

## Lab 4 i Datanettverk

Utstyr	Beskrivelse
2 Laptops	Begge kjører ubuntu 64 bit
Cat 5e kabel	Maskinene er koblet fysisk via denne
Ruter?	Vi har ikke tilgang til en ruter eller tilgang til den som er i kollektivet ettersom at vi bor i et kollektiv med skolen sitt nett.
Tjener	Khayam
Klient	Heljar

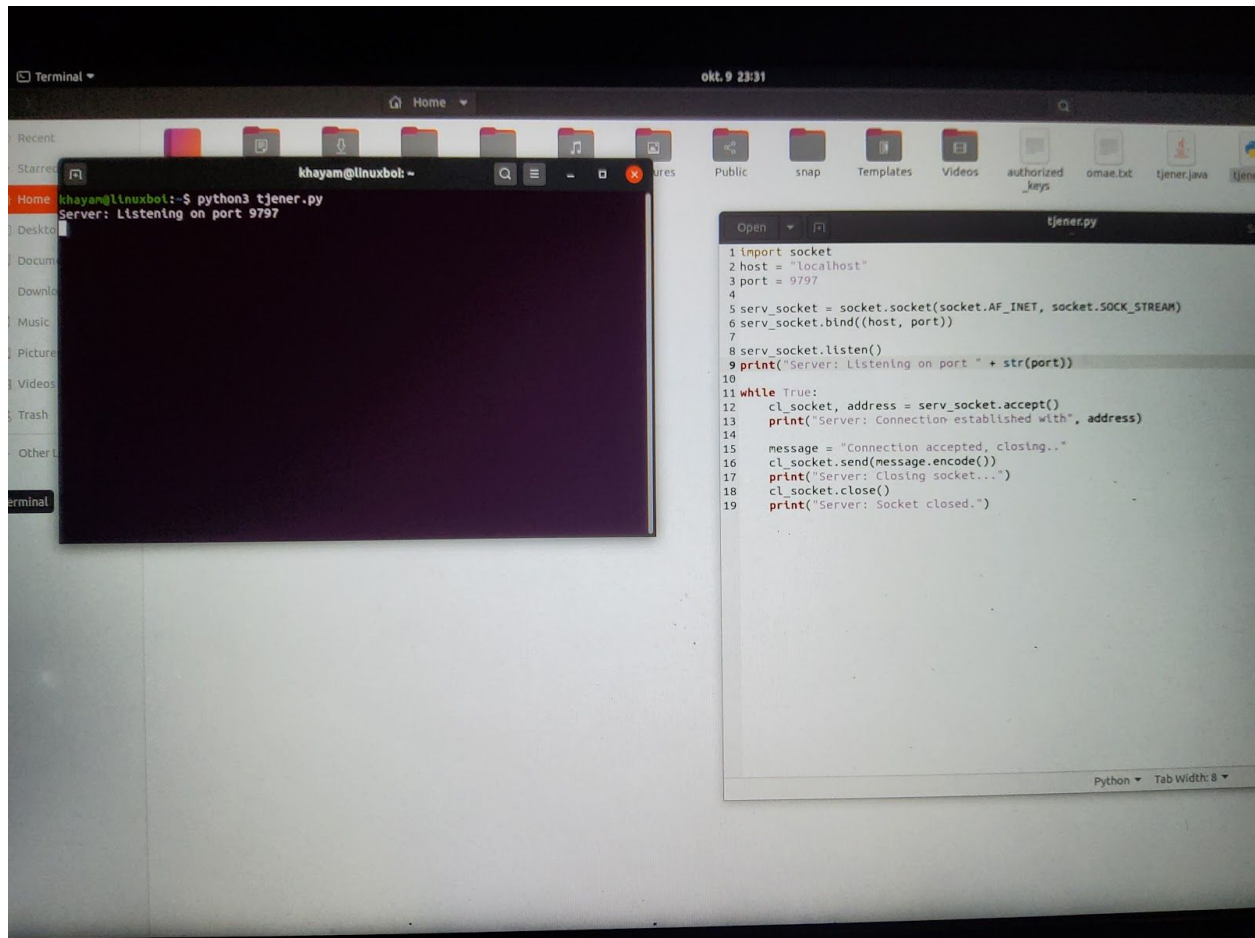
### Oppgave 1:

Først så lagde vi filene til python programmene. Videre så sjekket vi om python var installert på maskinen ved å prøve å kjøre filene uten å gjøre noen endringer.

