



Computer Engineering Department

Networks1 (10636454)

Programming HW 2

Chatting Between Peers

Instructor name:
Dr. Eng. Saed TARAPIAH

Academic Year: **2023-2024**

Semester: **Fall**

Credit Hours: **3**

Date: **18/November/2023**

Student Name:.....

Serial number: :رقم تسلسلي :.....

Registration Number:

Section:.....

Total Exam Mark: **15**

Exam Weight: **15%**

Deadline: 2/December/2023 midnight

Question	Points	ILO's	ILO's%	Question Grade	Required Time
Q1	6				
Q2	9				
Student Grade					

Exam Notes:

- 1- Closed Books & Notes.
- 2- Read each problem carefully before attempting to solve it.
- 3- Write all work on this exam paper.

Q1: [6 points]

In this project Peer to Peer Application that runs on a Network. You will Write it in Java.

Project Parts (Assignments):

1. Part 1 (Question1)

In this part, you are required to write Peer-to-peer Application Chatting Java **UDP** Socket Programming: Write a GUI application by using Java. Your program should have Text boxes, Buttons, Text Areas, Drop Downs etc. You should enter Source and destination IP addresses, port numbers. Also, you should display sent and received messages. See the sample for GUI Interface on **Page 3**. Two clients Are shown. The Buttons for Login, Logout, and the List of Online users is Not Required for This part. They are Required in Par2. Blocking Code is Not Accepted.

Page 2 Shows Two Clients and sample GUI

Extra requirements:

- Add the timestamp for each exchanged message (for both sent and received)
- Use **RED** color for sent messages
- Use **GREEN** color for received messages
- Add button to delete selected message or delete all conversation from both sides.
- Replicate the same task by using **TCP** Socket instead of **UDP** Socket.

Q2: [9 points]

2. Part 2 (Question2)

In this part you are required to apply TCP Java Socket programming, but we will add Two Parts: A TCP Client to the Application described in 1. This Client requires the additions of buttons to register to the TCP Server Described in 2.2. The Server will simply keep a list of UDP clients involved in the Chatting. The TCP Server will send a message to each TCP client in the Active Chatting Client described in 1 to inform it of the List of Active Clients. This is similar to Skype and other chatting Servers. The server just keeps a list of those that Active Chatting Clients. So, you will need to modify the code in 1 to accommodate this Requirement.

The Actual chatting will remain peer-to-peer. But the User chooses which Client to talk to from the List Provided by the Server. The login in the Clients is used for Registration to the TCP server to keep track of online Clients. You Should display the List of Active Client. **See pages 4, 5**

3
Add a TCP Server that keeps track of the Active Chatting Clients as described in 2.1. It should Show a GUI with the Active List of Clients. The login in the Clients is used for Registration. **See pages 4, 5**

Blocking Code is Not Accepted.

Part1(HW1): This Figure Shows Two Clients used in Part 1 and 2. The Login/Logout and TCP Server are not required in Part1 but will be Required in Part2.

Extra requirements:

- Enable sent to all option
- Add the name for each online user next to its corresponding IP and port number i.e. Ali 192.168.1.25 600
- Add file at the TCP server side, which contains the credential for users, login information as user name and password as follows:

User Name	Password
Ali	1234
Saly	A20B
Aws	ABcd
Adam	1Cb2

Both user name and password must be valid for login process, if OK, then the client can proceed and a notification can appear as (Logged in successful), else, a message will appear (invalid login information, either user name or password)

- enable the logout button.
- Use different test color for each user.
- Consider the username and passwords are **not case sensitive**. i.e. Ali = ali = aLi = ...

ClientChat

Username: Login Logout

Rem: Hello from 6000
Rem: Hello2
Me: Hoeelo from 5000 12444444444444

enter text here

Status: Received From:IP=192.168.1.112, Port = 6000

TCP Server IP: 127.0.0.1
TCP Server Port: 8888

Available Interfaces
Wi-Fi:192.168.1.112

Local IP: 192.168.1.112
Local Port: 5000
Remote IP: 192.168.1.112
Remote Port: 6000

