German University in Cairo
Faculty of Media Engineering and Technology
Amr El Mougy
Yasmeen Essam
Hana Medhat
Mariam Samy

DMET 602 Networks and Media Lab
Spring Term 2018
Project #2: Web server

1. Introduction

The objective of this project is to give you basic experience of developing a webserver application using socket programming. The projects allows you to create and process HTTP requests and create responses.

2. System Model

The system consists of one server hosting one web application. Any number of clients can access the server to request resources (GET). The server processes the requests and generates responses back to the client.

2.1 Connection Management

The web application uses **HTTP on TCP Port 80**. The server can accept **multiple simultaneous TCP requests from multiple clients**. The TCP connection management is **multi-threaded**, meaning that the server has a **listening socket** that accepts connection requests. Upon receiving a connection request, a **connection socket** will be opened using a threaded process. The connections are **persistent**, meaning that only the client can close the connection. No pipelining or parallel processing is needed. Once the client finishes downloading all resources, it will close the connection with the server.

2.2 Processing HTTP Requests

The server is always listening to all open connection sockets. Once an HTTP request is received, it joins a queue. This queue is simply a data structure that saves all received requests in the order in which they arrive. The server takes one HTTP requests from the head of the queue, processes it, generates a response, sends the response through the appropriate connection socket, then after all this is done it takes the following HTTP request from the queue.

The HTTP requests are generated by the client. They should consist of 4 lines:

```
Method  URL   Version
Host
Accepted format
Connection
```

The method can only be “GET”. GET is used to retrieve resources from the server. In this project, you are required to store 10 objects at the server: 4 different text files, 2 JPEG images, 2 PNG images, and 2 mp4 files. Thus, GET can be used to retrieve any of these resources. You are required to create a “docroot”

folder on your server, where all the stored objects will be available. The URL in the HTTP request is simply the path of the resource in the docroot folder (see lecture slides for more details).

In addition, the "Host" in the HTTP request is the name of your web application, which you choose yourself. The "accepted format" header specifies the format in which you want the object (txt, JPEG, etc.). The server should not send back a response unless it has the requested object in the requested format. Finally, the "connection" header specifies what the client wants the server to do with the TCP connection. This value should be "keep-alive" for all HTTP reqeusts, except for the last one requested by the client. In the last one only, it should be "close". Once the client generates an HTTP request with "Connection: Close", the server will close the connection directly after sending the HTTP resposne. The client will also close the connection after receiveing the HTTP response.

Once the HTTP request is received at the server and makes it to the head of the processing queue, the server will start the parsing process. Here, the objective is to parse the HTTP request into a data structure (see lecture slides for details). This data structure has entries for each header, plus their corresponding values. This will simplify the response generation later. After the parsing is done, the processing phase is over.

2.3 Generating the HTTP Response

After the HTTP request is parsed, the server will start the process of generating the response. The HTTP response should have 4 lines, plus the requested object (if found):

```
Status   Version
Timestamp
Format
Connection
```

The status can either be:

- 200 OK: if the HTTP request has been processed successfully (the object was found in the requested format).
- 404 Not Found: this should be inserted if the object has not been found in the requested format.

The "timestamp" header is the date and time when the response was created. In addition, the header "format" is the format of the object in the response. Finally, "connection" can either be keep-alive or close, depending on what was specified in the HTTP request.

The server performs the following steps to generate the HTTP response:

1. The server fetches the resource specified in the URL from the docroot.
2. The server generates an HTTP response, with the status line depending on the outcome of STEP 1.

The server then pushes the HTTP response through the appropriate connection socket. To do this last step, the server must keep track of which HTTP request is currently being processed.

German University in Cairo
Faculty of Media Engineering and Technology
Amr El Mougy
Yasmeen Essam
Hana Medhat
Mariam Samy

3. Submission Phases

You are required to develop the project through three major milestones.

3.1 Milestone I

Submission Date: Labs starting 27th of March

Deliverables:

1. Multi-threaded server capable of accepting many connection requests from many clients.
2. The server should store the list of open connections, with their proper identifiers (this doesn't necessarily have to be IP addresses, but you have to be able to uniquely identify each connection. You can use usernames if you like).
3. The server should be always listening for incoming connection requests from the clients. Once a request arrives, a connection socket should be open and saved on the list at the server.
4. Clients should be able to generate HTTP requests as specified before.
5. The client should send the HTTP request to the server through the appropriate connection socket.
6. The server receives the HTTP request and adds it into a queue (a data structure that you create).

Note that at this step you do not need to create a docroot at the server. Thus, you can add any dummy URL in the HTTP request, just for illustration.

3.2 Milestone II

Submission Date: Labs starting 3rd of April

Deliverables:

1. Create the docroot and store the specified 10 objects. Specify the URL of each one. Assume all URLs are known to all clients.
2. The server takes one request from the head of the queue to process it.
3. The request is parsed into a data structure.
4. Create HTTP response with appropriate status line.
5. Send response to the client through the appropriate socket.
6. A simple GUI should be designed to view the HTTP requests and responses at the client. The GUI should be initialized with the 10 objects that are stored on the server. The user should be able to click any object, and this would automatically create the corresponding HTTP request and send it to the server.

4. Grading and Notes

- You may use any parts of your code in the first project.

German University in Cairo
Faculty of Media Engineering and Technology
Amr El Mougy
Yasmeen Essam
Hana Medhat
Mariam Samy

- You may work individually or in teams of two. However, team members should be from the same tutorial.
- Every team member should be clearly responsible for certain tasks so that they are evaluated on these tasks only. If you choose to collaborate on all tasks, then both team members will get the same grade.