Assignment 2
EECS 3214
First Name: Baidi
Last Name: Liu
York ID:211559093
EECS Account:liubd

In this assignment, I built a simple TCP based P2P chatting program base on the first assignment.
This program has all the functions of the first assignment(JOIN,LEAVE,LIST,QUIT). Additionally, I add the TCP based chatting function to the program, so the client will be able to chat with another client who is already in the "online and willing to chat" list.

How it works:
I will talk about the P2P function more in this part, since the other functions were discussed in assignment 1.
JOIN: if the client is not in the list, JOIN the list.
LEAVE: LEAVE the list if the client is not in the list.
LIST: List all the members in the list to client.
QUIT: if the client is in the list, remove it and close the socket and quit. Else close the socket and quit.

P2P Chatting:
Assume there are several clients in the list, each of the Clients can be a P2P Server or P2P Client.
For example, assume client A tries to chat with client B.
First of all, A and B both have to be in the list. That means the clients have to use the command "JOIN" to JOIN the list.
Secondly, A will send a TCP request to B via the server with client B's IP address and port number(need to be typed manually) That means client A has to look through the "willing to chat " list to fetch client B 's information by type in command "LIST".
Client A will setup a TCP PSP server socket to wait for client B's connection. In this time, client A will become a P2P server. The request connect message will be sent via server.
Third, Client B will act as a P2P Client to connect to A. Then a TCP socket between client A and Client B is setup.
P2P connection complete.
They can chat with each other by typing **CHAT_TK: x**(x is the words you want to say). If client A or Client B wants to end the conversation, they can type in **CHAT_Quit** to end the connection. The TCP P2P socket will be closed.
During the whole process , the Sever remains activate, so there will be 2 TCP sockets connections at the same time. Therefore the Clients can use the JOIN,LEAVE,LIST,QUIT commands at any time.

Improvement:
1, The program can be improved by adding a user name and password system for the clients. So the privacy of the clients can be secured.
2, In my program, if the client wants to chat with other client, he/she has to list all the clients online and fetch the other client's information and request a connection manually. It can be improved by adding functions that check if the target client is online, if online, then connect the client by using his/her user name. All these actions can be done by one command.

Situations when my program does not work correctly:
1, For the P2P connection, the client can only connect and chat with only one other client.
2,After the P2P connection is setup, if the client does not type "CHAT_TK:" in front of the words, the message would not be sent to the another side, it will go straight to the Server.

How to compile:
Compile and run *Server.java* first, then compile *Client.java* and execute *Client.java* like below:
java Client xxx,xxx,x,xx(Host IP address) 29093(Host port number)
For example: **java Client 192.168.0.14 29093**

Use "JOIN" command to join the list. Use "LIST" command to get the list of online users.
Find the IP address and port number of the Client which you wanted to connect to.

Type the command below to connect to target:
CHAT_RQ:target IP address,target port number
For example: if you want to connect to a Client with IP:192.168.0.14, port number:58281:
CHAT_RQ:192.168.0.14,58281

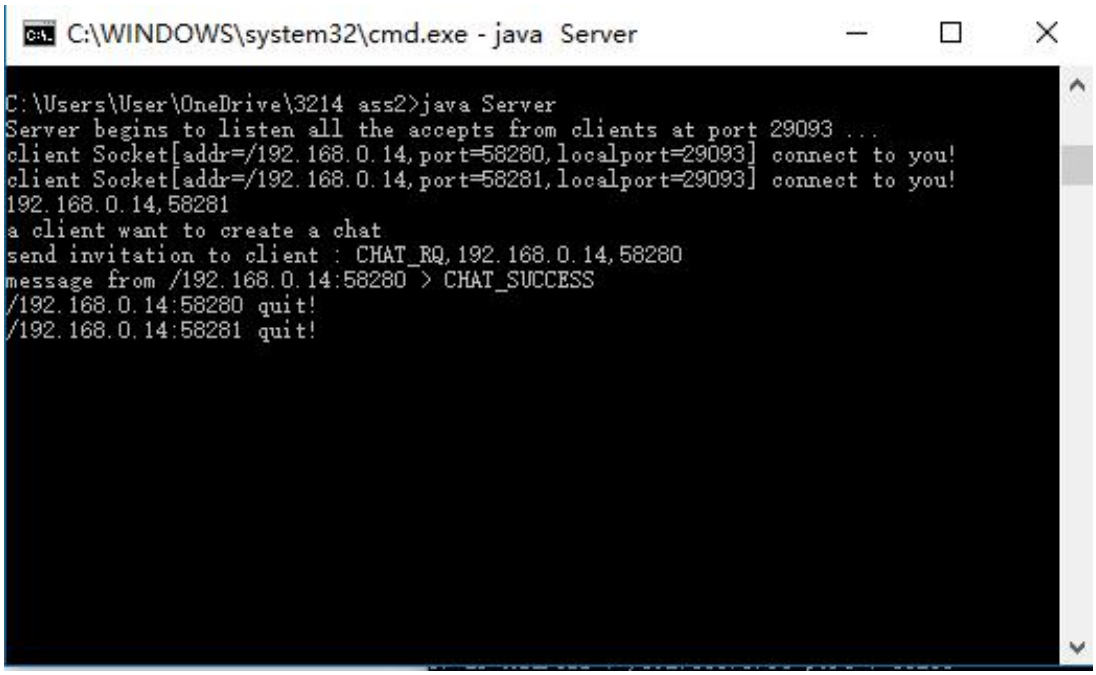
After the connection has been setup, you can chat with the connected client by type in:
CHAT_TK:some words.
For example, you want to say "hello":
CHAT_TK:Hello