Send test Button

Online Users

ClientChat

Username: Login Logout

Me: Hello from 6000
Me: Hello2
Rem: Hoeelo from 5000 12444444444444

enter text here

Status: Received From:IP=192.168.1.112, Port = 5000

TCP Server IP: 127.0.0.1
TCP Server Port: 8888

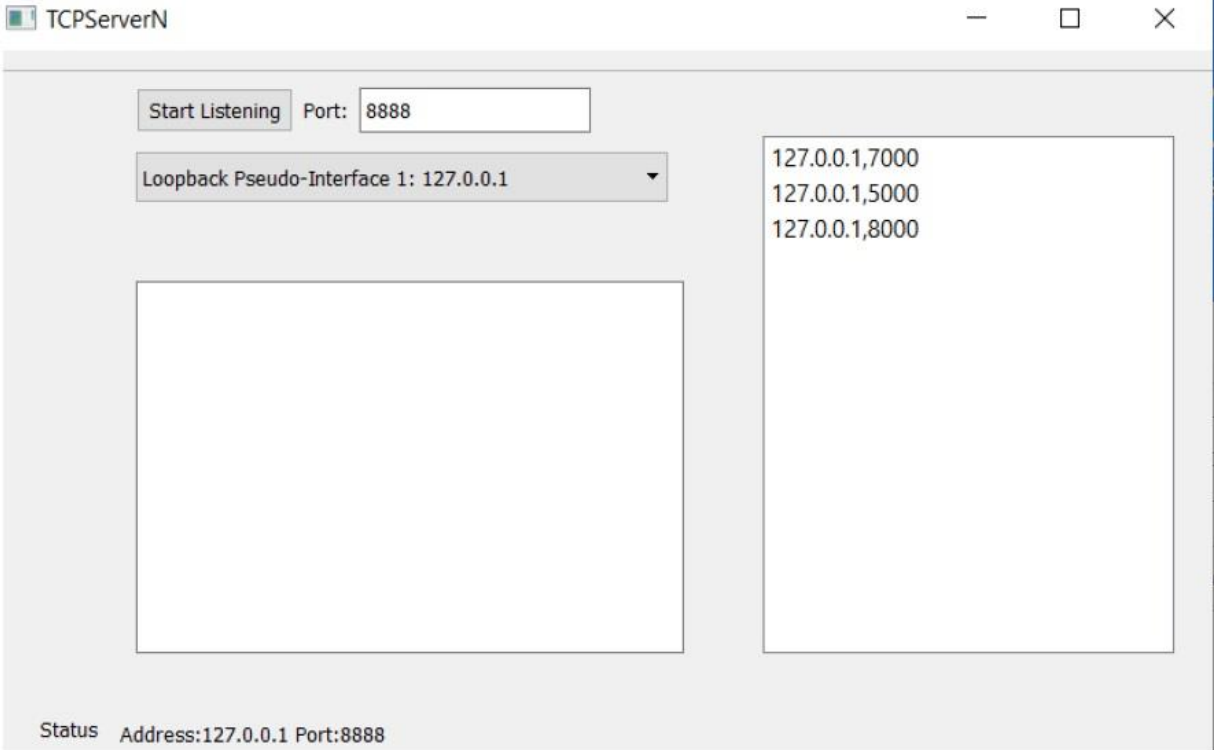
Available Interfaces
Ethernet:169.254.49.56

Local IP: 192.168.1.112
Local Port: 6000
Remote IP: 192.168.1.112
Remote Port: 5000

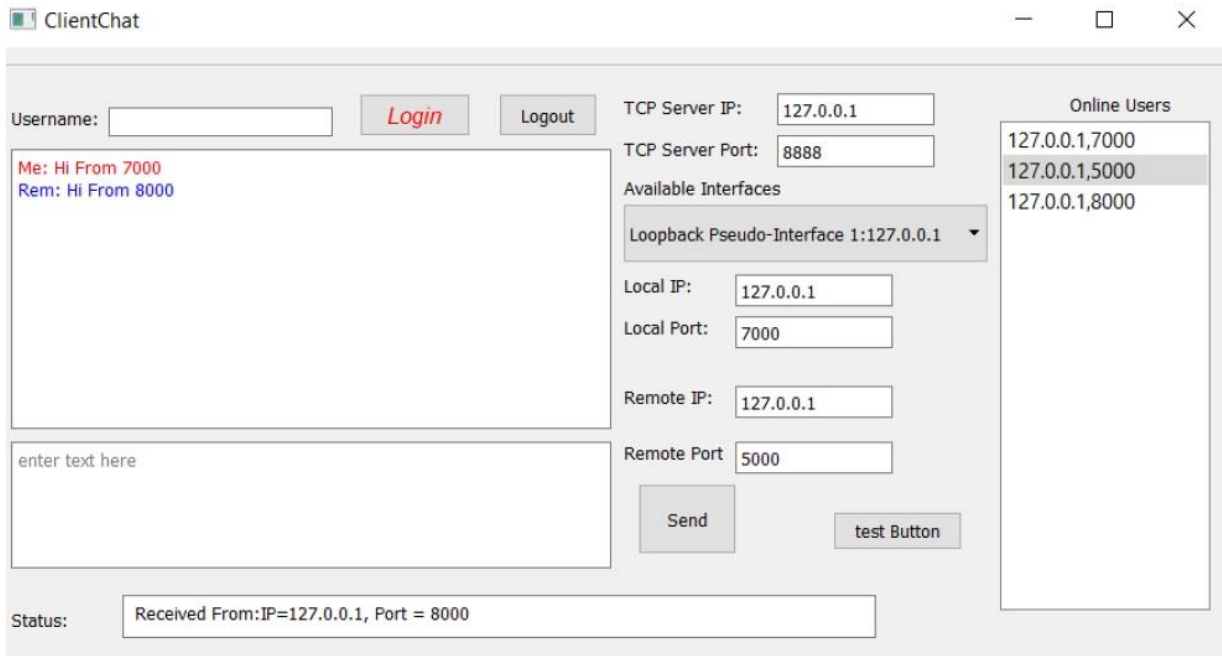
Send test Button

Online Users

Part 2(HW2): Here we show The TCP Server and # Chatting Clients each has a UDP and a TCP Client in the Code. Same Code for the Clients



Client1
Client1



Client 2:

ClientChat

Username:

Login

Logout

Me: Hello from 5000

Rem: Hi From 7000

enter text here

Status: Received From:IP=127.0.0.1, Port = 7000

TCP Server IP: 127.0.0.1

TCP Server Port: 8888

Available Interfaces

Loopback Pseudo-Interface 1:127.0.0.1

Local IP: 127.0.0.1

Local Port: 5000

Remote IP: 127.0.0.1

Remote Port: 8000

Send

test Button

Online Users

127.0.0.1,7000

127.0.0.1,5000

127.0.0.1,8000

Client 3

Client 3

ClientChat

Username:

Login

Logout

Rem: Hello from 5000

Me: Hi From 8000

enter text here

Status: Received From:IP=127.0.0.1, Port = 5000

TCP Server IP: 127.0.0.1

TCP Server Port: 8888

Available Interfaces

Loopback Pseudo-Interface 1:127.0.0.1

Local IP: 127.0.0.1

Local Port: 8000

Remote IP: 127.0.0.1

Remote Port: 7000

Send

test Button

Online Users

127.0.0.1,7000

127.0.0.1,5000

127.0.0.1,8000