Terminalkoden på host:

```
Cd /home/khayam
```

```
Python3 tjener.py
```

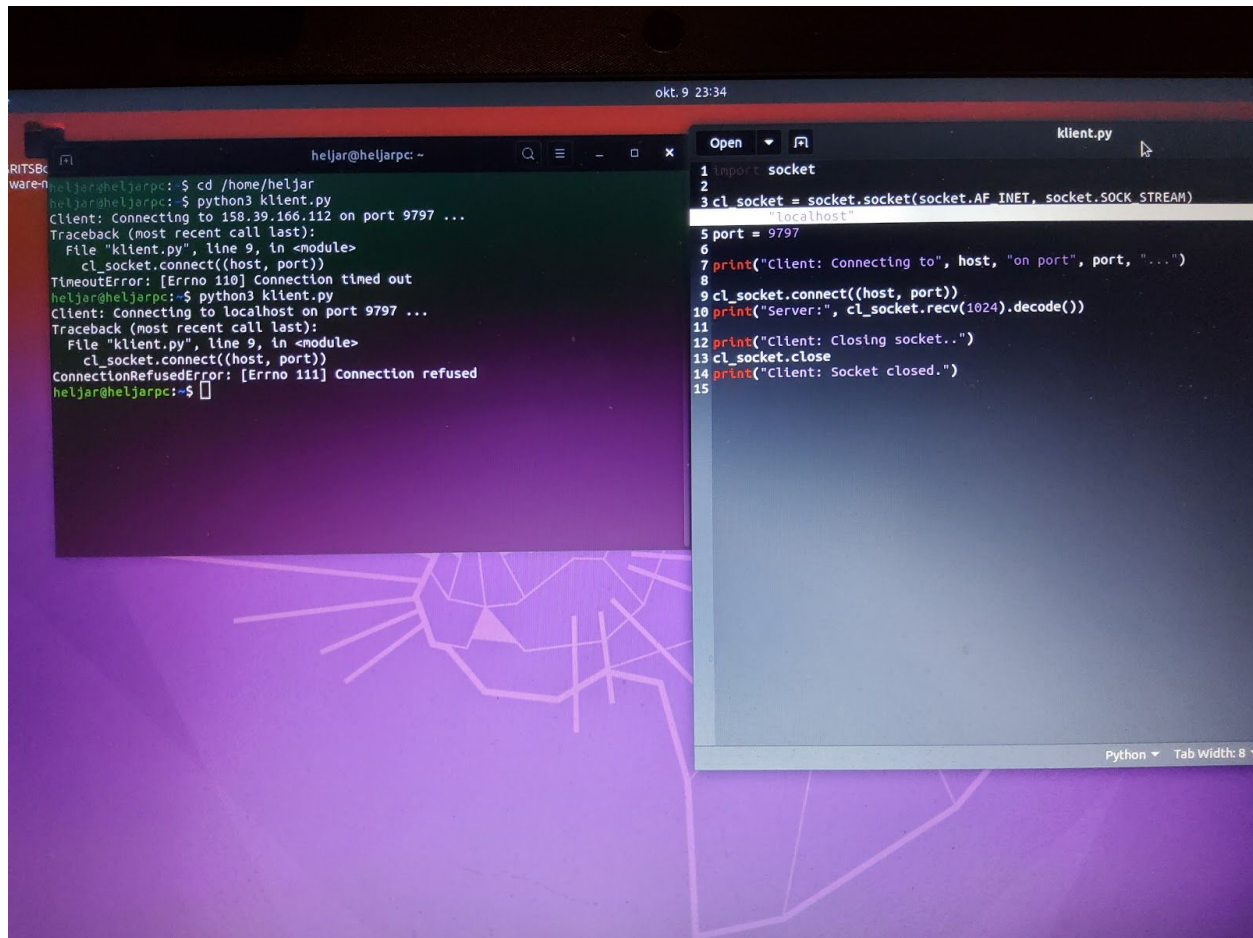


Så kjørte vi filen på klienten.

Terminal klient:

