Name: Mohit Sharma **Entry Number: 2019CS10372**

Assignment 2: Multiple Client Based Chat Application

NOTE: All msg from client to server or server to client (other than the content of chat which is send after mentioning content-length in header) should end with "\n\n" format, i.e., an empty line (First "\n" is for end of previous line and second "\n" is for an empty line). Also, the maximum size of such a message should be 1024 bytes. Set because any normal username would not cause any required header to cross this size and hence would detect an ambiguous stream of bytes quite early.

The content message should have a maximum size of 100000 set for storage purpose of the content.

User Registration:

1) Client to server registration for sending messages
   REGISTER TOSEND [username]\n\n
2) Server to client, if username is well formed
   REGISTERED TOSEND [username]\n\n
3) Client to server registration, for receiving messages
   REGISTER TORECV [username]\n\n
4) Server to client, if username is well formed
   REGISTERED TORECV [username]\n\n
5) Server to client, if username is not well formed
   ERROR 100 Malformed username\n\n
6) Server to client message in response to any communication until registration is complete
   ERROR 101 No user registered\n\n

Rules for username:

1) Username contains only alphabets and numbers.
2) Username are case sensitive.
3) Empty string cannot be a username.
4) The string "ALL" cannot be a username.
5) Two different users cannot have same username. It will cause ERROR 100.
6) A too large username such that the register message from client to server is not stored in a buffer size will cause ERROR 100.

Message Sending from Client to server:

1) Client to server message
   SEND [recipient username]\n
   Content-length: [length]\n\n
   [message of length given in the header above]
2) Server to client message, if message is delivered
   SEND [recipient username]\n\n
3) Server to client message, if message is undelivered
   ERROR 102 Unable to send\n\n
4) Server to client message, if header is incomplete
   ERROR 103 Header incomplete\n\n

Rules for sending message:

1) In the client application type the message as @username [message]
2) Username should be a valid username following the defined rules or it can also be “ALL”.
3) If the username does not follow the given rules then client rejects the message and if it still reaches server then server sends ERROR 103.
4) If corresponding username is not registered then ERROR 102 will be received.
5) If “ALL” is used as the username then the message is broadcasted to all registered users (other than he himself), if any user sends ERROR 103 then server will revert with ERROR 103, else if any user sends ERROR 102 then server will revert with ERROR 102, else server will revert with SEND ALL message.
6) In case of ERROR 103, the server will close both connection with the client but client will get to know about SEND connection only after trying to send message to server.
7) If the username mentioned is his own, then the server will revert with ERROR 102.
8) The server will wait for the content-length amount of message to be received from the user and hence if the message is of smaller length, then it will get stick waiting to read more message as no timeout is used. If the message length is larger than mentioned length then only mentioned length of the message will be sent and the rest of message will hinder with the next message and cause unwanted errors. (There is a commented code also to handle the other situation to send ERROR 103 message length not matches as mentioned in header, it assumes the message will not be defragmented, but I don’t think this approach is correct and hence I have not uncommented it and made it the main approach to follow).

Message Forwarding at Server:

1) Server to client message
   FORWARD [sender username]\n
   Content-length: [length]\n\n
   [message of length given in the header above]
2) Client to server message, if message is received
   RECEIVED [sender username]\n\n
3) Client to server message, if header is incomplete
   ERROR 103 Header incomplete\n\n

Rules for forwarding message:

1) The message is forwarded to the corresponding user (if username is not “ALL”) and according to the received message from the client or from the fact that it is not at all sent, the server reverts back to original sender of the message.
2) If username used is “ALL” then the message is broadcasted **parallelly** to all the registered users (other than itself). And even if one of them fails then it is considered that the overall broadcast fails.
3) If the receiver client sends ERROR 103 to server then it also closes the RECV connection with it as the correct server will not do so hence it is a corrupt server.

Connection Closing:

1) When client is shutdown then server closes both connections with it and a new user can be created with username of closed client.
2) When server is shutdown then instantly RECV connection of client is closed but the SEND connection is not closed and when user tries to send some message then the connection is closed.

To Run Server:
Compile using: **g++ -pthread -o server server.cpp**
To run: **./sever**

To Run Client:
Complie: **g++ -pthread -o client client.cpp**
To run: **./client [username] [ipaddress]**

Here [username] is the username for the client to register with server and [ipaddress] is the IP address (not domain name) on which the server is running (well this is confusing as the port number was not said to take as input, so PORT number is hardcoded as 8080). My server code is hardcoded to run on IP “127.0.0.1” and PORT “8080”.