COMP3331 Assignment Report -- z5271698

- **Version:** Python 3.7, server.py and client.py can both work on CSE.

- **Program design:**
  To run the program, we must start the server first, then the client. The client should ask the user input username and password, then this should be sent to server and authenticate them. If it is a new user, the program will ask you confirm your password again, the password should be the first you enter. If successful, the client should support all the commands. If not successful, an appropriate error message will be seen by the client.

  When a client enter command message and send to server, then client only waiting for the server's answer and displaying it to the user.

  In client, first we check the login. Then we have two threads, one is sent command message to server and another is receive the response from server and display the response.

  In server, we also have two threads. One is receive command message from client and analyze the data and send the client an appropriate response. Another is check timeout, send timeout message and sending pending message to clients.

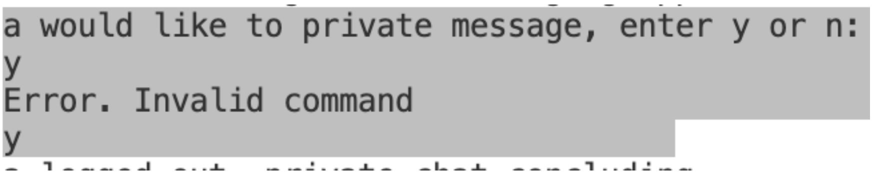
- **Data structure design:**
  - Client: client.py
  - Server: server.py
  - Additional: Individual.py – store each user's information and status.

- **The application layer message format**:
  All communications sent from the client to the server were command. When a client sends a message to the server, all message uses json format expressions to figure out what the user wants to accomplish. After matching a command, the server nee check error and use json format send response to be shown in the client's terminal.

- **How your system works:**
  - Run python3 server.py 4000 60 120
  - When the server starts, it begins to listen for new client connections and creates a thread for them, all messages from clients are checked in send_to_client and response messages are sent back to the client. When a user is not online, the messages will

    store in pending_message list, and when the user returns online, another thread will check the pending list and send the messages. And on the same time, in this thread, server will check all user's timeout.

  - Run python3 client.py 4000
  - When the client starts, it establishes a TCP connection to the server and creates two threads. One is sent command message to server and another is receive the response from server and display the response. For P2P, client getting another client's address and port number and establishing a TCP connection with that user directly in client.
  - IMPORTANT: when client receive "would like to private message, enter y or n:", please enter TWICE. This is the multi thread problem I didn't solve. There are two input in two thread, so when test startprivate, please confirm twice. The second y/n will be send to server.



```
a would like to private message, enter y or n:
y
Error. Invalid command
y
```

- **Design trade-offs**:
  - I use a new class called Individual to store each user's information and status. In server.py, we have dictionary to store all Individual class which is users_dict. It makes more easy to manage users status.
  - I use json format for all communicate messages between server and clients, which because it is easy to read and check. Only need to encode and decode.
  - Multiple threads in server and client for easy to manage the communicates between server and client, client and client at same time.
- **Reference**
  - Code from lab answer and assignment website COMP3331 provide.
  - About threading: https://blog.csdn.net/dongfuguo/article/details/53899426