CN End Sem Lab Exam U11866

→ Basically speaking chat application can be done in both ways using TCP support or UDP support. But in general all chat applications need to be quick in delivering their messages to all. So when compared to TCP is quick but less reliable. So I am going to use UDP for the following.

Idea to satisfy requirements:

→ Assuming that all clients know the server socket we will proceed.

→ Whenever a new user joins the user will be authenticated via terminal al to keep things simple.

→ Server will get a request from new client to register. We will use a structure array to keep track of client socket numbers

→ Whenever a new user is added al server table it need change or notify of it with others.

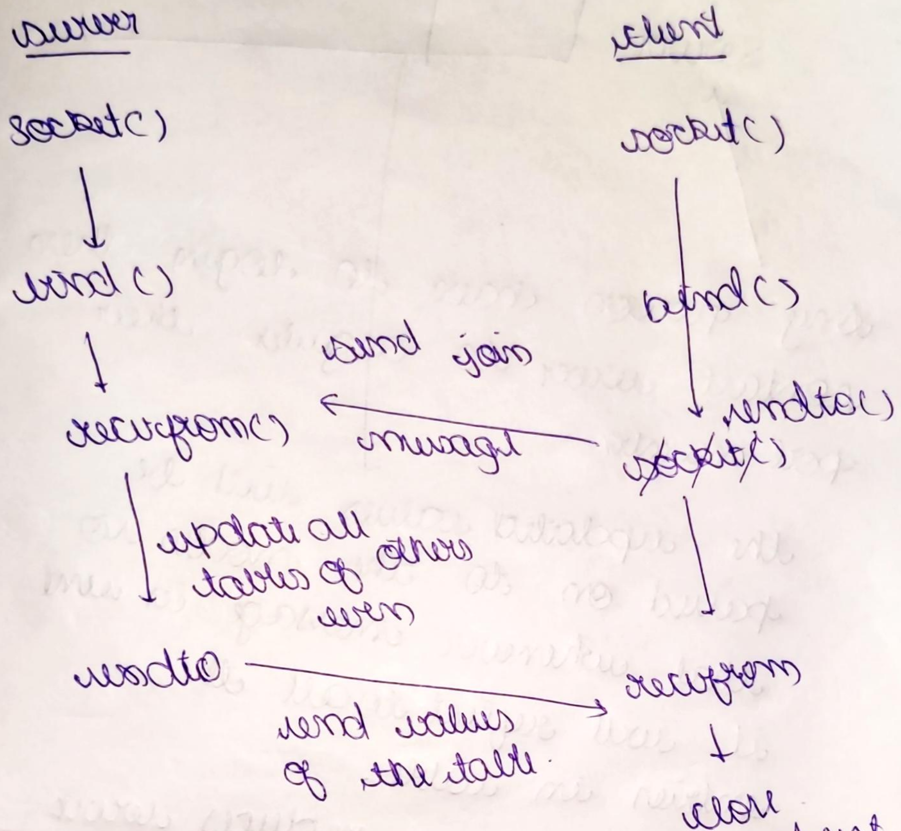
→ So basically every user will 411860 have table of currently logged in users.

→ This table will be updated by by server now and then by sending a message of newly added user.

→ Whenever user sends a message it will reach to all people present in this table.

→ This can be done using any of the TCP or UDP architecture.

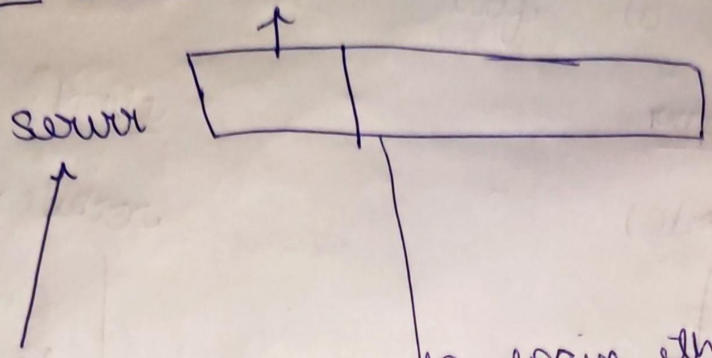
→ We can use threads to seperate the send and recieve processes so that they can occurr simultaneously without any interruptions flow chart for server client whent join at first U11866



→ So we will add bind to client side even so that port will not be changed during the whole time

→ whenever sender sends message it will be sent to all in the list.

Flow chart to store client to id 411860



Any person have to login then contact server to register their port number.

the updated values will be passed on to the nodes so that whenever message is sent it will reflect to all the entries in table.

This mechanism reduces server side busyness. Server can just look after the tables only and send whenever updated.