Cd /home/heljar

Python3 klient.py

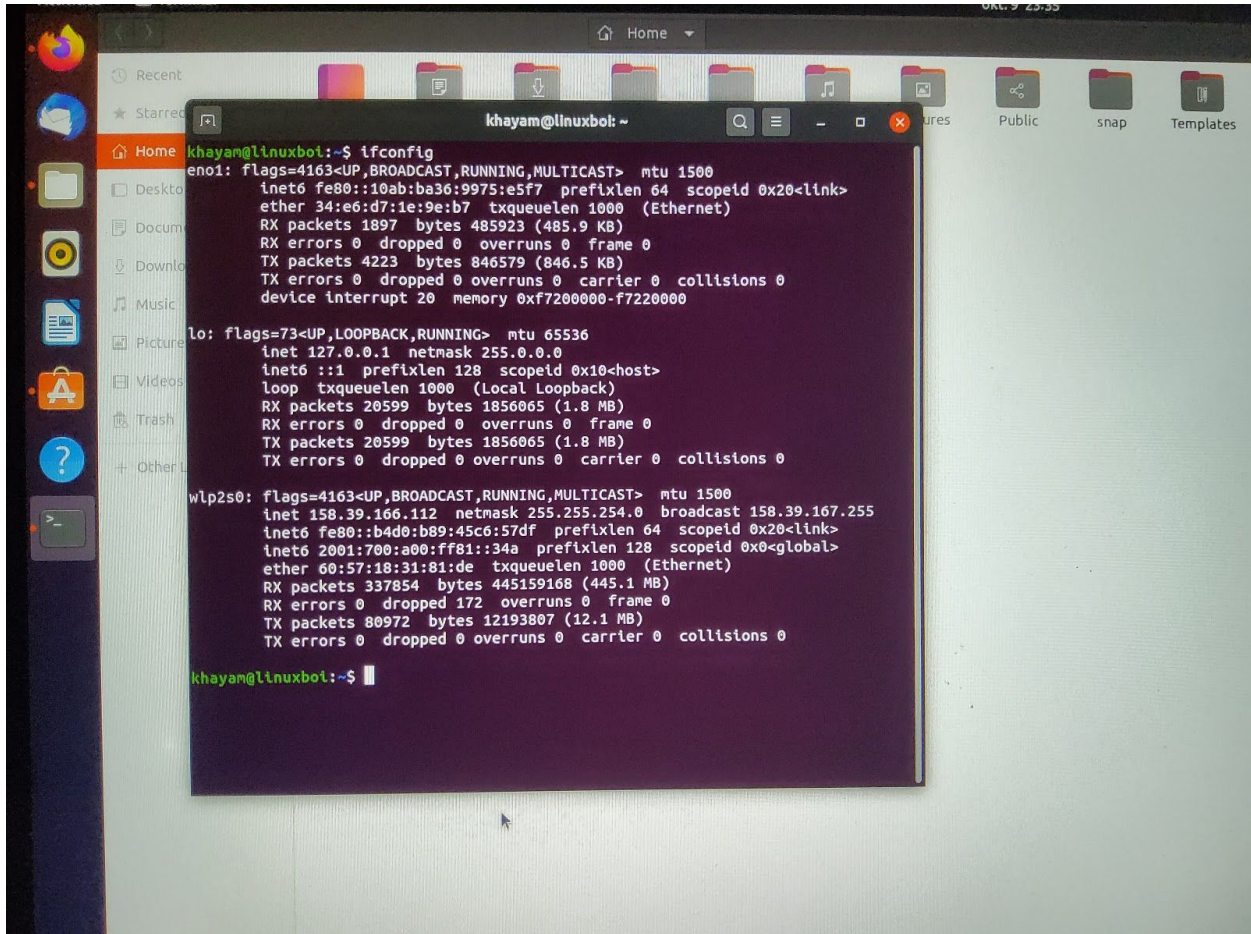


The screenshot shows a terminal window on the left and a code editor on the right. The terminal window displays the execution of a Python script named `klient.py`. The user first runs the script with the default host `localhost`, which results in a `TimeoutError: [Errno 110] Connection timed out`. After changing the host to `158.39.166.112`, the script runs again, but this time it results in a `ConnectionRefusedError: [Errno 111] Connection refused`. The code editor on the right shows the source code of `klient.py`, which is a simple socket client that connects to a server on `localhost` at port `9797`, prints the server's response, and then closes the socket.

```
heljar@heljarpc: ~  
heljar@heljarpc:~$ cd /home/heljar  
heljar@heljarpc:~/home/heljar$ python3 klient.py  
Client: Connecting to 158.39.166.112 on port 9797 ...  
Traceback (most recent call last):  
  File "klient.py", line 9, in <module>  
    cl_socket.connect((host, port))  
TimeoutError: [Errno 110] Connection timed out  
heljar@heljarpc:~$ python3 klient.py  
Client: Connecting to localhost on port 9797 ...  
Traceback (most recent call last):  
  File "klient.py", line 9, in <module>  
    cl_socket.connect((host, port))  
ConnectionRefusedError: [Errno 111] Connection refused  
heljar@heljarpc:~$
```

```
1 import socket  
2  
3 cl_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)  
4  
5 port = 9797  
6  
7 print("Client: Connecting to", host, "on port", port, "...")  
8  
9 cl_socket.connect((host, port))  
10 print("Server:", cl_socket.recv(1024).decode())  
11  
12 print("Client: Closing socket..")  
13 cl_socket.close()  
14 print("Client: Socket closed.")  
15
```