If you want to quit the chat, you can type in:
CHAT_QUIT
Then the p2p connection will be disconnected.

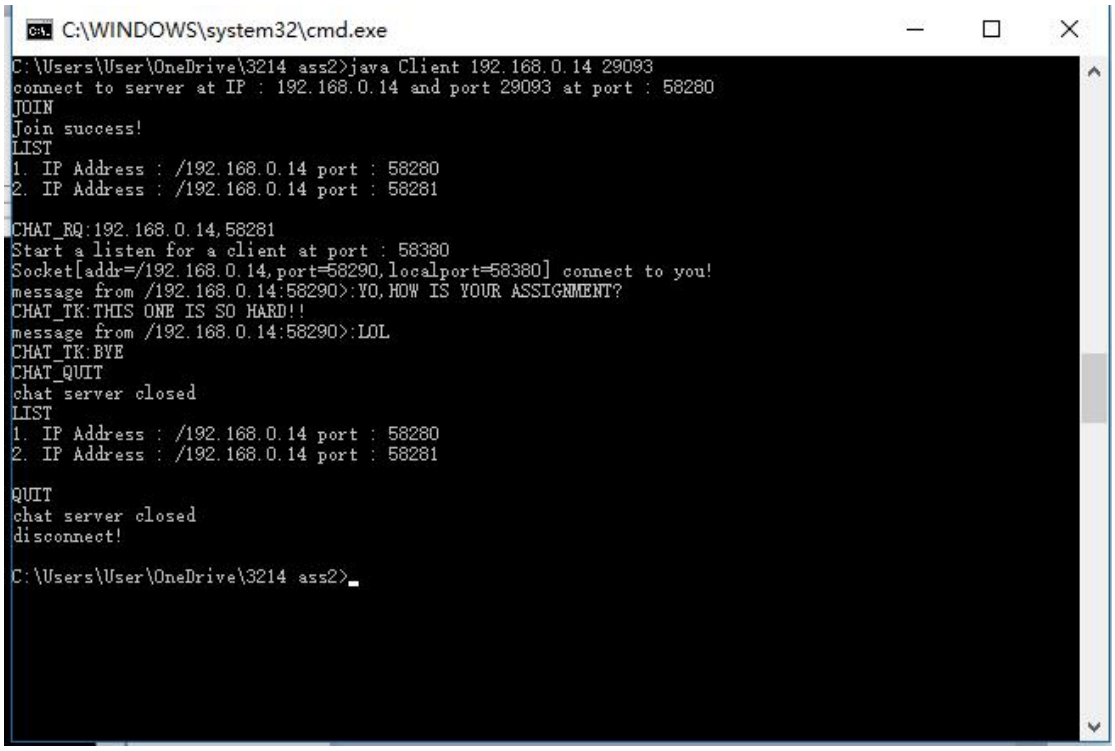
Testing:

