Programming Assignment 1 (HTTP and Web Server Lab)

Problem Statement

In this lab, you will learn the basics of socket programming using TCP connections in Python:

- how to create a TCP socket,
- how to bind it to a specific address and port,
- how to send and receive an HTTP packet.

You need to implement a simple client that sends an HTTP request at a time. You also need to develop a web server that handles the HTTP request.

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, and then send the response directly to the client. More specifically, if the requested file is present in the server's file system, your web server should send an `HTTP/1.1 200 OK` **AND** the `content` of the requested file to the client; if the requested file is not present in the server's file system, the server should send an HTTP `404 Not Found` message back to the client.

Your client should display **ALL** the messages received from the web server. In particular, your client should display all the content of the requested file if it is present in the server's file system.

You are provided is a skeleton code for the Web server. You are required to complete the skeleton code `server.py`. The places where you need to fill in code are marked with `???????????`. Each place may require one or more lines of code. However, you need to develop your client code from scratch. (**Note:** You may modify `server.py` any way you like, as long as it meets the above requirements. You may even ignore this sample code and start all over from scratch.)

You will also need some basics of HTTP header format in this programming assignment.

Run the Server/Client inside CSC361-VM

Within CSC361-VM, open a terminal and type `sudo mn -x` to create a simple network topology (more details about the created topology can be found in Mininet Walkthrough). Then pick h1 and h2 to run your server/client code, and start Wireshark to capture packets. Note that you should not open the Wireshark before `sudo mn -x` command.

How to Run the Server

Determine the IP address of the host that is running the server (e.g., the IP address of host h1). Then run a working version of your server program `server_web.py`. Note that you should run your server first. The following is an input command format to run the server.

```
python server_web.py <IP address of your server> <port number of your ser
ver>
```

For example, if your server is running over IP address A.B.C.D and port number 6789, then the command format can be

```
python server_web.py A.B.C.D 6789
```

How to Run the Client

Your client will connect to the server using a TCP connection, send an HTTP request to the server, and display the server response as an output. You can assume that the HTTP request sent is a GET method. The client should take command line arguments specifying the server IP address or host name, the port at which the server is listening, and the path at which the requested object is stored at the server. The following is an input command format to run the client.

```
python client_web.py <IP address of your server> <port # of your server>
<requested file name>
```

For example, if the client wants to request `hello.html` file, the command format can be

```
python client_web.py A.B.C.D 6789 hello.html
```

Evaluation

You must demonstrate your two pieces of code inside the `CSC361-VM` and use **Wireshark** to capture all HTTP related packets. Your server and client must use **different IP addresses**, i.e., you are not allowed to use `127.0.0.x`.

Put the provided `hello.html` file in the same folder as where your server is and test your HTTP client/server. In the test case that your client requests `hello.html` file, your client is supposed to see `HTTP/1.1 200 OK` and the **content** of the request file; In the test case that your client requests a random file, e.g., `random.html`, your client is supposed to see `404 NOT FOUND`.

You are required to create **one single document** before your demo that has

- The screen shot of all your HTTP packets captured in Wireshark
- The screen shot of your client (two test cases)

You are **also** required to answer a few questions about your programming assignment during the demo in ECS 360.

The evaluation scheme is:

- (50%) Correctness of Python code
- (20%) The created document before your demo
- (30%) Questions and Answers during the demo (at least 5 questions)

What to Hand in

You will hand in the **complete client/server code** along with the **single document created** to Connex before 19:00, Jan 31 2020.