Assignment Four

Webserver

Set: *22nd of December 2022*
Due: *7th of January 2023 @ 23:55 CEST*

Synopsis:
Implement a HTTP compatible webserver that can serve static files to a browser.

Introduction

This is the fourth of five assignments in the *High Performance Parallel Systems* course. The four assignments are practical in nature, and will give you practical experience in the topics, as well as a deeper understanding.

Implementation

You must implement a web server that is capable of serving files from a local folder. This means that you are only required to implement the HTTP GET method, but you may want to implement the HEAD method as well, as it may be useful for testing. The server should be able to run indefinately, by which it is meant that it should not just process a given number of responses and then stop.

The server should work as a plain file server and should accept a path to a folder and a port number. The server may also take an address to listen on, but if this is not implemented, the server should listen on all addresses on the port specified. Files and folders in specified path should be served up by the webserver. If a folder on the server is accessed (with a URL ending in a `/`) then you should serve up the contents of file named `index.html`, if one exists, or a listing of the files and folders, if `index.html` does not exist.

When a request is received, it should be parsed and verified as a valid HTTP request. The URL from the request is then mapped to the underlying file system. If the requested file does not exist, the server responds with error code 404 and a short message, otherwise the file contents are returned. You will need to use other error codes as appropriate. Your server should be resilient, and so malformed messages should not cause it to crash and prevent it recieving future messages.

In general you should familiarize yourself with the HTTP RFC (RFC 2616), which explains how the protocol is supposed to be interpreted. You must implement support for at least the following headers:

- Host

- Accept
- Accept-Encoding
- Connection
- If-Modified-Since
- If-Unmodified-Since
- User-Agent

Be aware that some of these headers are very large in scape, and you should by no means attempt to support all related features. For instance, the "Accept-Encoding" header can specify a variety of different encryption algorithms but you just need to demonstrate a single one to say it is supported. It is up to you to determine simple ways of demonstrating some implementation of each header.

Note, you are expected to support all necessary Response headers so if you request content is encrypted then your response should contain the appropriate headers.

Your web server should be implemented in Python. Your implementation *MUST* use socket programming, meaning that the use of an HTTP aware library is not allowed. You can use other libraries, e.g., text parser, regular expressions, url parser, etc. It is sufficient if your server can only serve a single request at a time (i.e. you do not need to implement multithreading).

You will need some client to implement/test your server. You are free to use whatever you wish, such as an internet browser, or a python implementation. No requirements have been made on this side of the interaction, so the client can be as complex or as simple as you deem necessary to correctly test your server implementation.

Handed-out Code

Note that no code is explicitly provided as part of the handout. However, you are reminded that the exercise code from 22/12/22 is for a Python server that responds to request from a client and so make act as a good start for your own implementation.

Your Report

Your reports should contain:

- A description of your web server design (including libraries and frameworks used)
- A description of your web servers limitations
- A description of what tests you have performed and their outcomes
- A description of supported headers, and how you have met the requirements.

Deliverables for This Assignment

You are encouraged to work in groups with 3 people for this assignment. We strongly encourage you to participate in the lab sessions, where we will use time to discuss the design, implementation etc.

You should submit the following items:

- A single PDF file, A4 size, no more than 3 pages, in ACM format, describing each item from report section above
- A single ZIP/tbz2 file with all code relevant to the implementation. If you have created any test scripts, or additional files these should be included.

Handing In Your Assignment

You will be handing this assignment in using Absalon. Try not to hand in your files at the very last-minute, in case the rest of the students stage a DDoS attack on Absalon at the exact moment you are trying to submit. **Do not email us your assignments unless we expressly ask you to do so**.

Assessment

You will get written qualitative feedback, and points from zero to 4. There are no resubmissions, so please hand in what you managed to develop, even if you have not solved the assignment completely. You will need a total of 8 points to qualify for the exam and at least 1 point from each assignment.