CS 4850/7850 Computer Networks I

Project: ChatRoom Version 2

Due Date: Wednesday April 15, before 11:00am.

1. Overview

In this project, you will extend version 1 by using threads to implement a chat room that includes multiple clients and a server that utilizes the socket API. The socket API is implemented in many programming languages. You are permitted to use your language of choice as long as it utilizes the socket API.

The client program provides commands: **login** (allow users to join the chat room), **newuser** (create a new user account), **send all** (sends a message to the server and the server broadcasts the message to all logged in clients), **send UserID** (sends a message to the server and the server send a unicast message to the user), **logout** (quit the chat room), and **who** (list all the clients in the chat room).

The server runs a chat room service, manages all the clients and distributes the messages.

2. Description

You will implement server program and client program. The server will use 1 plus the last four digits of your student ID as the server port number to avoid conflicting with other students' server program. For example, if the last four digits of your student ID is 3456, then as the server port number is 13456.

In this project, there are multiple clients connecting to the server at the same time. *The server can support MAXCLIENTS number of concurrent clients. For the grading purpose, se*t *MAXCLIENTS = 3.* The commands listed in item 3 below need to be implemented at the client side and the server side implements the corresponding functions required to support these commands. When the server starts, it should first read the user account information from the given file users.txt. For grading purpose, the initial user accounts (UserID, Password) are (Tom, Tom11), (David, David22), and (Beth, Beth33).

3. Client/Server Functions to be implemented

1. **login** UserID Password

The client will send the UserID/Password information to the server. If the server can verify the UserID and the Password, the server will send a confirmation message to the client and inform all other clients that this client joins the chat room; otherwise, the server will decline login and send an error message to the client.

2. **newuser** UserID Password

Create a new user account. The length of the UserID should be less than 32, and the length of the Password should be between 4 and 8 characters. The server will reject the request if the UserID is already there. The users' IDs and passwords should be kept in a file.

3. **send all** message

Send the "message" to the server. The server will precede the message with the UserID of the sender and broadcast the message to all other clients.

4. **send UserID** message

Send the "message" to the server. The server will precede the message with the UserID of the sender and unicast the message to the client "UserID".

5. **who**

List all the clients in the chat room.

6. **logout**

Logout from the chat room. Once logout, the connection between the server and client will be closed. The server will inform all other clients that this client left.

4. Program specifications

- While logged out, a user should only be able to either login or create a new user. All other commands should be invalid while logged out.
- While logged in, a user should only be able to send messages or log out. The user should not be able to login while already logged in or create a new user while logged in.
- Password length and username length restrictions should be implemented as outlined above.
- New user accounts should persist between sessions.
- Usernames must be unique. A new user cannot be created with the same user name as an existing user
-

5. Programming Language

You can use any programming language you like (C, C++, Java, Python, Ruby,…etc). Server and client should be implemented as console applications using the socket API, so please do not add a Graphic User Interface to your program. You should run the code the same way as you did in Version 1 except that multiple clients will be open simultaneously. As most of you are familiar with C, client and server skeleton programs in C are posted on Canvas, including Visual Studio project files and compile instructions, as a starting point.

6. Grading (Total 100 Points)

Demonstrate your application to the TA during his office hours in the week specified.

- 10 points for each of the six commands (60 points total)
- 40 points for neat source code and implementing appropriate error messages. Your source code must be well commented, including an overall header with student name, date, program description, etc.
- You will lose 80 points for any bug that causes the program to crash or makes the program exit abnormally even if all commands can be demonstrated.
- You will lose 100 points if you do not utilize the socket API.

7. Code submission and test

You have to submit your source code files through the course Canvas site. Late or email submissions, or submission of executables will not be accepted. After the submission deadline, you will have to meet with one of our TAs to demonstrate the functionality of your program(s) by downloading your source code from Canvas, and compiling and running it on your own computer in front of the TA.

8. Sample Outputs

Sample client output for Tom.

```
My chat room client. Version Two. >
>send
> Denied. Please login first.
>login Tom Tom11
> login confirmed
>who
> Beth, David, Tom
> Beth: when is project 2 due?
>David: I do not know.
>send all April 26.
>David: really?
>David left.
```

Sample client output for David.

```
My chat room client. Version Two. >
>login David David22
> login confirmed
>who
> Beth, David
>send Beth are you there?
>Beth: yes
>Tom joins.
>Beth: when is project 2 due?
>send all I do not know.
>Tom: April 26.
>send all: really?
>logout
```

Sample server output.

```
My chat room server. Version Two.
Beth login.
David login.
David (to Beth): are you there?

Beth (to David): yes
Tom login.
Beth: when is project 2 due?
David: I do not know.
Tom: April 26.
David: really?
David logout.
```