Basic Chat Portal Application

Directory Structure

Follwing is the directory structure of my system:

1. database: The directory contains all the database related files of the system, such as, user registration database, group database, etc.

2. documentation: The drectory contains documentation for the system.

3. file: The directory is a storage for files that gets transferred between users via the chat portal.

4. header: The directory contains all the header files that are used by my system.

5. helper: The directory contains all the helper code that is used by my main server and client code, such as, writing username and password to the database when a user registers in the chat portal, etc.

6. main: The directory contains the main server and client code.

7. test: The directory contains the test code, i.e, code to check that atmost 20 users can simultaneously register and atmost 20 users can simultaneously login at a time.

What the system does?

The system supports the following functionalities:

1. A user can register with a desired username and password.
2. A user can login by supplying his username and password.
3. A user can check who all other users are currently logged in to the chat portal .
4. A user can send a message to any registered user.
5. A user can create a group.
6. A user can join a group.
7. A user can send message to a group.
8. A user can send a file to another user.
9. A user can receive a file that is sent to him by another registered user.

Commands to use the system?

The system supports the following commands that can be executed by the client:

1. who : Check who all other registered users are logged in.
2. msg : Send message to a particular user.
3. create_grp : Create a group.
4. join_grp : Join a group.
5. send : Send file to another registered user.
6. msg_group : Broadcast message to a particular group once you have joined it.
7. recv : Receive the file that is being sent by someone.

Assumptions made by me

Following are the assumptions that i made while making the code :

1. Username and password will be maximum 40 characters long.
2. Username and password will be continuous, i.e, no space is present in username and password.
3. Usernames are case sensitive.
4. If a user sends a message to a group, then only those users in the group which are online will get the message.
5. Every message sent by a user will be maximum 920 characters long.
6. Group name will be maximum 40 characters long.
7. Group name will be continuous, i.e, no space is present in group name.
8. Group names are case-sensitive.
10. The absolute path of the directory in which the file to be sent is present will be maximum 1024 characters long.
11. The name of the file to be sent will be maximum 1024 characters long.
12. The complete absolute path where the received file is to be saved will be maximum 1024 characters long.
13. The chat portal supports transfer of files with simple file extensions (.txt, .c, .cpp), i.e, files that can be opened in a text editor like sublime, gedit, etc.

Corner cases handled by me

Following are the corner cases that i handled while making the code :

1. 20 users can be simultaneously logged in at a time.
2. 20 users can simultaneously register at a time.
3. No two users can have same username.
4. No user can login without registering.
5. No two groups can have identical names.
6. A user can only join a group if and only if the group already exists.
7. A user cannot join a group if he is already a member of the group.
8. A user can send a message to another user iff the receiving user is online.
9. A user can send a message to the group iff the group exists and the user is a member of the group.
10. A user can send a file to another user iff the receiving user is online.
11. A user can only send a file if and only if the size of the file is less than 1 MB.
12. A user can receive only one file at a time, i.e, multiple users cannot simultaneously send file to the same user. The receiving user must first receive a file from one user and once the receive is complete then he can receive a file from another user.

Errors handled by me

Following are the major errors that i handled while making the code :

1. I was able to register the client into the chat portal but wasn't able to log the client into the chat portal. I found out that i forgot to call the connect() function on the client socket handling the login functionality.

2. I was trying to make the server accept the incoming connections on both the ports simultaneously. I found out that this is not a good approach because the accept() function blocks the server, so i couldn't login if my server was blocked on waiting for a connection on registration port and vice-versa. So then i used select() function to make my server achieve the desired functionality.

3. While making the test case "to check that 20 users can simultaneously register at a time", the program was exiting after making one client registration connection. I realised that i need to spawn 21 threads where each thread makes one registration connection at a time. I still was getting a random output and then i realised that i need to put the code making registration connect request to the server inside mutex lock.