COP-4338 Systems Programming

Programming Assignment 1:
Multi-threading & Internet Networking in Java

In this assignment, you will implement a simple chat room. This will require a server program.

1 Problem Specification

The server is multi-threaded. It maintains a list (a ArrayList will work well) of all the active connections. It will listen on a port for a new connection. You have the authority to choose the port number and design appropriate protocol for client-server communication (your choice must be clearly specified in the readme.txt file that you will upload along with your program on Canvas). When a client joins the chatroom, it spawns a thread to handle the connection, and the main thread resumes listening. The spawned thread sends out a message asking for an ID name on the socket. When the client replies, the connection information is added to the master list of chat room occupants (so there is a shared ArrayList of PrintWriter references). From that point on, anything that is typed by the client is relayed to all chat room occupants (including the client), with the ID identifying the client. Of course, if null is read from the client, or the special value LOGOUT then the client has disconnected and you should remove the connection from the shared ArrayList and then exit the thread gracefully (i.e. stop the infinite loop in the run method). Don't worry too much about error checking; however, you will need to be careful to synchronize access to the array list using locks, since otherwise you might delete an entry while another thread is trying to broadcast to everyone, potentially resulting in an out-of-bounds ArrayList access or worse. You can test the server by starting it and connecting via several telnet windows. (You can turn the echo on for the telnet windows to see what you are typing.)

Broadcast a message when the client first joins the chatroom, and then broadcast a message when the client leaves the chatroom (in the finally block, after adjusting the ArrayList, but immediately prior to terminating the run method).

Bonus Part (20%)

You should be able to have multiple chat rooms simultaneously. Also, the server should make sure that there are no two clients with the same ID name in a chatroom at any given time. If a client wants to join with an ID acquired by another chatroom occupant, the server has to notify the client to choose a different ID.

2 Submissions

You need to submit a *.zip* file compressing the following files:

- Java source file(s) of your program (*.java* files)
- *Readme.pdf* file specifying
  - the list of command/response pairs in your protocol along with the meaning of each one
  - parts of assignment that you have completely/partially implemented
  - snapshots of your machine showing the execution of your code if you have successfully implemented different parts of the chatroom.