CY350X - Computer Networks

Project #2 – Web Server Using http.server

In this programming assignment, you will build a small web server that accepts connections from a web browser and responds to GET requests. If the requested file does not exist, your server must respond as directed. Additionally, if the request contains an "If-Modified-Since" header, you must respond appropriately. All responses from your server must supply the properly formatted HTTP headers as described below, where relevant. You will practice what you've learned about HTTP and programming with sockets. You will code in **Python3** and your final code must execute on your CY350X VM.

Checkpoint due 1 Mar 2023 at 2359.

Completed project due 30 Mar 2023 at 2359.

Requirements

1. Create a directory called "pr2" in the home directory of your VM. This is where your server will run. Both Firefox and Safari will prevent users from requesting files in the parent directory of your server root directory, so you don't have to worry about users exploiting your server to view your home directory contents.
2. Your code will process GET requests from web browsers and respond appropriately.
3. If the GET request includes an "If-Modified-Since" header:
    a. If the requested object exists and has not been modified since the date specified in the request header, respond with a "304 Not Modified" and include the same headers you would for a "200 OK" response.
    b. If the object exists but has been modified, then respond with the appropriate "200 OK" response and object, if required.
    c. If the object does not exist, respond with a "404 Not Found".
4. If the requested object exists, your server will:
    a. Respond with the "200 OK" response code
    b. Include accurate Last Modified header
    c. Indicate that the server does not accept persistent connections
    d. Properly deliver the requested file (remember that a Content Length header and, for some files, one other header will be required for this).
5. If the requested file does not exist, your server must determine if it has moved. There will be a text file called "moved" in your root pr2 directory that is a list of file names and their new locations. If the file has moved, your server must respond with the "301 Moved Permanently" response code and supply the new location in the response header. Use Wireshark on the instructor's example webpage to see how this works.
6. If the requested file does not exist your server must supply a "404 Not Found" response code.
7. You must use the *http.server* module in Python for your server implementation, creating a subclass of *http.server.BaseHTTPRequestHandler*. You may use any other non-network modules you need. If you have a question about whether a specific module is allowed, ask your instructor.

8. Once your webserver is up and running, design a basic webpage that includes text, picture(s), a video, and at least one JavaScript object (must be different than the provided JavaScript object in the provided base HTML file). Configure your server so that the URL http://<You>.cy350.eecs.net/index.html serves your webpage (there is already a DNS record that associates that URL with your VM's IP address). **Up to five extra credit points will be awarded for style, originality, and creativity.**

Submissions

You will electronically submit the code for your webserver as one file, named <you>_pr2.py, to Canvas. Use in-line code documentation where applicable. Additionally, you must attach a digital coversheet to your submission. You must also ensure that there is a working server up and running on your VM in the /home/cadet/pr2 directory, as I will be connecting to your server to grade it.

Along with your code, you will submit a one-page analysis of your design philosophy, problems that you encountered during implementation, and lessons learned.

For the checkpoint, you will submit a document that contains:

- A list of modules you intend to use in addition to the *http.server* module
- The HTTP response header fields your server will be supporting
- A flowchart of the basic structure of your program

Resources

A sample set of files is available to test your server. **Your server must be able to properly serve any of these files, in addition to the simple webpage you design (that should be accessible at http://<you>.cy350.eecs.net/index.html).** For example, the URL http://<you>.cy350.eecs.net/test1.html should serve the provided test1.html page.

The set of files includes:

- pr2_shell.py – a base file to begin writing your server
- test1.html
- test2.html
- moved
- redirect.html
- images/
    - heavy_drop.jpg
    - JNN_STT_airfield.jpg
    - test.mp4

I have implemented my own version of this server offering these files and it is available for you to query when you are connected to eecsNET. It is at http://robert.ryan.cs484.eecs.net/

Pro Tips

- Use Wireshark to look at the HTTP interaction when accessing my web server!
- On your VM, python should be run as "python3".
- If you have VPN'd into eecsNET, your normal laptop browser should be able to send queries to your server or my server.

- Can’t get a response from your server? 1) Verify it is running. 2) Make sure your firewall is either turned off, or you have configured it to allow port 80 traffic.
- Remember, you will have to run as superuser to open a socket on port 80.
- Start with a simple working server example and build on it incrementally – read the documentation for *http.server*.
- Test your additions as you go.
- You should be able to request pages as http://localhost/ provided you are running your browser on the same host as the server. On a different host, you should be able to request pages with http://<you>.cy350.eecs.net
- Python documentation can help you with file manipulation, time, and date formats.
- Use print statements to see what part of the code your server is executing and what your partial responses look like.
- Use Wireshark to see what your packets look like on the wire. Are they following the HTTP protocol?