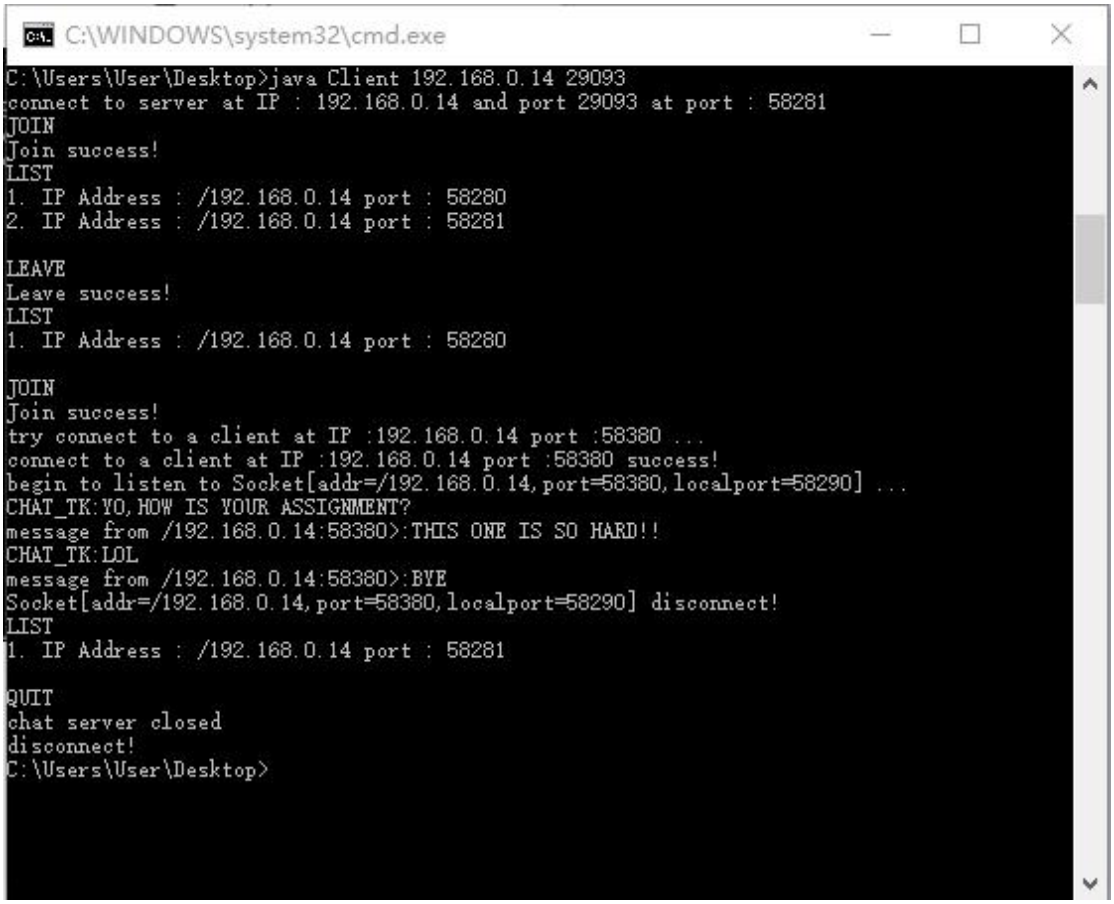
Server.java:

```
C:\Users\User\OneDrive\3214 ass2>java Server
Server begins to listen all the accepts from clients at port 29093 ...
```



Client.java




C:\WINDOWS\system32\cmd.exe
C:\Users\User\Desktop>java Client 192.168.0.14 29093
connect to server at IP : 192.168.0.14 and port 29093 at port : 58281
JOIN
Join success!
LIST
1. IP Address : /192.168.0.14 port : 58280
2. IP Address : /192.168.0.14 port : 58281
LEAVE
Leave success!
LIST
1. IP Address : /192.168.0.14 port : 58280
JOIN
Join success!
try connect to a client at IP :192.168.0.14 port :58380 ...
connect to a client at IP :192.168.0.14 port :58380 success!
begin to listen to Socket[addr=/192.168.0.14,port=58380,localport=58290] ...
CHAT_TK:YO,HOW IS YOUR ASSIGNMENT?
message from /192.168.0.14:58380>:THIS ONE IS SO HARD!!
CHAT_TK:LOL
message from /192.168.0.14:58380>:BYE
Socket[addr=/192.168.0.14,port=58380,localport=58290] disconnect!
LIST
1. IP Address : /192.168.0.14 port : 58281
QUIT
chat server closed
disconnect!
C:\Users\User\Desktop>