

OBLIG 2 Datanettverk og skytjenester

Thomas Knutsen

Studentnummer:386108

Mail: thknu3595@oslomet.no

Oppgave 1

Kode for simpleserver:

```
import argparse
from socket import *

def send_http_request(server_ip, server_port, file_path):
    """Metode som sender request """
    # Opprett en TCP-klient
    client_socket = socket(AF_INET, SOCK_STREAM)
    client_socket.connect((server_ip, server_port))

    # Bygger HTTP GET-forespørselen
    http_request = f"GET {file_path} HTTP/1.1\r\nHost: {server_ip}\r\nConnection: close\r\n\r\n"
    client_socket.send(http_request.encode())

    # Mottar og skriver ut responsen
    response = b""
    while True:
        data = client_socket.recv(1024)
        if not data:
            break
        response += data

    print(response.decode())

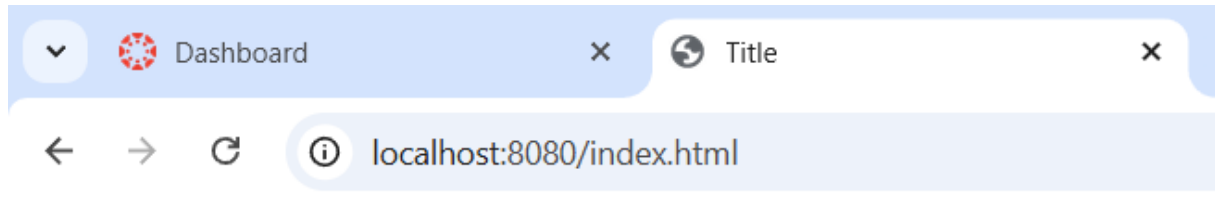
    client_socket.close()

if __name__ == "__main__":
    # Definer kommandolinjeargumenter
    parser = argparse.ArgumentParser(description="Simple HTTP Client")
    parser.add_argument("-i", "--ip", required=True, help="Server IP address")
    parser.add_argument("-p", "--port", type=int, required=True, help="Server port")
    parser.add_argument("-f", "--file", required=True, help="Filename or path to request")

    # Parse argumentene
    args = parser.parse_args()
```

```
# Kjør HTTP-forespørselen
send_http_request(args.ip, args.port, args.file)
```

Screenshots av at det fungerer :



Velkommen til min HTTP server

Oppgave 2

Kode for Klienten:

```
import argparse
from socket import *

def send_http_request(server_ip, server_port, file_path):
    """Metode som sender request """
    # Opprett en TCP-klient
    client_socket = socket(AF_INET, SOCK_STREAM)
    client_socket.connect((server_ip, server_port))

    # Bygger HTTP GET-forespørselen
    http_request = f"GET {file_path} HTTP/1.1\r\nHost: {server_ip}\r\nConnection: close\r\n\r\n"
    client_socket.send(http_request.encode())

    # Mottar og skriver ut responsen
    response = b""
    while True:
        data = client_socket.recv(1024)
        if not data:
            break
        response += data

    print(response.decode())

    client_socket.close()
```

```

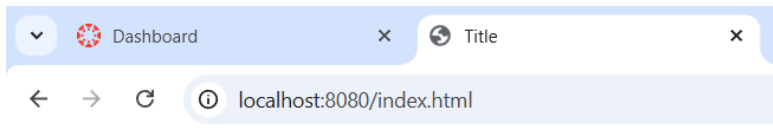
if __name__ == "__main__":
    # Definer kommandolinjeargumenter
    parser = argparse.ArgumentParser(description="Simple HTTP Client")
    parser.add_argument("-i", "--ip", required=True, help="Server IP address")
    parser.add_argument("-p", "--port", type=int, required=True, help="Server port")
    parser.add_argument("-f", "--file", required=True, help="Filename or path to request")

    # Parse argumentene
    args = parser.parse_args()

    # Kjør HTTP-forespørselen
    send_http_request(args.ip, args.port, args.file)

```

Screenshots av at det fungerer:



Velkommen til min HTTP server

For oppgave 2

```

(venv) PS C:\Users\oyvin\OneDrive - OsloMet\oblig2datanet> python CLIENT.py -i 127.0.0.1 -p 8080 -f /index.html
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 198

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>

<h1>Velkommen til min HTTP server</h1>
<h2>For oppgave 2</h2>

</body>
</html>
(venv) PS C:\Users\oyvin\OneDrive - OsloMet\oblig2datanet>