Her så fikk vi bare en timeout error etter ca. 3-4 minutter. Så vi gikk over filene og endret host fra "localhost" om til ip-adressen til tjeneren som vi fant ut via ifconfig i terminalen. Denne adressen var 158.39.166.112. Så vi gikk inn i begge filene og bytter "host = "localhost" om til "host = "158.39.166.112"



```
khayam@linuxbot: ~  
khayam@linuxbot:~$ ifconfig  
eno1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet6 fe80::10ab:ba36:9975:e5f7 prefixlen 64 scopeid 0x20<link>  
    ether 34:e6:d7:1e:9e:b7 txqueuelen 1000 (Ethernet)  
    RX packets 1897 bytes 485923 (485.9 KB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 4223 bytes 846579 (846.5 KB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
    device interrupt 20 memory 0xf7200000-f7220000  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 20599 bytes 1856065 (1.8 MB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 20599 bytes 1856065 (1.8 MB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
wlp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 158.39.166.112 netmask 255.255.254.0 broadcast 158.39.167.255  
    inet6 fe80::b4d0:b89:45c6:57df prefixlen 64 scopeid 0x20<link>  
    inet6 2001:700:a00:ff81::34a prefixlen 128 scopeid 0x0<global>  
    ether 60:57:18:31:81:de txqueuelen 1000 (Ethernet)  
    RX packets 337854 bytes 445159168 (445.1 MB)  
    RX errors 0 dropped 172 overruns 0 frame 0  
    TX packets 80972 bytes 12193807 (12.1 MB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
khayam@linuxbot:~$
```

Etter endringen så prøvde vi å kjøre programmene på nytt etter å ha avsluttet den tidligere. Vi fikk samme timeout error som fikk oss til å tenke at kanskje brannmuren blokkerte porten (9797) så vi sjekket brannmuren for å se om den blokkerte innkommende fra den porten. Så vi valgte skrive inn et unntak med "ufw allow 9797" for å tillate data gjennom den porten. Etter å ha gitt eksplisitt tilgang til denne porten i brannmuren så prøvde vi å kjøre python filene på nytt. Ikke noe flaks der. Fikk fortsatt samme timeout error.



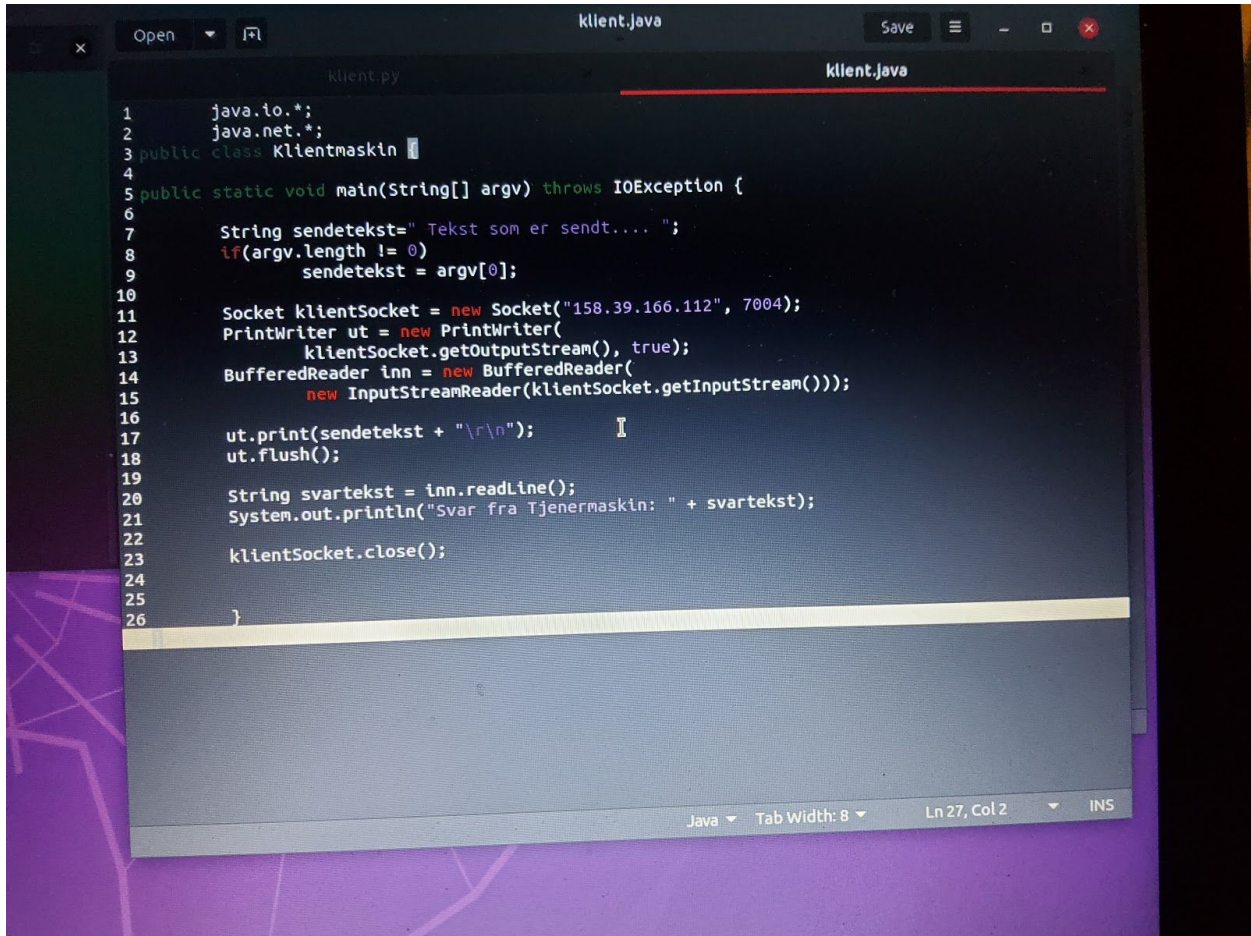
```
eth0: 00:17:18:31:81:0e txqueuelen 1000 (Ethernet)
RX packets 337854 bytes 445159168 (445.1 MB)
RX errors 0 dropped 172 overruns 0 frame 0
TX packets 80972 bytes 12193807 (12.1 MB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

khayam@linuxbol:~$ ufw status
ERROR: You need to be root to run this script
khayam@linuxbol:~$ sudo -i
[sudo] password for khayam:
root@linuxbol:~# ufw status
Status: active

To                        Action      From
--                        -
22/tcp                    ALLOW      Anywhere
9797/tcp                   ALLOW      Anywhere
9797                       ALLOW      Anywhere
22/tcp (v6)               ALLOW      Anywhere (v6)
9797/tcp (v6)             ALLOW      Anywhere (v6)
9797 (v6)                 ALLOW      Anywhere (v6)

root@linuxbol:~#
```

