Computer Networks A365

Programming Assignment 1

This assignment must be done in Python 3

In this assignment, you will write a client and server application similar to FTP. The client connects to the server and can down-load any file accessible to the server application. Any number of files can be down-loaded and no access control is performed.

More specifically, the client initiates a control connection to the server port (the default port is #5000 if a port number is not given). All commands are sent over this connection. The client supported commands are: *list*, *get*, and *exit*. The client will request a list of file names that are available on the server, (list of files in the same directory as the server.py file). The client can ‘get’ a file by entering ‘*get filename.ext’*. When the client requests to down-load a file using *get*, the server sends the requested file back. After the client receives the file and saves it to disk, the user should be able to issue another *list*, *get* or *exit* command.

Additional requirements:

1. The client will need to verify that it received the entire file. (Therefore, the server must let the client know the file size before sending it).
   1. The server message will be the in the following format:

*‘sending filename.ext size (The size in bytes)’*

1. All connections must be properly closed when the client is finished, i.e. sends: *exit*.
2. The server must be able to handle simultaneous requests from multiple clients. (Threads can be used)
3. The client may ask for a file that the server cannot provide, so the client must be able to respond appropriately with an error message (since the client can obtain a list of files). (No crashing!)
4. Do not assume a file size, i.e. send the file in chunks of 1024 bytes.
5. Do not assume ‘localhost’.
6. The client should put the received file in the same directory as the client.py file (The client.py file could be on a different host).
7. Test your client and server with other students to see if you have made any assumptions.
8. If your program does not completely work, it should not crash, the user should receive the message ‘Not Implemented Yet’.
9. Do not hardcode any port numbers or IP addresses for the client. I should be able to provide the server and port number for the client side using command line arguments. Example: *client.py 129.32.56.2 12001*
10. Do no hardcode any port numbers for the server. To run the program, I will type python3 server.py *port\_number (or python3 server.py) and python3 client.py ipaddress port\_number*
11. Don’t do anything that would make your code specific to an operating system, such as windows pathnames, or Unix pathnames.

Submission Instructions:

1. Turn in 3 files server.py, client.py and README.txt, zipped with your last\_name.zip as the title of the assignment. DO NOT ADD ANYTHING ELSE TO THE ZIP FILE NAME.
2. All functions and variables should have meaningful names.
3. Use meaningful comments for functions
4. I will place 3 files, .txt, .jpg, .png in the server directory when I unzip your submission. I will run the server and clients on different machines, so test your applications similarly. (Use Pickle to send objects back and forth, such as lists and non-string objects.)
5. I will initiate several of your clients at the same time. You should test this also.
6. The README.txt should:
7. Let me know if your program is not fully implemented.
8. Identify whether you developed the application on a Windows, MAC or Unix machine.
9. Include specific instructions to type on the command line to run your program.
10. Forewarn of any implementation issues