

CP372 - Programming Assignment

CP-372-C

Dr. Heider Ali

Brandon Dang, Daniel Gao

169026034, 169041875

Due: February 10th, 2025

Introduction/Brief Description

To begin, our code starts with the launching of a server which initializes itself to an address and socket, in this case using localhost. Then, a client can be launched which will immediately attempt to connect to the server using the localhost address which the server will accept if there are less than 3 clients already connected. Once the connection is established, the client sends a name to the server to identify itself outside of the server-side usage of “Client##” where ## is an incrementing index stored within the cache. The server will then request that the client sends a message to name themselves which is then echoed back by the server to confirm that the client has been named. Now the client is able to send generic messages through the Command Line Interface (CLI) which are echoed by the server alongside an “ACK” flag. Certain messages are interpreted as commands, notably; status, list, get [file_name]. This is done by feeding all messages received by the server into a function that checks message content. Status will have the server send the list of all clients and their information which have connected to the server, both currently connected and previously disconnected clients. List has the server send back what files are stored within a specific repo/folder. Get [file_name] then makes the server stream the file contents back to the client which requested it. All messages sent by any client are also echoed by the server to all other clients with the name of the client that initially sent the message. There is also a variable that tracks the number of messages sent by a client that is used in the code currently to only determine what is the first message sent to determine what to name the client.

For special considerations, a client object class was created. Through its initialization, this new object would hold all the important data elements needed, allowing our group to bundle and reference names and client numbers by object.

Walk Through

Start Server.py

Start up to 3 Client.py (any additional ones are told that the server is full)

Each client will then be prompted to name themselves by sending a message to the server which will then send a reply back indicating that they have been successfully named

Now clients can send messages; if they are not recognized commands, they will receive an ACK from the server while all other clients will see that that client has sent a message

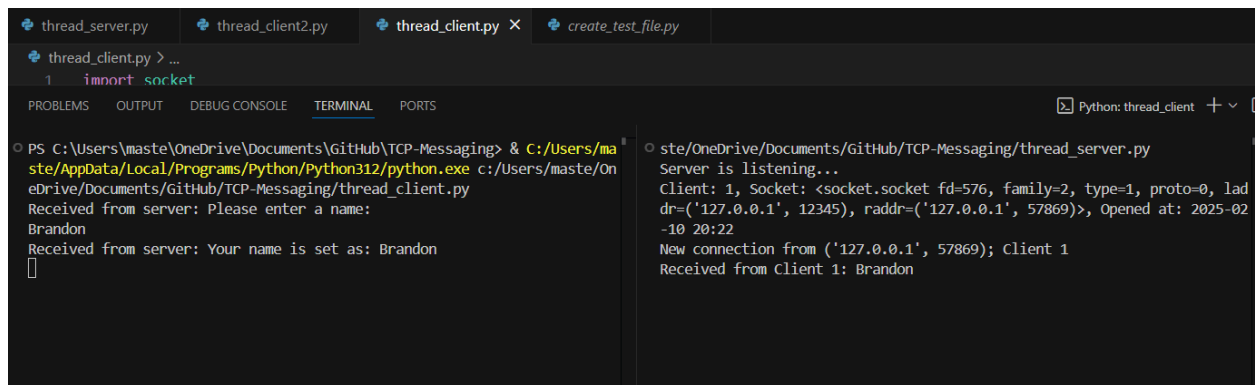
When a recognized command is sent, the server first ACKs the message before also sending the associated response from the server is sent back; in the case of using “get” with an invalid file name, the server will simply indicate that the file path is invalid.

Difficulties Faced

A key difficulty faced was the program returning an error when a client disconnected, and a connected client tried to message the server. This was because on the server side, the message sent by the client would be relayed to every client recorded in the cache. Because the cache needed to keep a history of disconnected sockets, the server would try to send the message to these sockets. To bugfix this, we implemented an additional test to check if the client object had a socket close_time. If this variable was set to None, it indicated the socket was still connected to the server and could relay the message.

Test Cases

1. Program can create a Server and client



```

thread_server.py  thread_client2.py  thread_client.py x  create_test_file.py
thread_client.py > ...
1  import socket

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
Python: thread_client + v

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:/Users/maste/AppData/Local/Programs/Python/Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client.py
Received from server: Please enter a name:
Brandon
Received from server: Your name is set as: Brandon
[]

ste/OneDrive/Documents/GitHub/TCP-Messaging/thread_server.py
Server is listening...
Client: 1, Socket: <socket.socket fd=576, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57869)>, Opened at: 2025-02-10 20:22
New connection from ('127.0.0.1', 57869); Client 1
Received from Client 1: Brandon
  