Etter å ha prøvd oss på python filene og ikke fått noe suksess der så prøvde vi oss på Java programmene hvor vi endret adressen i filene til tjeneren sin adresse, men første måtte vi installere java. Etter å ha installert den så prøvde vi å kjøre dem, men det gikk ikke heller og vi fikk fortsatt en timeout error fra denne også.



```
1  java.io.*;
2  java.net.*;
3  public class Klientmaskin {
4
5  public static void main(String[] argv) throws IOException {
6
7      String sendetekst=" Tekst som er sendt.... ";
8      if(argv.length != 0)
9          sendetekst = argv[0];
10
11     Socket klientSocket = new Socket("158.39.166.112", 7004);
12     PrintWriter ut = new PrintWriter(
13         klientSocket.getOutputStream(), true);
14     BufferedReader inn = new BufferedReader(
15         new InputStreamReader(klientSocket.getInputStream()));
16
17     ut.print(sendetekst + "\r\n");
18     ut.flush();
19
20     String svartekst = inn.readLine();
21     System.out.println("Svar fra Tjenermaskin: " + svartekst);
22
23     klientSocket.close();
24
25
26 }
```

Etter å ha prøvd så og si alt vi kunne tenke oss så prøvde vi å få hjelp via Discord serveren "Norsk Programmering" hvor vi forklarte situasjonen vår og fant ut at vi trengte en ruter eller mer riktig sagt tilgang til en ruter ettersom at vi ikke var på samme intranett. Noe som hadde vært bli informert om i oppgaveteksten.

Etter dette så prøvde vi å brainstorme hvilke muligheter vi hadde og de eneste måtene vi kunne tenke oss var å bruke vpn'er, men ettersom at vpn'en til skolen ikke gir oss mulighet til port forwarding og de private vi hadde fra før (CyberGhostVPN) ikke gir mulighet for port forwarding så kom vi fram til at vi ikke har det nødvendige utstyret til å utføre oppgaven.

Vi fikk kort tid før innleveringen vite at som studenter på IT ved hiof så har vi tilgang til Azure, men ettersom at vi ikke har hatt noe opplæring i dette i faget så vet vi ikke hvordan vi skulle ha satt det opp selv i en hypotetisk situasjon hvor vi muligens hadde hatt mer tid.

---

Hvordan vi ville gjort det i teorien om vi hadde det tilstrekkelige utstyret...

Oppgave 1 og 2:

Først så tester vi programmene. Så hadde vi fått en error om at de ikke kunne koble seg ettersom som at disse to fysiske maskinene ikke er på samme fysiske enhet. Mest sannsynlig så ville vi fått en timeout error fra klienten sin siden på grunn av at denne ikke finner tjeneren.

Original kode til tjener:

```
import socket

host = "localhost"
port = 9797

serv_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
serv_socket.bind((host, port))

serv_socket.listen()
print("Server: Listening on port " + str(port))

while True:
    cl_socket, address = serv_socket.accept()
    print("Server: Connection established with", address)

    message = "Connection accepted, closing.."
    cl_socket.send(message.encode())
    print("Server: Closing socket...")
    cl_socket.close()
    print("Server: Socket closed.")
```

Endret kode til tjener:

```
import socket

host = "158.39.166.112"
port = 9797

serv_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
serv_socket.bind((host, port))

serv_socket.listen()
print("Server: Listening on port " + str(port))

while True:
    cl_socket, address = serv_socket.accept()
    print("Server: Connection established with", address)
```

```
message = "Connection accepted, closing.."
cl_socket.send(message.encode())
print("Server: Closing socket...")
cl_socket.close()
print("Server: Socket closed.")
```

