Ameer Khan
ID: 1476214
3/15/19

CMPE 156: Final Project Documentation

In order to compile the program, first go to src folder which has the makefile, client.c and server.c. First run make clean, then run make which will create an executable called client and server in the bin folder. In order to run the server file in the src folder, you must do ../bin/server <port number>  and in order to run the client file in the src folder, you must do ../bin/client <ip address> <port number> <client name>.

Now I will explain the protocol. The client will send a connection message to server with its client id at the beginning. When the server receives a connection from the client, a new thread is created to handle the client. Each client that is created has its own thread. There are four important commands, the client can enter /list, /wait, /connect and /quit. When the /list command is entered, the server maintains a list of waiting clients and sends it to the clients that requests it. When the /wait command is entered, the client will create a port to listen to from other clients that requested a connection to this client. When the /connect command is entered, the client sends the id to the server and the server relays that message to the client in the wait state and connects both clients. When the /quit command is entered, it will exit the program by sending a string /quit to the client from the server letting it know to quit. The final command is control C which is used to leave the wait and chat state and return to the info state. I send a message exit to let the other client know control C was pressed if it is in the chat state. I send a message stop to let the client know to stop waiting if control C was pressed in the wait state.

Now I will explain the message formats. On the server side, the user has to insert one argument which is the port number. On the client side, the user has to insert three arguments.

The first argument is the ip address, the second argument is the port number, and the third argument is the client id. Once the arguments have been entered on the client side and server side, I have print statements on the server that prints the buffer that the client entered. On the clients, I have messages printed depending on the situation. For example, if chat state is ended, I have the message Left Conversation with client id printed. If control c was pressed during the wait state, I print the message Stopped Waiting. I also have error handling messages printed to stderr.

Now I will talk about error handling done in the client and server programs. The error checking done on the client side is to check the correct number of arguments input by the user, error checking for opening the socket, error checking for checking if the ip address is valid, error checking for connecting to the server, error checking for writing/reading from the server. The error checking done on the server side is to check the correct number of arguments input by the user, error checking for opening the socket, error checking for binding the socket to the port number, error checking to see if connection is valid with the client, error checking for reading in the socket from the client.