```

```

(venv) PS C:\Users\oyvin\OneDrive - OsloMet\oblig2datanet> python CLIENT.py -i 127.0.0.1 -p 8080 -f /thomas.html
HTTP/1.1 404 Not Found
Content-Type: text/html

<html><body><h1>404 Not Found</h1></body></html>
(venv) PS C:\Users\oyvin\OneDrive - OsloMet\oblig2datanet>

```

Oppgave 3

Kode for multithreaded server:

```
from socket import AF_INET, socket, SOCK_STREAM
import os
import threading

def handle_request(conn_sd):
    """Behandler en HTTP-forespørsel og sender et svar tilbake."""
    try:
        request = conn_sd.recv(1024).decode()
        if not request:
            conn_sd.close()
            return

        print("HTTP Request:\n", request)

        request_line = request.split("\n")[0] # Første linje: "GET /index.html HTTP/1.1"
        parts = request_line.split()

        if len(parts) < 2 or parts[0] != "GET":
            conn_sd.close()
            return

        filename = parts[1][1:] if parts[1] != "/" else "index.html"

        if os.path.exists(filename):
            with open(filename, "rb") as file:
                content = file.read()
            response_headers = (
                "HTTP/1.1 200 OK\r\n"
                "Content-Type: text/html\r\n"
                f"Content-Length: {len(content)}\r\n"
                "\r\n"
            )
            conn_sd.sendall(response_headers.encode() + content)
        else:
            response = (
                "HTTP/1.1 404 Not Found\r\n"
                "Content-Type: text/html\r\n"
                "\r\n"
                "<html><body><h1>404 Not Found</h1></body></html>"
            )
            conn_sd.sendall(response.encode())
    finally:
        conn_sd.close()

def main():
```

```

"""Starter en multithreaded HTTP-server."""
server_sd = socket(AF_INET, SOCK_STREAM)
port = 8080
server_ip = '127.0.0.1'
server_sd.bind((server_ip, port))
server_sd.listen(5) # Lytt på opptil 5 samtidige forbindelser
print(f"Server kjører på http://{server_ip}:{port}")

while True:
    conn_sd, addr = server_sd.accept()
    print(f"Tilkobling fra {addr}")
    client_thread = threading.Thread(target=handle_request, args=(conn_sd,))
    client_thread.start()

if __name__ == "__main__":
    main()

```

Oppgave 4

Screenshots av at det fungerer i Mininet:

```

vboxuser@ubuntu:~/oppgave4$ python3 CLIENT.py -i 127.0.0.1 -p 8080 -f /index.html
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 186

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Title</title>
</head>
<body>

<h1>Velkommen til min HTTP server</h1>
<h2>For oppgave 2</h2>

</body>
</html>
vboxuser@ubuntu:~/oppgave4$ python3 CLIENT.py -i 127.0.0.1 -p 8080 -f /thomas.html
HTTP/1.1 404 Not Found
Content-Type: text/html

<html><body><h1>404 Not Found</h1></body></html>
vboxuser@ubuntu:~/oppgave4$ S

