Report

Note: Please use py 3.7 to run code

Client

The client is designed so that when you first try to login, it has to run through 2 nested while loops, which it has to go through to verify the username and password. If the user is a new user, it has to run a block of code that allows the server to add a new user. Once this is verified by the user, a new thread is created to receive messages where the message is split into different strings depending on the number of arguments outlined in the spec. Also, the application layer message format is determined by the string in the first element in the list when the split() function is called. This would run the relevant functions. The main thread sends messages to the server. When the P2P system is established, the client sends a request to connect to another client, and this would be run on another thread. When the client is logged out, exit is called and it runs the logout() function.

Server

The server's application layer message format is similar to the client's layer, where messages are parsed by splitting the message into different sections and the first element in the list is used to parse the command. The data structure design is the use of dictionaries for each storage type. For the blocked users section, a list is used for each key to store the various blocked users. Also, the online clients store a tuple, to make it easier to send messages to that specific socket. Some design tradeoffs are that there are no checks for empty strings and while loops are used extensively which slows the program down. Currently the P2P code does not work as intended. The P2P system works similarly to the server, but both clients can send and receive messages.

References for borrowed code

multithreading - How to exit the entire application from a Python thread? - Stack Overflow

python - How do I parse a string to a float or int? - Stack Overflow