Project

Project Objective

- This project aims to develop a socket program to implement a Web service using the HTTP protocol.

Project Requirements

- In this project, each student is required to develop a multi-threaded Web server that is capable of processing HTTP requests sent from browsers or some other client programs. This multi-threaded program will be able to handle multiple requests at the same time. Specifically, your Web server will
  I. create a connection socket when contacted by a client (browser);
  II. receive the HTTP request from this connection;
  III. parse the request to determine the specific file being requested;
  IV. get the requested file from the server's file system;
  V. create an HTTP response message consisting of the requested file preceded by header lines;
  VI. send the response over the TCP connection to the requesting client. If the client requests a file that is not present in your server, your server should return a "404 Not Found" error message.

Project Requirements

- Your task is to implement the server program, run your server program, and then test your server program by sending requests from the client programs running on different hosts.
  - You may run the server on your own computer, using the IP address of 127.0.0.1.
  - If you run your server on a host that already has a Web server running on it, then you should use a different port than port 80 for your Web server.

Project Requirements

- You can develop your code in two stages.
  - In the first stage, you can simply implement the server program to receive the HTTP request messages and display the contents.
  - After this is running properly, you can add the code to generate appropriate responses in the second stage.
- The Web server needs a log file to record statistics of the client requests.
  - Each request corresponds to one line of record in the log.
  - Write down client hostname/IP address, access time, requested file name and response type for each record.
- Your Web server also needs to handle some simple errors, such as web-page not found.

Project Requirements

- You can use either Python, Java or C/C++ languages for the project.
- When implementing the Web server, you are expected to use basic socket programming classes to build the Web server from scratch instead of using the HTTPServer class directly.

What to submit?

You need to submit a project package to Learn@PolyU, containing the following documents:

- A project report that contains
  - A cover page includes your name and student number;
  - A summary of your design and implementation of the server program;
  - A demonstration of executing your program and screen capturing of results of all functions;
  - A log file that records the historical information about the client requests and server responses.
- Complete source code
  - Your code should be commented appropriately.
- A README text file of how to compile and run your program
- The due time of the project is 11:59pm, April 30, 2025, Sunday, determined by Learn@PolyU.
  - Do not challenge this time and submit your project package a little earlier.
  - Late submission will cause the marks deducted 25% per day.

How to assess your project?

Your work will be graded according to the following criteria:

- Design and implement the Web server program to support the following functions (70 marks)
  - Proper request and response message exchanges (5 marks)
  - GET command for both text files and image files (10 marks, 5 marks each)
  - HEAD command (5 marks)
  - Six types of response statuses, including 200 OK, 304 Not Modified, 400 Bad Request, 403 Forbidden, 404 File Not Found, 415 Unsupported Media Type (30 marks, 5 marks each)
  - Handle Last-Modified and If-Modified-Since header fields (10 marks)
  - Handle Connection header field for both HTTP persistent connection (keep-alive) and non-persistent connection (close) (10 marks, 5 marks each)

How to assess your project?

- Quality of your project’s report (25 marks)
  - A good summary of your design and implementation of the server program (10 marks)
  - A full demonstration of executing your program and screen capturing of results of all functions (5 marks)
  - A complete log file (5 marks)
  - A clear README text file (5 marks)
- Quality of your project’s source code (5 marks)
  - Complete source code for the project
  - Good naming and coding convention used in your source code
  - Compile the source code successfully
  - Execute the program without runtime errors

Marking sheet

Student Information	Name		Student ID	
TA Information				
Check List				Marks
Design and implement the Web server program (70 marks)				
○ Proper request and response message exchanges (5 marks)				
○ GET command for both text files and image files (10 marks)				
○ HEAD command (5 marks)				
○ Six types of response statuses (30 marks)				
○ Handle Last-Modified and If-Modified-Since header fields (10 marks)				
○ Handle Connection header field (10 bonus marks)				
Quality of your project's report (25 marks)				
○ A good summary of your design and implementation (10 marks)				
○ A full demonstration of executing your program and screen capturing of results of all functions (5 marks)				
○ A complete log file (5 marks)				
○ A clear READMET text file (5 marks)				
Quality of your project's source code (5 marks)				
○ Complete source code for the project ○ Good naming and coding convention used in your source code ○ Compile the source code successfully ○ Execute the program without runtime errors				
Total				