Experiment : 8
Date : 07/06/2021
Author : Bonnie Simon

# Multi User Chat Server using TCP

## AIM

Implement a multi user chat server using TCP in python.

## ALGORITHM

### Server.py

1. Start
2. Instantiate instance of socket as server
3. Bind server to localhost and port 55555 and listen
4. Receive nicknames from clients
    a. Print nicknames
    b. Broadcast all messages that comes from the clients
5. Stop

### Client.py

1. Start
2. Instantiate instance of socket as client
3. Connect to localhost at port 55555
4. Receive on a thread[target=receive]
    a. Print messages received by the client
5. Write on a thread[target=write]
    a. Input message from client
    b. Send it to server
6. Stop

## PROGRAM

Client.py

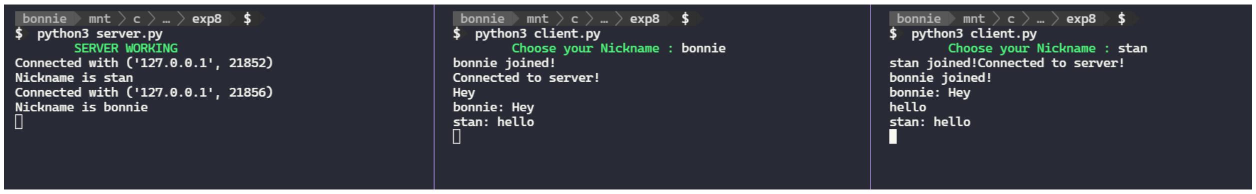
```
import socket
import threading

# Choosing Nickname
nickname = input("\33[32m \tChoose your Nickname : \33[0m")

# Connecting To Server
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(('127.0.0.1', 55555))

# Listening to Server and Sending Nickname
def receive():
        while True:
        try:
        # Receive Message From Server
        # If 'NICK' Send Nickname
        message = client.recv(1024).decode('ascii')
        if message == 'NICK':
                client.send(nickname.encode('ascii'))
        else:
                print(message)
        except:
        # Close Connection When Error
        print("An error occured!")
        client.close()
        break

# Sending Messages To Server
def write():
        while True:
        message = '{}: {}'.format(nickname, input(''))
        client.send(message.encode('ascii'))

# Starting Threads For Listening And Writing
receive_thread = threading.Thread(target=receive)
receive_thread.start()

write_thread = threading.Thread(target=write)
write_thread.start()
```

Server.py

```
import socket
import threading

# Connection Data
host = '127.0.0.1'
port = 55555

# Starting Server
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind((host, port))
server.listen()
print("\33[32m \tSERVER WORKING \33[0m")

# Lists For Clients and Their Nicknames
clients = []
nicknames = []

# Sending Messages To All Connected Clients
def broadcast(message):
        for client in clients:
        client.send(message)

# Handling Messages From Clients
def handle(client):
        while True:
        try:
        # Broadcasting Messages
        message = client.recv(1024)
        broadcast(message)
        except:
        # Removing And Closing Clients
        index = clients.index(client)
        clients.remove(client)
        client.close()
        nickname = nicknames[index]
        broadcast('{} left!'.format(nickname).encode('ascii'))
        nicknames.remove(nickname)
        break

# Receiving / Listening Function
def receive():
        while True:
```

```
        # Accept Connection
        client, address = server.accept()
        print("Connected with {}".format(str(address)))

        # Request And Store Nickname
        client.send('NICK'.encode('ascii'))
        nickname = client.recv(1024).decode('ascii')
        nicknames.append(nickname)
        clients.append(client)

        # Print And Broadcast Nickname
        print("Nickname is {}".format(nickname))
        broadcast("{} joined!".format(nickname).encode('ascii'))
        client.send('Connected to server!'.encode('ascii'))

        # Start Handling Thread For Client
        thread = threading.Thread(target=handle, args=(client,))
        thread.start()

receive()
```

## OUTPUT



```
bonnie  mnt > c > … > exp8  $
$  python3 server.py
        SERVER WORKING
Connected with ('127.0.0.1', 21852)
Nickname is stan
Connected with ('127.0.0.1', 21856)
Nickname is bonnie
```

```
bonnie  mnt > c > … > exp8  $
$  python3 client.py
        Choose your Nickname : bonnie
bonnie joined!
Connected to server!
Hey
bonnie: Hey
stan: hello
```

```
bonnie  mnt > c > … > exp8  $
$  python3 client.py
        Choose your Nickname : stan
stan joined!Connected to server!
bonnie joined!
bonnie: Hey
hello
stan: hello
```

## RESULT

The python program has been executed and verified successfully.