```

```

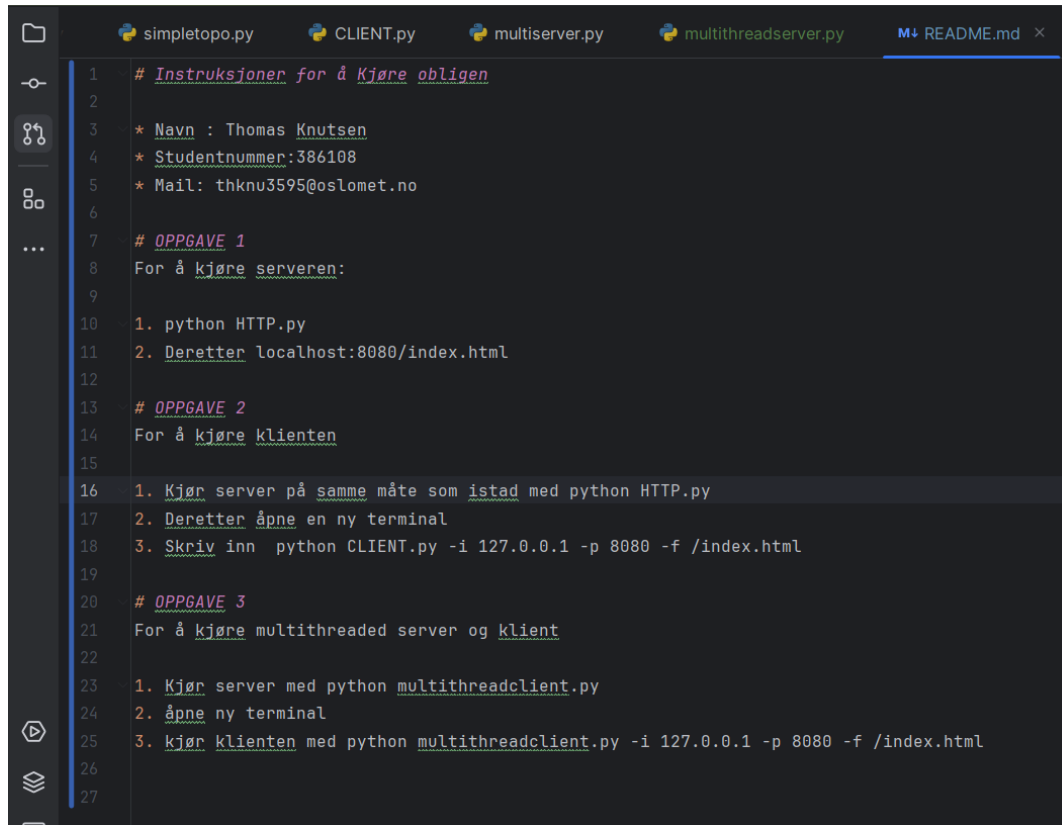
vboxuser@ubuntu:~/oppgave4$ ls
CLIENT.py  HTTP.py  index.html  multiclient.py  multiserver.py  oblig2datanet.inl  README.md  simpletopo.py
vboxuser@ubuntu:~/oppgave4$ python3 HTTP.py
Server kjører på http://127.0.0.1:8080/
Tilkobling fra ('127.0.0.1', 41344)
HTTP Request:
  GET /index.html HTTP/1.1
Host: 127.0.0.1
Connection: close

Tilkobling fra ('127.0.0.1', 39330)
HTTP Request:
  GET /thomas.html HTTP/1.1
Host: 127.0.0.1
Connection: close

```

Oppgave 5

README.md filen med instruksjonene:

A screenshot of a code editor window with a dark theme. The editor has several tabs at the top: 'simpletopo.py', 'CLIENT.py', 'multiserver.py', 'multithreadserver.py', and 'M+ README.md'. The 'README.md' tab is active. The code is written in a light-colored font on a dark background. It includes a header section with instructions, followed by three numbered tasks (OPPGAVE 1, 2, and 3) with their respective steps. The code is as follows:

```
1 # Instruksjoner for å Kjøre obligen
2
3 * Navn : Thomas Knutsen
4 * Studentnummer:386108
5 * Mail: thknu3595@oslomet.no
6
7 # OPPGAVE 1
8 For å kjøre serveren:
9
10 1. python HTTP.py
11 2. Deretter localhost:8080/index.html
12
13 # OPPGAVE 2
14 For å kjøre klienten
15
16 1. Kjør server på samme måte som istad med python HTTP.py
17 2. Deretter åpne en ny terminal
18 3. Skriv inn python CLIENT.py -i 127.0.0.1 -p 8080 -f /index.html
19
20 # OPPGAVE 3
21 For å kjøre multithreaded server og klient
22
23 1. Kjør server med python multithreadclient.py
24 2. åpne ny terminal
25 3. kjør klienten med python multithreadclient.py -i 127.0.0.1 -p 8080 -f /index.html
26
27
```

README.md fil

Instruksjoner for å Kjøre obligen

* Navn : Thomas Knutsen
* Studentnummer:386108
* Mail: thknu3595@oslomet.no

OPPGAVE 1

For å kjøre serveren:

1. python HTTP.py
2. Deretter localhost:8080/index.html

OPPGAVE 2

For å kjøre klienten

1. Kjør server på samme måte som istad med python HTTP.py
2. Deretter åpne en ny terminal
3. Skriv inn python CLIENT.py -i 127.0.0.1 -p 8080 -f /index.html

OPPGAVE 3

For å kjøre multithreaded server og klient

1. Kjør server med `python multithreadclient.py`
2. åpne ny terminal
3. kjør klienten med `python multithreadclient.py -i 127.0.0.1 -p 8080 -f /index.html`