Original kode til klient:

```
import socket

cl_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
host = "localhost"
port = 9797

print("Client: Connecting to", host, "on port", port, "...")

cl_socket.connect((host, port))
print("Server:", cl_socket.recv(1024).decode())

print("Client: Closing socket..")
cl_socket.close()
print("Client: Socket closed.")
```

Endret kode til klient:

```
import socket

cl_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
host = "158.39.166.112"
port = 9797

print("Client: Connecting to", host, "on port", port, "...")

cl_socket.connect((host, port))
print("Server:", cl_socket.recv(1024).decode())

print("Client: Closing socket..")
cl_socket.close()
print("Client: Socket closed.")
```



## Oppgave 3:

Det er ikke mye å si her når det kommer til å gjøre sånn at en melding blir sendt begge veier utenom å bare legge til en message i klienten ettersom at tjeneren har allerede en message som blir sendt.

Endret kode fra klienten:

```
import socket

cl_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
host = "158.39.166.112"
port = 9797

print("Client: Connecting to", host, "on port", port, "...")

cl_socket.connect((host, port))

message = "Fish"

print("Server:", cl_socket.recv(1024).decode())

print("Client: Closing socket..")
cl_socket.close
print("Client: Socket closed.")
```

Endret kode til tjener:

```
import socket

host = "158.39.166.112"
port = 9797

serv_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
serv_socket.bind((host, port))

serv_socket.listen()
print("Server: Listening on port " + str(port))

while True:
    cl_socket, address = serv_socket.accept()
    print("Server: Connection established with", address)

    message = "Connection accepted, closing.."
```

```
cl_socket.send(message.encode())

print("Server:", cl_socket.recv(1024).decode())

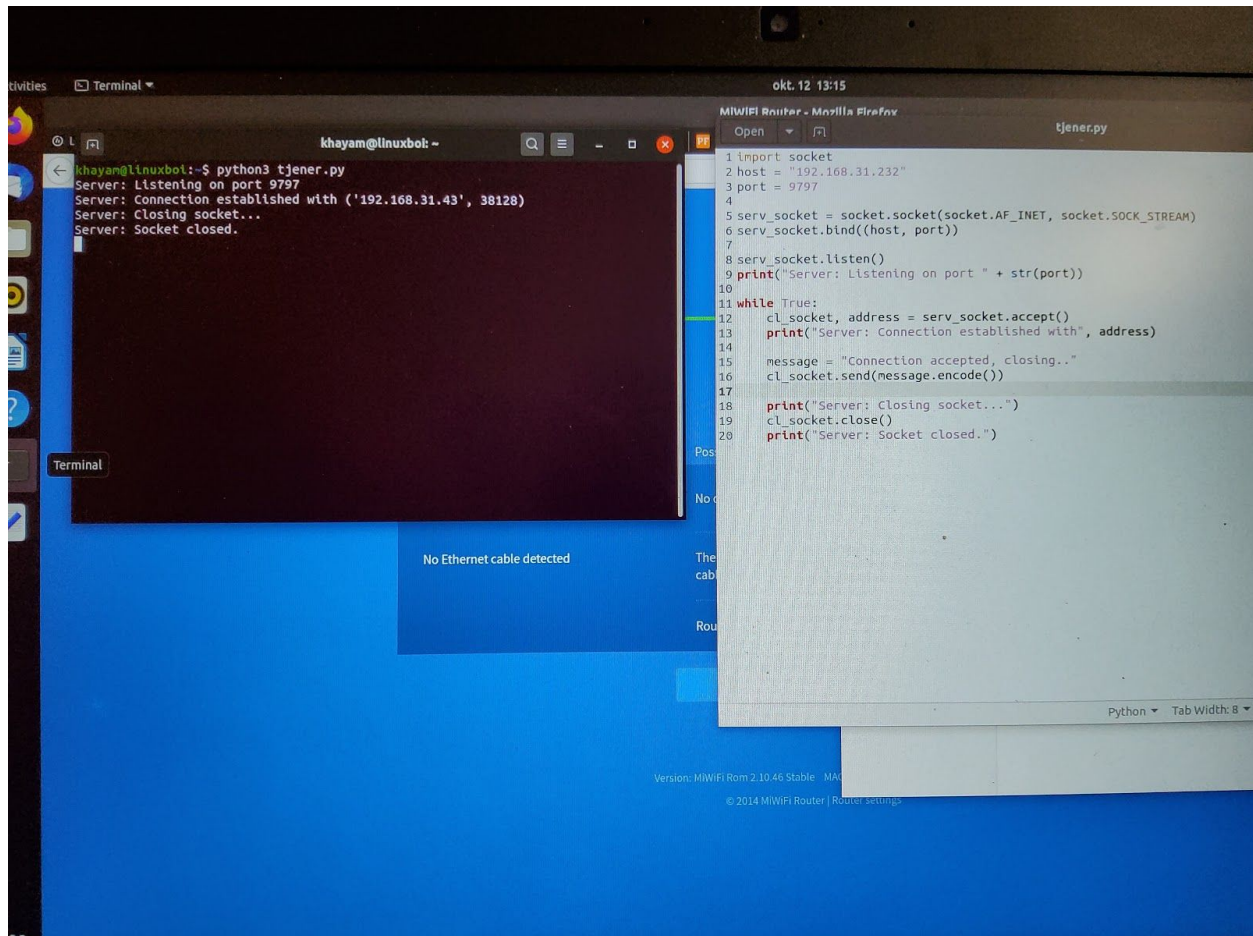
print("Server: Closing socket...")
cl_socket.close()
print("Server: Socket closed.")
```

