```
# TCP (lan) based chatting app
# ------------------------------------------------------------
#                       client.py
# ------------------------------------------------------------

# req. libs
import socket
import threading



# global
## connection data
host = '127.0.0.1' # server ip
port = 4444        # free/open port
print(f"[#] connecting to {host}:{port}")

## nickname
nickname = input("[+] enter nickname:")

## starting client
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect((host, port))
print(f"[+] connected to {host}:{port}")



# listening to server & sending nickname
def receive():
    while True:
        try:
            # receive msg
            msg = client.recv(1024).decode('ascii')

            # print msg
            if msg == 'NICKNAME':
                client.send(nickname.encode('ascii'))
            else:
                print(msg)
        except:
            print("[!] error 404\n[#] terminating...")
            client.close()
            break

# send msg to server
def write():
    while True:
        msg = f"{nickname}: {input('')}"
        client.send(msg.encode('ascii'))
```

```
# starting threads for listening & writing
receive_thread = threading.Thread(target=receive)
receive_thread.start()

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


# TCP (lan) based chatting app
# ------------------------------------------------------------
#                       server.py
# ------------------------------------------------------------

# req. libs
import socket
import threading



# global
## connection data
host = '127.0.0.1' # server ip
port = 4444       # free/open port

## starting server
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        # AF_INET - using internal socket, rather than unix socket
        # SOCK_STREAM - using tcp (& not udp)
server.bind((host,port))
server.listen()

## clients & nicknames
clients = []
nicknames = []

print(f"[+] Server is running on {host}:{port}")


# send msg to all clients
def broadcast(msg):
        for client in clients:
                client.send(msg)


# handling clients
def handle(client):
        while True:
                try:
                        msg = client.recv(1024)
```

```
                broadcast(msg)
            except:
                # disconnecting inactive clients
                index = clients.index(client)

                # del f/ clients
                try: clients.remove(index)
                except: pass
                client.close()

                # del f/ nicknames
                nickname = nicknames.index(index)
                nicknames.remove(nickname)

                # display
                print(f"[-] '{nickname}' left")
                break


# listening
def receive():
    while True:
        # accept conn
        client, addr = server.accept()
        print(f"[+] '{str(addr)}' connected")

        # request & store nickname
        client.send('NICKNAME'.encode('ascii'))
        nickname = client.recv(1024).decode('ascii')

        # store
        clients.append(client)
        nicknames.append(nickname)

        # display
        print(f"[+] '{nickname}' connected")

        # send msg
        broadcast(f"[#] {nickname} connected to the server\n".encode('ascii'))

        # handle
        client_handler = threading.Thread(target=handle, args=(client,))
        client_handler.start()


receive()
```

Output:

```
python cns8_server.py
[+] Server is running on 127.0.0.1:4444
[+] '('127.0.0.1', 64211)' connected
[+] 'Anvesha' connected

python cns8_client.py
[#] connecting to 127.0.0.1:4444
[+] enter nickname:Anvesha
[+] connected to 127.0.0.1:4444
[#] Anvesha connected to the server
```