```

2. Each client created is assigned a name with correct number

```

thread_server.py x thread_client2.py x thread_client.py create_test_file.py
thread_client2.py > ...
1 import socket

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:/Users/maste/AppData/Local/Programs/Python/Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client.py
Received from server: Please enter a name:
Brandon
Received from server: Your name is set as: Brandon
[]

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:/Users/maste/AppData/Local/Programs/Python/Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client2.py
Received from server: Please enter a name:
Daniel
Received from server: Your name is set as: Daniel
[]

ste/OneDrive/Documents/GitHub/TCP-Messaging/thread_server.py
Server is listening...
Client: 1, Socket: <socket.socket fd=576, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57869)>, Opened at: 2025-02-10 20:22
New connection from ('127.0.0.1', 57869); Client 1
Received from Client 1: Brandon
Client: 2, Socket: <socket.socket fd=564, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57880)>, Opened at: 2025-02-10 20:24
New connection from ('127.0.0.1', 57880); Client 2
Received from Client 2: Daniel

```

3. Server can handle multiple clients (multi-threading)

```

thread_server.py x thread_client2.py x thread_client.py create_test_file.py
thread_client2.py > ...
1 import socket

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:/Users/maste/AppData/Local/Programs/Python/Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client.py
Received from server: Please enter a name:
Brandon
Received from server: Your name is set as: Brandon
Hello World
Received from server: Hello World ACK
Received from server: Daniel: Testing
[]

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:/Users/maste/AppData/Local/Programs/Python/Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client2.py
Received from server: Please enter a name:
Daniel
Received from server: Your name is set as: Daniel
Received from server: Brandon: Hello World
Testing
Received from server: Testing ACK
[]

ste/OneDrive/Documents/GitHub/TCP-Messaging/thread_server.py
Server is listening...
Client: 1, Socket: <socket.socket fd=576, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57869)>, Opened at: 2025-02-10 20:22
New connection from ('127.0.0.1', 57869); Client 1
Received from Client 1: Brandon
Client: 2, Socket: <socket.socket fd=564, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57880)>, Opened at: 2025-02-10 20:24
New connection from ('127.0.0.1', 57880); Client 2
Received from Client 2: Daniel
Received from Client 1: Hello World
Received from Client 2: Testing

```

4. Server is able to force number of connected clients to 3 clients

```

thread_server.py  thread_client2.py  thread_client3.py  thread_client4.py  thread_client.py  create_test_file.py
thread_client4.py > ...
1 import socket

Python/Python312/python.exe c:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging\thread_client.py
Received from server: Please enter a name:
Brandon
Received from server: Your name is set as: Brandon
Hello World
Received from server: Hello World ACK
Received from server: Daniel: Testing
[]

Python/Python312/python.exe c:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging\thread_client2.py
Received from server: Please enter a name:
Daniel
Received from server: Your name is set as: Daniel
Received from server: Brandon: Hello World
Testing
Received from server: Testing ACK
[]

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:\Users\maste\AppData\Local\Programs\Python\Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client3.py
Received from server: Please enter a name:
Third
Received from server: Your name is set as: Third
[]

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:\Users\maste\AppData\Local\Programs\Python\Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client4.py
Received from server: Server is full, try again later.

Python: thread_server + v
Received from Client 1: Hello World
Received from Client 2: Testing
Client: 3, Socket: <socket.socket fd=568, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57890)>, Opened at: 2025-02-10 20:26
New connection from ('127.0.0.1', 57890); Client 3
Received from Client 3: Third
[]

```

5. Server and client can exchange messages as described

```

thread_server.py  thread_client2.py  thread_client3.py  thread_client4.py  thread_client.py  create_test_file.py
thread_client2.py > ...
1 import socket

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:\Users\maste\AppData\Local\Programs\Python\Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client2.py
Received from server: Please enter a name:
Brandon
Received from server: Your name is set as: Brandon
Hello World
Received from server: Hello World ACK
Received from server: Daniel: Testing
[]

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:\Users\maste\AppData\Local\Programs\Python\Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client3.py
Received from server: Please enter a name:
Daniel
Received from server: Your name is set as: Daniel
Received from server: Brandon: Hello World
Testing
Received from server: Testing ACK
[]

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:\Users\maste\AppData\Local\Programs\Python\Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_server.py
Server is listening...
Client: 1, Socket: <socket.socket fd=540, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57911)>, Opened at: 2025-02-10 20:29
New connection from ('127.0.0.1', 57911); Client 1
Client: 2, Socket: <socket.socket fd=556, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57915)>, Opened at: 2025-02-10 20:29
New connection from ('127.0.0.1', 57915); Client 2
Received from Client 1: Brandon
Received from Client 2: Daniel
Received from Client 1: Hello World
Received from Client 2: Testing