### Bilder etter at vi har blitt tildelt utstyr fra skolen:

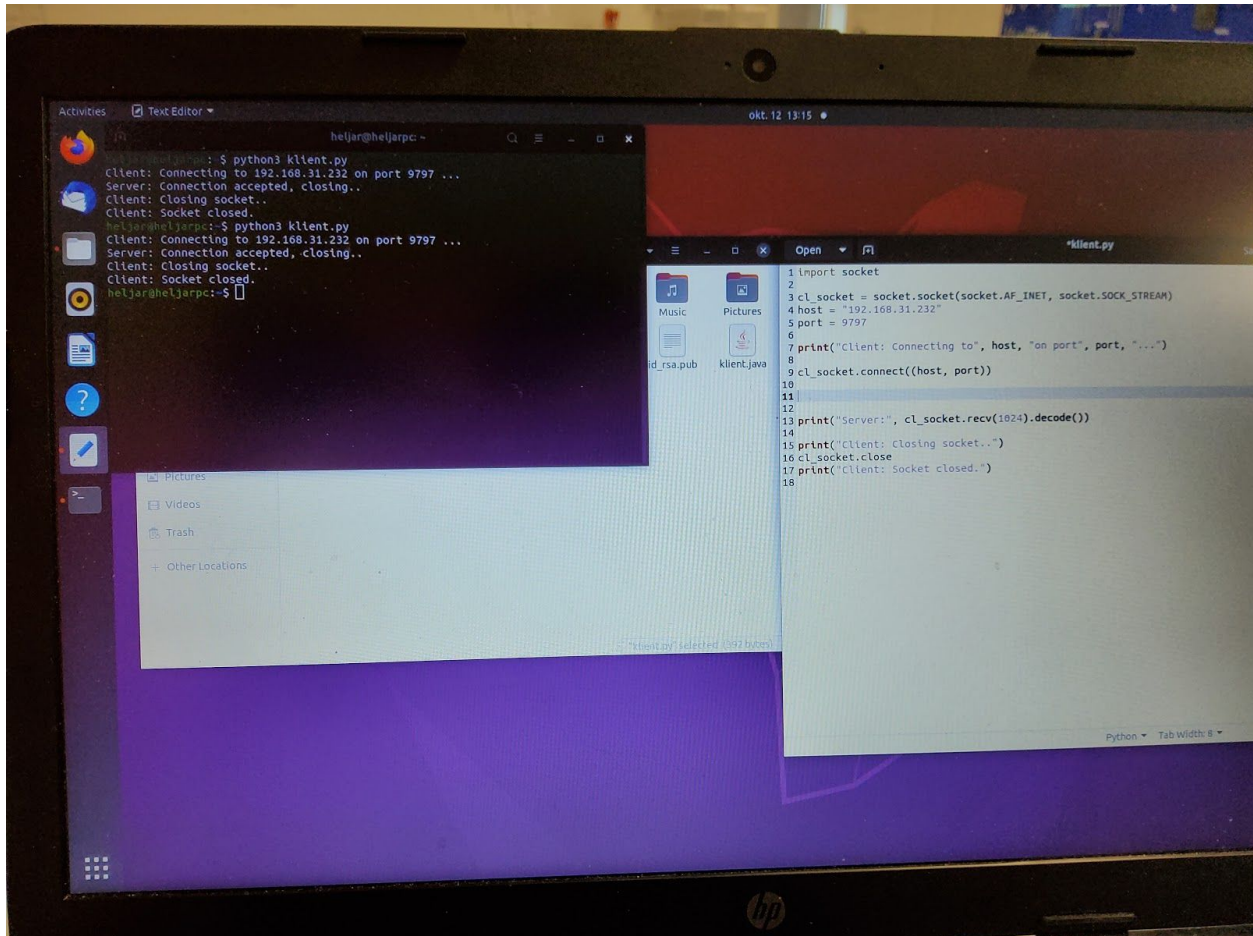
#### Oppgave 2

I denne delen av oppgaven så var det bare å endre ip adressen på " host = "localhost" " om til " host = "192.168.31.232" " på filen til både tjener og klient.

