Dept. of Computer Science and Engineering

CSE3214 – Computer Network Protocols and Applications

Project 3

Chat Server

The purpose of this project is for you to design and implement a simple client/server application. **You can work in this Project in groups of 2**. You can use either Java or C to do the project.

In this project, you will implement a simple chat program. The application consists of a chat server, and chat clients. You have to implement both.

The server

The server application is continuously running waiting for clients to register. When a client registers with the server it sends its login and password. For this project we will not concern ourselves with authentication. Any password will do (it will be ignored by the server anyway, just accept it for future improvements/modifications).

This main specifications of the project is mentioned below. However, you still have to make many design decisions. You have to justify these decisions in your report.

The server keeps a database of users. Ideally, the user should have database of all users and authenticate any new user who wants to login. However in this project, we assume that any client can login/register by sending his/her name and passwd.

The client

The client can contact the server to perform one of the following functions/requests

1. Register
2. Request a list of logged in users
3. Start a chat with one of the logged in users
4. terminate the session

The Protocol

The client contacts the server at the well known port and asks for registration. The registration packet is on the form

`\REGISTER<crlf>`

The server responds with a message

`login:`

The sender sends its login name

The server responds

`passwd:`

The client sends its password, the server will accept it no matter what (no authentication).

The server responds by

`SUCCESSFUL REGISTRATION`

If the client is already in the database (logged in already), the server sends the following message to the client

`You are already logged in`

The client can request a list of registered users by sending

`\LIST<crlf>`

The server responds by sending the login names of all registered users one name per line

The client can request a chat with a specific user

```
\CHAT <user_name> <crlf>
```

Where <user_name> is one of the users logged in currently.

If there is no such user is logged in, the server responds with the message

```
NO such user
```

If the user exists, the chat server contacts him/her to start a chat. The server sends the following message to the target user

```
_user_name_ wants to chat with you, do you accept? Please
answer Yes or No
```

If accepted, it informs the user who requested the chat, and the chat starts. If declined, it also informs the other party that the request was denied (note that the first and last underscore are literally a part of the message, for example if the user name is joh, then, the message is

```
_John_ wants to chat with you, do you accept?  Please
answer Yes or No
```

When the chat starts, every line the user sends is displayed at the other side as follows

```
_user_name_: the text sent by the user
```

The text is sent when the user types a new lines <enter> at the terminal.

Any user can end the chat by sending a line that starts with “.” (that is a period as the first character of the line, the rest of the line is ignored).

The user can also terminate the session (logout) by sending the following message to the server. The client must terminate the chat before logging out.

```
\LOGOUT<crlf>
```

The server responds

```
Good Bye
```

Then the connection is terminated

NOTE

In this project, I did not specify what protocol to use. I also did not specify if the server will relay messages between the 2 sides, or they contact each other and perform the chat without the server intervention.

These are design decisions that you have to make and justify in your report. You are also allowed to add to the above specs to improve your project. One condition though, every thing that you add should be very well documented in your report.

The report

Your report should contain the following.

- A description of the design, justifying all of your design decisions.
- If you added any bells and whistles to the basic problem I described above, document them.
- The code for both the server and client.

Any modifications/corrections to this documented will be typed in red

Please read the academic honesty guidelines page (at the course web page).