHm41C2k8E2kKN8ZKg==
:/data/app # cd com.android.vending-8l_5ynPhIm69qFf1wajziw==
:/data/app/com.android.vending-8l_5ynPhIm69qFf1wajziw== # ls
base.apk base.dm lib oat split_config.en.apk split_config.x86_64.apk
:/data/app/com.android.vending-8l_5ynPhIm69qFf1wajziw== #
```

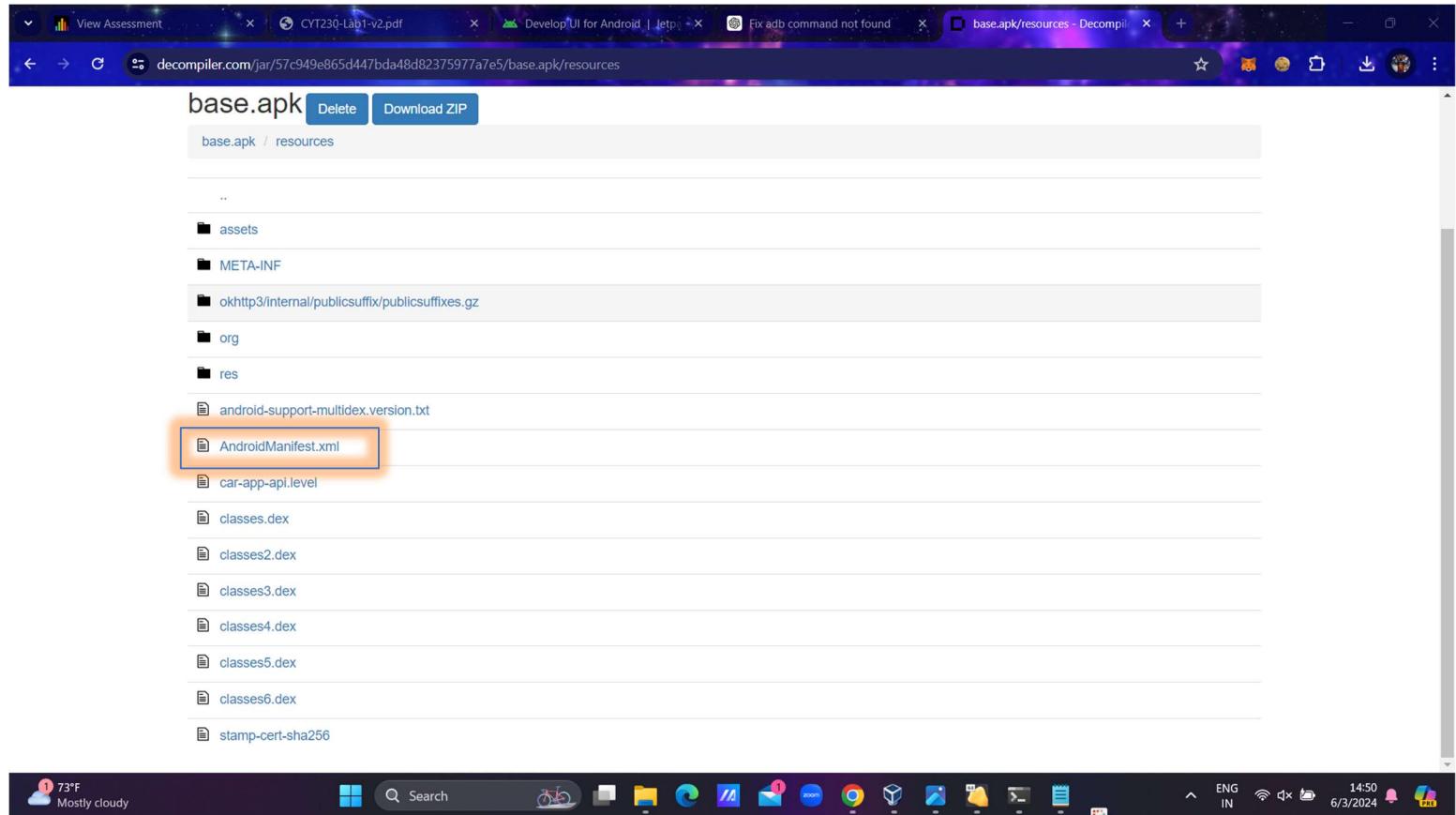
These are the applications I found in the android and I selected the first one. And in the last second line you can see that there is the base.apk file, now I need to pull this file to the host machine.

```
C:\Users\ASUS>adb pull /data/app/com.android.vending-8l_5ynPhIm69qFf1wajziw==/base.apk C:\Users\ASUS\Desktop  
/data/app/com.android.vending-8l_5ynPhIm69qFf1wajziw==/base.apk pulled, 0 skipped. 9.2 MB/s (24022079 bytes in 2.493s)  
C:\Users\ASUS>
```

I have used the online decompiler to get to the android.manifest.xml file.

The screenshot shows a web browser window with multiple tabs open. The active tab is titled "base.apk - Decompiler.com" and displays the contents of the APK file. The page header includes the site name "Decompiler.com", a "JAR String Editor" link, and links for "Release Notes" and "support@decompiler.com". A banner at the top of the page reads "Auth. Built for Devs, by Devs. Add login, SSO, MFA, passkeys and more in minutes. Start for free." Below this, a message states: "Decompilation of your artifact takes longer than usual. You may start observing already decompiled files, while others are still being decompiled. This message will disappear when decompilation is fully completed." The main content area shows the APK file "base.apk" with options to "Delete" or "Download ZIP". Under the file name, there are two folder icons labeled "resources" and "sources". The browser's taskbar at the bottom shows various pinned icons, including a weather widget for "73°F Mostly cloudy", a search bar, and several application icons like File Explorer, Edge, and Mail. The system tray on the right shows the date "6/3/2024", time "14:50", and battery status.

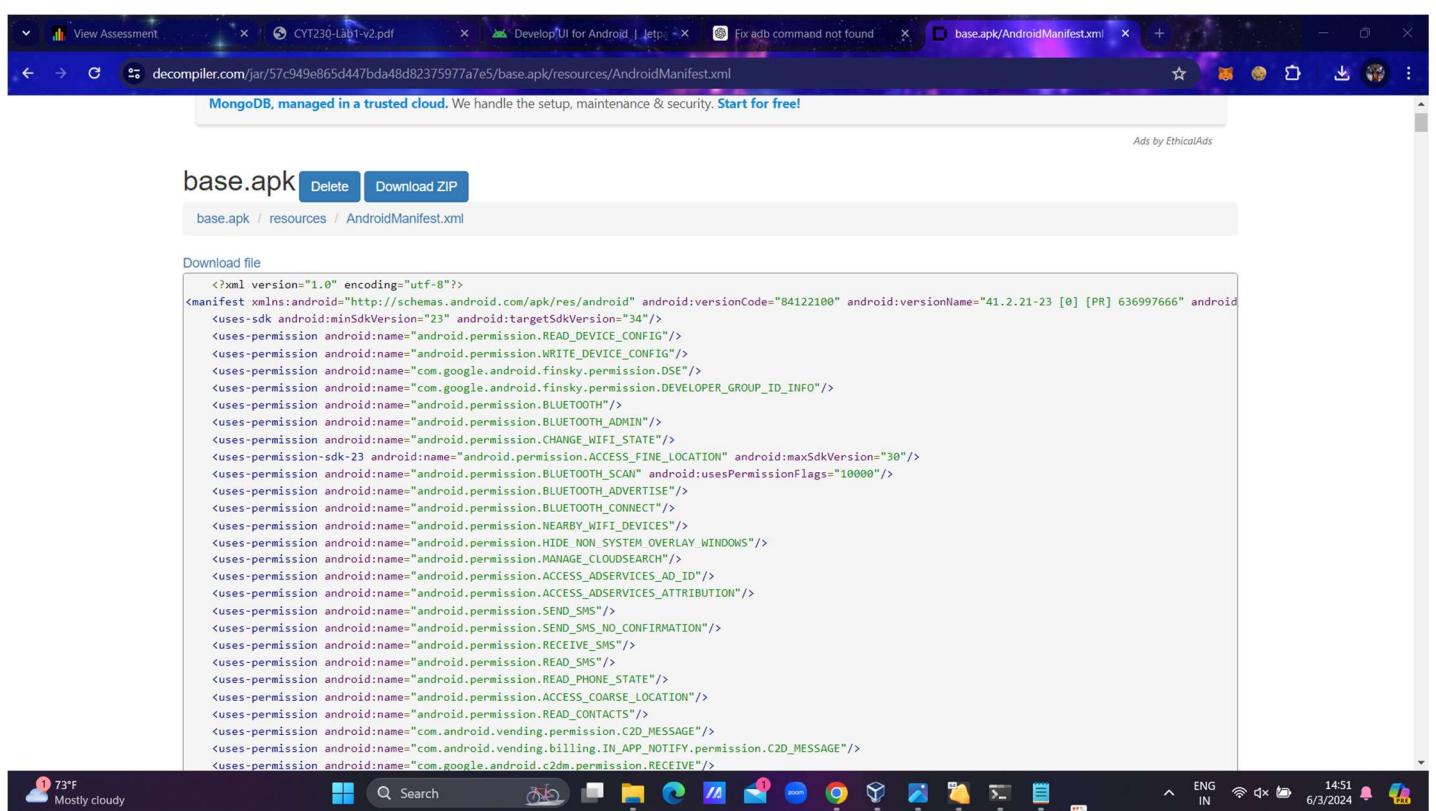
Now if we go to the resources, we can find the manifest file.



The screenshot shows a Windows desktop environment. At the top, there is a taskbar with various icons and a system tray showing the date and time (6/3/2024). The main window is a file explorer for the file 'base.apk/resources'. The left sidebar shows a tree view of the directory structure:

- ..
- assets
- META-INF
- okhttp3/internal/publicsuffix/publicsuffixes.gz
- org
- res
 - android-support-multidex.version.txt
 - AndroidManifest.xml**
 - car-app-api.level
 - classes.dex
 - classes2.dex
 - classes3.dex
 - classes4.dex
 - classes5.dex
 - classes6.dex
 - stamp-cert-sha256

The 'AndroidManifest.xml' file is highlighted with a red box.



The screenshot shows a Windows desktop environment. The taskbar at the bottom includes weather information (73°F, Mostly cloudy), a search bar, and various application icons. The main window is a web browser displaying the contents of the 'AndroidManifest.xml' file from the 'base.apk' resource. The page header says 'MongoDB, managed in a trusted cloud. We handle the setup, maintenance & security. Start for free!' and includes an 'Ads by EthicalAds' notice. The content of the XML file is displayed in a large text area:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="84122100" android:versionName="41.2.21-23 [0] [PR] 636997666" android:allowBackup="true" android:label="base.apk" android:icon="@mipmap/ic_launcher" android:debuggable="false">
    <uses-sdk android:minSdkVersion="23" android:targetSdkVersion="34"/>
    <uses-permission android:name="android.permission.READ_DEVICE_CONFIG"/>