Tjener:

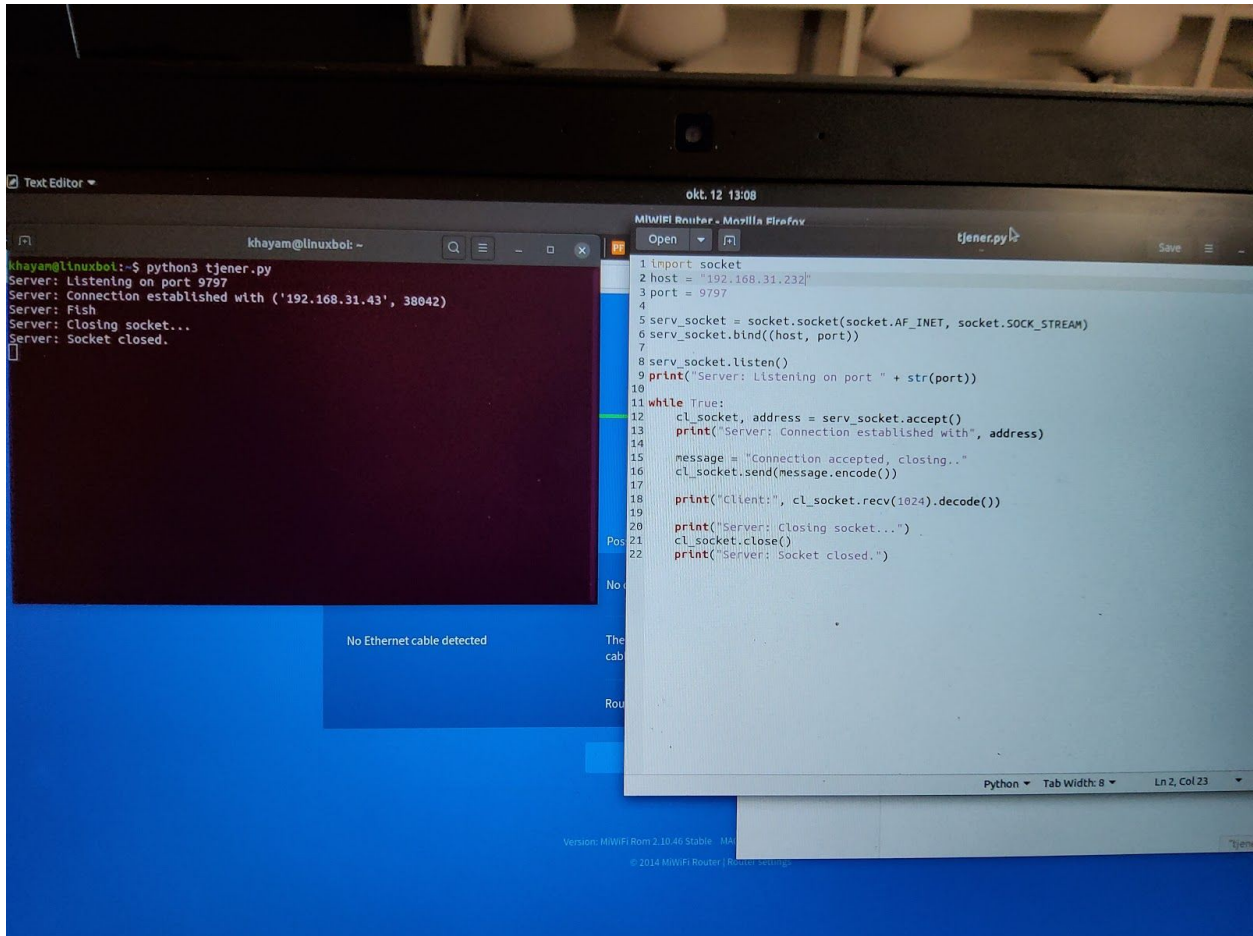


Klient:



### Oppgave 3

Tjener:



The screenshot shows a computer screen with a terminal window and a code editor. The terminal window, titled 'khayam@linuxbot: ~', shows the execution of a Python script 'tjener.py'. The output is as follows:

```
khayam@linuxbot:~$ python3 tjener.py
Server: Listening on port 9797
Server: Connection established with ('192.168.31.43', 38042)
Server: Fish
Server: Closing socket...
Server: Socket closed.
```

The code editor, titled 'tjener.py', shows the source code of the script:

```
1 import socket
2 host = "192.168.31.232"
3 port = 9797
4
5 serv_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
6 serv_socket.bind((host, port))
7
8 serv_socket.listen()
9 print("Server: Listening on port " + str(port))
10
11 while True:
12     cl_socket, address = serv_socket.accept()
13     print("Server: Connection established with", address)
14
15     message = "Connection accepted, closing.."
16     cl_socket.send(message.encode())
17
18     print("Client:", cl_socket.recv(1024).decode())
19
20     print("Server: Closing socket...")
21     cl_socket.close()
22     print("Server: Socket closed.")
```

At the bottom of the screen, there is a blue banner with the text "No Ethernet cable detected".

Klient:



