**COMP3331 assignment**

**Z5163479**

**Program Design:**

**Language:** Python 3.7

**Platform:** Windows and Linuxs

**Features implemented:**

**Client-server:**

* login,
* block after multiple logins,
* multiple logins from different machine,
* logout,
* detect duplicate login
* Download\_tempID
* Upload\_contact\_log

**Problems or Potential Problems:**

1. Multiple clients work. But the first client will need to logout or disconnect, otherwise other later user will wait for it
2. There is not registration for new user.

**Data Structure used:**

Dictionary and list

**Application Layer Protocol:**

Message format used:

* Header (Encoded length of encoded content)
* Data (Encoded content)

During authentication, the message format just the message header + message content. Like {header + credentials}. Then server will send back response with successfully login or other reminders.

After authentication, user will be able to use command, the sent message format from client is still same as during authentication. But this time, server will send back the message with the format {user\_header + user\_information + message\_header + message\_content}

**Transport Layer Protocol:** TCP

**Possible Improvements:**

1. OOP programming, it will make look cleaner and better code quality
2. Better login during the while loop(receive message), since there are too many if\_else statement in my code to control the loop.

**How my program works:**

1. Must run server.py <port> <block duration>
2. then client.py <host> <port> <p2p\_port>
3. **Authentication:**
   1. Login: client will send username and password to server, after server check, it will send back a reminder
   2. Server will block the user with 3 unsuccessful username and password
   3. After server unlock, it will also reminder the user that is been unblocked
   4. Server will detect the duplicate login
   5. Server is able to receive multiple logins from different machine
4. **Download\_tempID**
   1. when a client login, server will generate a tempID for this client, and store the tempID\_start\_time and end\_time in local variable, also in tempIDs.txt
   2. every time user require Download\_tempID, the server will check the end\_time of tempID. If it is expired, server will generate a new one, then send back to client
5. **Upload\_contact\_log:**
   1. Client will read z5163479\_contactlog.txt, and send its content to server
   2. Then server will stdout what it receive, therefore do contact\_log checking
   3. For contact\_log checking, server will stdout the matching UserID with the tempID and its start\_time
   4. After successfully receive the contact\_log and checking, it will send a successful reminder to client

**References:**

1. The UDP multi-threaded code provided below the assignment specification