    <uses-permission android:name="android.permission.WRITE_DEVICE_CONFIG"/>
    <uses-permission android:name="com.google.android.finsky.permission.DSE"/>
    <uses-permission android:name="com.google.android.finsky.permission.DEVELOPER_GROUP_ID_INFO"/>
    <uses-permission android:name="android.permission.BLUETOOTH"/>
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
    <uses-permission android:name="android.permission.CHANGE_WIFI_STATE"/>
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" android:maxSdkVersion="30"/>
    <uses-permission android:name="android.permission.BLUETOOTH_SCAN" android:usesPermissionFlags="10000"/>
    <uses-permission android:name="android.permission.BLUETOOTH_ADVERTISE"/>
    <uses-permission android:name="android.permission.BLUETOOTH_CONNECT"/>
    <uses-permission android:name="android.permission.NEARBY_WIFI_DEVICES"/>
    <uses-permission android:name="android.permission.HIDE_NON_SYSTEM_OVERLAY_WINDOWS"/>
    <uses-permission android:name="android.permission.MANAGE_CLOUDSEARCH"/>
    <uses-permission android:name="android.permission.ACCESS_ADSERVICES_AD_ID"/>
    <uses-permission android:name="android.permission.ACCESS_ADSERVICES_ATTRIBUTION"/>
    <uses-permission android:name="android.permission.SEND_SMS"/>
    <uses-permission android:name="android.permission.SEND_SMS_NO_CONFIRMATION"/>
    <uses-permission android:name="android.permission.RECEIVE_SMS"/>
    <uses-permission android:name="android.permission.READ_SMS"/>
    <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
    <uses-permission android:name="android.permission.READ_CONTACTS"/>
    <uses-permission android:name="com.android.vending.permission.C2D_MESSAGE"/>
    <uses-permission android:name="com.android.vending.billing.IN_APP_NOTIFY.permission.C2D_MESSAGE"/>
    <uses-permission android:name="com.google.android.c2dm.permission.RECEIVE"/>
```

For the second part, it was quite easy as it was in android studio. I could easily locate the android manifest file.

The screenshot shows the Android Studio interface with the project 'ishancyt230summer2024' selected. The 'app' module is open, and the 'AndroidManifest.xml' file is the active tab in the code editor. The code editor displays the XML manifest file with syntax highlighting for tags like <manifest>, <application>, and <activity>. The 'Structure' tool window on the left shows the project's directory structure, including 'app', 'manifests', 'res' (with drawable, mipmap, values, and xml subfolders), and 'gradle Scripts'. The bottom status bar shows various developer tools and system information, such as 'Microsoft Defender configuration' and the current date and time.

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

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="ishan.cyt230-summer2024"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Ishancyt230summer2024"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="ishan.cyt230-summer2024"
            android:theme="@style/Theme.Ishancyt230summer2024">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Components: Android Emulator, Android SDK
Platform-Tools
Update...