COMP3331 Assignment Report

Program Design

My program uses Python 3.7

Client folder contains: client.py

Server folder contains:

- server.py
- server_utility.py: includes a *Forum* class and *ForumThread* class which I have used to represent the state and interaction of the forum and the messages in the forum.

Application Layer Message Format

I have chosen to send and receive messages between client and server through a JSON format. This is because JSON format provides a simple way of accessing and putting data into a JSON object as it is compatible with Python's data types such as lists and dictionaries. The JSON string can be converted to a JSON string (for encoding) and to a JSON object (for storing/accessing).

The message format of a client request to a server contains these fields and values:

- 'cmd': identifies the client's input command
- 'sender': identifies the sender of the message
- 'content': contains any information associated with the command (if any)

```
message = {
    "cmd" : "",
    "sender" : "",
    "content" : ""
}
```

While a response message from the server includes an additional 'status' field to indicate the server's response to the client's request

```
message = {
    "cmd" : "",
    "response": "",
    "sender" : "",
    "content" : ""
}
```

Working commands

User authentication and all commands are working under single server and client configuration.

Also works under single server and multiple clients.

EXCEPT for SHT command under multiple clients, it partially works. The user that initiates SHT will shut down that user and the server however other users must enter a valid command and then press enter for those users to also shut down.

How my system works?

Place “client.py” in client directory.

Place “server.py” and “server_utility.py” in a server directory.

Server:

- Server sets up socket and listens in a loop for any client connections, it accepts the connection and creates a new thread for each client connection.
- Server performs authentication checks for the client requesting connection.
- Then, the server will continually wait for client request messages where it will parse the request message by extracting the relevant fields. The information is passed to the ‘Forum’ object to apply its methods based on the request command and send back a response to the client with a status code.

Client:

- Client establishes a TCP connection with server and then gets authenticated.
- Client will input commands which will be parsed into a JSON message to be sent to the server.
- The server will respond back with a message and the output is printed to the client’s screen.

Design Trade-offs

I have created my own Forum and ForumThread data structure to maintain the state of the forum, its threads, and the messages within the threads. The Forum class contains a hash map of threads where the key is the thread name and value is the ForumThread object. The ForumThread class contains all the information of the thread including title, list of messages (ordered), creator and files.

I created these classes so that the forum and thread could be abstracted away where the server is only required to access the relevant forum methods/functions to handle the client’s command. This provides a simpler and easy to read interface. The trade-off was the extra space required to keep track of the state of the forum.

Also, as mentioned above, the use of the JSON format provided better compatibility and easier parsing compared to that of strings.

Improvements and Extension

An improvement I would make is on the structure of my client in order to successfully implement the ‘SHT’ command for multiple users. I initially wrote and structured the client

program that maintains a linear order to read user input, send message request and then receive server response. Having this order made it difficult to be able to receive a message from a server or potentially any other clients other than as a response to the initial message sent to the server.

Hence, an improvement I could make is to structure the client to have a send and receive function/process so that I can create two threads to send and listen for messages. This would allow the client to detect the SHT command at any time.