```

6. A client can send “exit” and server cleanly disconnects the client for new clients

```

thread_server.py  thread_client2.py X  thread_client3.py  thread_client4.py  thread_client.py  create_test_file.py
thread_client2.py > ...
1 import socket

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:/Users/maste/AppData/Local/Programs/Python/Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client.py
Received from server: Please enter a name:
Brandon
Received from server: Your name is set as: Brandon
Hello World
Received from server: Hello World ACK
Received from server: Daniel: Testing
exit
PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging>

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:/Users/maste/AppData/Local/Programs/Python/Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client2.py
Received from server: Please enter a name:
Daniel
Received from server: Your name is set as: Daniel
Received from server: Brandon: Hello World
Testing
Received from server: Testing ACK

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:/Users/maste/AppData/Local/Programs/Python/Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_server.py
Server is listening...
Client: 1, Socket: <socket.socket fd=540, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57911)>, Opened at: 2025-02-10 20:29
New connection from ('127.0.0.1', 57911); Client 1
Client: 2, Socket: <socket.socket fd=556, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57915)>, Opened at: 2025-02-10 20:29
New connection from ('127.0.0.1', 57915); Client 2
Received from Client 1: Brandon
Received from Client 2: Daniel
Received from Client 1: Hello World
Received from Client 2: Testing
Received from Client 1: exit
Connection closed: ('127.0.0.1', 57911)

```

7. Server maintains clients' connections details

```

thread_server.py  thread_client2.py X  thread_client3.py  thread_client4.py  thread_client.py  create_test_file.py
thread_client2.py > ...
1 import socket

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:/Users/maste/AppData/Local/Programs/Python/Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client.py
Received from server: Please enter a name:
Brandon
Received from server: Your name is set as: Brandon
Hello World
Received from server: Hello World ACK
Received from server: Daniel: Testing
exit
PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging>

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:/Users/maste/AppData/Local/Programs/Python/Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_client2.py
Received from server: Please enter a name:
Daniel
Received from server: Your name is set as: Daniel
Received from server: Brandon: Hello World
Testing
Received from server: Testing ACK
status
Received from server: status ACK
Received from server: Client: 1, Name: Brandon, Socket: <socket.socket [closed] fd=-1, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57911)>, Opened at: 2025-02-10 20:29, Closed at: 2025-02-10 20:29
Client: 2, Name: Daniel, Socket: <socket.socket fd=556, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57915)>, Opened at: 2025-02-10 20:29, Closed at: None

PS C:\Users\maste\OneDrive\Documents\GitHub\TCP-Messaging> & C:/Users/maste/AppData/Local/Programs/Python/Python312/python.exe c:/Users/maste/OneDrive/Documents/GitHub/TCP-Messaging/thread_server.py
Server is listening...
Client: 1, Socket: <socket.socket fd=540, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57911)>, Opened at: 2025-02-10 20:29
New connection from ('127.0.0.1', 57911); Client 1
Client: 2, Socket: <socket.socket fd=556, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 57915)>, Opened at: 2025-02-10 20:29
New connection from ('127.0.0.1', 57915); Client 2
Received from Client 1: Brandon
Received from Client 2: Daniel
Received from Client 1: Hello World
Received from Client 2: Testing
Received from Client 1: exit
Connection closed: ('127.0.0.1', 57911)
Received from Client 2: status

```

Bonus:

8. Server is able to list files in its repository

See attached image for 9.

9. Server streams files of choice to clients

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
Python: thread_client + - [ ] ... ^ x

/usr/local/bin/python3 /Users/lonelywanderer/eclipse/CP372/TCP-Messaging/thread_server.py
lonelywanderer@Mac TCP-Messaging % /usr/local/bin/python3 /Users/lonelywanderer/eclipse/CP372/TCP-Messaging/thread_server.py
Server is listening...
New connection from ('127.0.0.1', 53485); Client 1
Received from Client 1: Client 1
Received from Client 1: status
Received from Client 1: list
Received from Client 1: get test.txt

Received from server: Please enter a name:
Client 1
Received from server: Your name is set as: Client 1
status
Received from server: status ACK
Received from server: Client: 1, Name: Client 1, Socket: <socket.socket fd=6, family=2, type=1, proto=0, laddr=('127.0.0.1', 12345), raddr=('127.0.0.1', 53485)>, Opened at: 2025-02-10 20:25, Closed at: None

list
Received from server: list ACK
Received from server: info.txt
test.txt
get test.txt
Received from server: get test.txt ACK
Received from server: Line 0
Line 1
Line 2
Line 3
Line 4
Line 5
Line 6
Line 7
Line 8
Line 9
Line 10
Line 11
Line 12
Line 13
Line 14
Line 15
Line 16
Line 17
Line 18
Line 19
Line 20
Line 21
Line 22
Line 23
Line 24
Line 25
Line 26
Line 27
Line 28
Line 29
Line 30
Line 31

```

(test.txt is just a text file that goes Line 0\n Line 1\nLine 2\n...Line 32\n, so this correctly prints the file contents)

Possible Improvements

- Optimizing how messages are sent to the other clients to not require browsing through disconnected clients, likely through an active clients list.
- Not echoing command messages to other clients.
- Using the message count statistic for other uses, like tracking it in the client info object.
- General optimizations.