Norges teknisk-naturvitenskapelige universitet
Institutt for informasjonssikkerhet og kommunikasjonsteknologi
**TTM4100 – Kommunikasjon – Tjenester og nett**

# Praksisøving 2

## HTTP Web Server

Denne praksisøvingen er én av seks praksisøvinger i emnet. Du må levere og få godkjent minst fire av disse praksisøvingene, i tillegg til fem av åtte teorivinger, for å kunne gå opp til eksamen. Hvis du lurer på noe angående øvingen kan du stille spørsmål på Piazza eller få veiledning av læringsassistent. Se emnesiden på Blackboard for tidspunkter og fremgangsmåte.

Oppgavene er ment å fokusere på læring og forståelse, og ikke debugging av kode. Du kan velge om du vil svare på norsk eller engelsk.

Les oppgaven og kildekoden nøye. Det anbefales på det sterkeste at du leser og forstår kapittel 2.7.2 i pensumboken før du starter øvingen.

Lever svarene dine på Blackboard innen fristen. IKKE lever som .zip-mappe, men last opp filene én og én.

Innleveringsfrist: **søndag 18. februar 2024, kl. 23:59.**

The goal of this exercise is to understand how the application code sends and receives data through a socket. You will learn the basics of socket programming for TCP connections in Python: how to create a socket, bind it to a specific address and port, as well as send and receive a HTTP packet. You will also learn some basics of HTTP header format.

You will develop a web server that handles one HTTP request at a time. Your web server should accept and parse the HTTP request, get the requested file from the server's file system, create an HTTP response message consisting of the requested file preceded by header lines, and then send the response directly to the client. If the requested file is not present on the server, the server should send an HTTP "404 Not Found" message back to the client.

**N.B.** It is strongly recommended to read **Sec. 2.7.2** of the textbook before (and while) doing this programming lab.

# Code

You will find the skeleton code for the Web server in the file "Skeleton_WebServer.py". You are to complete the skeleton code with any Python version of your choice. Beware there are some syntax differences in Python 2.7 and Python 3. The places where you need to fill in code are marked with `#FILL IN START` and `#FILL IN END`. Each place may require one or more lines of code. The comments in the skeleton explain what the code you are to fill in is supposed to do.

Relevant commands to understand are:

- .socket()
- .bind()
- .listen()
- .recv()
- .send()
- .close()

# Running the Server

Put an HTML file (e.g., HelloWorld.html) in the same directory as the server file. If you do not know how to create an HTML file, there are many examples on the web.

Run the server program. Determine the IP address of the host that is running the server (e.g., 128.238.251.26). You can do this locally by using the default localhost

IP (127.0.0.1). From another host, open a browser and provide the corresponding URL. For example:

http://128.238.251.26:6789/HelloWorld.html
or
localhost:6789/HelloWorld.html

'HelloWorld.html' is the name of the HTML file you placed in the server directory. Note the use of the port number after the colon. You need to replace this port number with whatever port you have used in your server code. In the above example, we have used port number 6789. The browser should then display the contents of HelloWorld.html. If you omit ":6789", the browser will assume port 80 and you will get the web page from the server only if your server is listening to port 80.

Then try to get a file that is not present in the same directory as the server file. You should get a "404 Not Found" message.

**Note:** The server does not handle termination of the TCP connection. This leads to an IndexError and a subsequent ConnectionAbortedError. It is perfectly normal (for the server in this assignment).

**Tips:** If you cannot connect to your HTTP server from a remote computer, make sure your two computers are in the same network, or try disabling the firewall on the server computer.

# What to Hand in

- The completed server code.
- A screenshot of your client browser displaying the contents of your HTML file, accessed through the running server. Make sure to display the browser address bar, showing your port number and HTML filename.
- A screenshot of your client browser displaying the error page after attempting to access a file not located in the server code file directory.

  Upload the code and screenshots to Blackboard directly, and do NOT put them in an archive (like e.g. .zip). Archived deliveries will not be approved.