|  |
| --- |
|  |
| Computer Networks Lab\_05 |
| Submitted By: Muhammad Ali Raza Roll No. : 2018-UET-NML-CS-31(1802031) |
|  |
|  |
|  |
|  |

|  |
| --- |
|  |

**commonthread.py File:**

from threading import Thread

import socket

import time

import pickle

SCHEME = "utf-8"

authorization = {}

user\_resourses = {}

class CommonThread(Thread):

    def \_\_init\_\_(self,clientsocket):

        Thread.\_\_init\_\_(self)

        self.clientsocket = clientsocket

    def run(self):

        print("Client Thread Started ... ")

        # self.clientsocket.send("Assalam ulaikum".encode(SCHEME))

        # data = self.clientsocket.recv(1024)

        # time.sleep(2)

        # print("From Client : ", data.decode(SCHEME))

        choice = self.clientsocket.recv(1024)

        choice = choice.decode(SCHEME)

        if(choice == "1"):

            #recieving the username and password

            username = self.clientsocket.recv(1024)

            password = self.clientsocket.recv(1024)

            username = username.decode(SCHEME)

            password = password.decode(SCHEME)

            #setting username and password

            authorization[username] = password

            #printing the username and password from userlist

            print("Users List \n")

            for x , y in authorization.items():

                print(" Username : ",x ,"\n" ,"password : ", y , "\n")

            #assigning resousrses to the user

            resourses = self.clientsocket.recv(4096)

            resourses\_list = pickle.loads(resourses)

            user\_resourses[username] = resourses\_list

            print(user\_resourses)

            self.clientsocket.send("User Added successfully \n".encode(SCHEME))

        elif(choice == "2"):

            #getting the username and password from clients

            user\_name = self.clientsocket.recv(1024)

            password = self.clientsocket.recv(1024)

            match = 0

            #decoding the username and password

            check\_name = user\_name.decode(SCHEME)

            check\_pass = password.decode(SCHEME)

            #verifing from our dictionary

            for users , val in authorization.items():

                if(users == check\_name and val == check\_pass):

                    match = match + 1

            #if any match found then it will authenticate otherwise not.

            print("Verifying Please Wait ... \n")

            time.sleep(2)

            if(match != 0):

                self.clientsocket.send("Authorized User".encode(SCHEME))

            else:

                self.clientsocket.send("Unautherized User".encode(SCHEME))

        elif(choice == "3"):

            res = 0

            #reciving the username and resourse no.

            username = self.clientsocket.recv(1024)

            username = username.decode(SCHEME)

            resourse = self.clientsocket.recv(1024)

            resourse = resourse.decode(SCHEME)

            #assigning the resourse no

            if(resourse == "1"):

                res = 0

            elif(resourse == "2"):

                res = 1

            elif(resourse == "3"):

                res = 2

            else:

                print("Error")

            #checking resourse

            print("Searching Please Wait... \n")

            time.sleep(2)

            for user , val in user\_resourses.items():

                print(user , val)

                if(user == username):

                    if(val[res] == 1):

                        self.clientsocket.send("User has access to the Resourse".encode(SCHEME))

                    else:

                        match = 0

                        self.clientsocket.send("User does not have access to the Resourse".encode(SCHEME))

            #if no matching user found

            self.clientsocket.send("No User Found".encode(SCHEME))

        elif(choice == "4"):

            data = pickle.dumps(authorization)

            self.clientsocket.send(data)

        else:

            print("Error Encountered")

        self.clientsocket.close()

**Multithreadedserver.py File:**

import socket

from threading import Thread

import time

from commonthread import CommonThread

ADDRESS = "127.0.0.1"

PORT = 2222

def main():

    s = socket.socket()

    s.bind((ADDRESS,PORT))

    s.listen(5)

    print("Listing for clients ...")

    while True:

        c , addr = s.accept()

        print("Client Connected :  "  , addr)

        clientThread = CommonThread(c)

        clientThread.start()

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Client.py File:**

from commonthread import SCHEME

import socket

import pickle

ADDRESS = "127.0.0.1"

PORT = 2222

c = socket.socket()

c.connect((ADDRESS,PORT))

print("Connected with server successfully \n")

print(" Press 1 to Add New User  \n Press 2 to verify User \n Press 3 to authenticate User  \n Press 4 to Display Users \n-----------------------------------\n")

choice = input("Enter Choice :  ")

if(choice == "1"):

    resourses = []

    #sending choice

    c.send(choice.encode(SCHEME))

    #adding new username and password and sending it to the server

    username = input("Enter Username :  ")

    password = input("Enter Password :  ")

    c.send(username.encode(SCHEME))

    c.send(password.encode(SCHEME))

    #setting resourses for the user

    print("please enter 1 for yes and 0 for no for resourses to use for the User")

    for i in range(3):

        print("for resourse R",i+1)

        resourse = int(input())

        resourses.append(resourse)

    data = pickle.dumps(resourses)

    c.send(data)

    #receiving the output from server

    data = c.recv(1024)

    print("From Server : " , data.decode(SCHEME))

elif(choice == "2"):

    c.send(choice.encode(SCHEME))

    username = input("Enter Username :  ")

    password = input("Enter Password :  ")

    c.send(username.encode(SCHEME))

    c.send(password.encode(SCHEME))

    #receiving the output from server

    data = c.recv(1024)

    print("From Server : " , data.decode(SCHEME))

elif(choice == "3"):

    c.send(choice.encode(SCHEME))

    username = input("Enter Username :  ")

    c.send(username.encode(SCHEME))

    resourse = input("Which resourse do you want to check \n  Enter R1 , R2 or R3 : ")

    if(resourse == "R1" or resourse == "r1"):

        res = 1

        c.send(str(res).encode(SCHEME))

    elif(resourse == "R2" or resourse == "r2"):

        res = 2

        c.send(str(res).encode(SCHEME))

    elif(resourse == "R3" or resourse == "r3"):

        res = 3

        c.send(str(res).encode(SCHEME))

    data = c.recv(1024)

    print("From Server : " , data.decode(SCHEME))

elif(choice == "4"):

    c.send(choice.encode(SCHEME))

    data = c.recv(4096)

    data\_list = pickle.loads(data)

    print(" ---- Users List --- \n")

    for users , val in data\_list.items():

        print(" UserName = " , users , "\n" , "Password = " , val,  "\n")

else:

    print("Invalid choice")

**-------------------------Tasks-----------------------------**

**Task 01:-**

Adding a user with given credentials and access level to resources

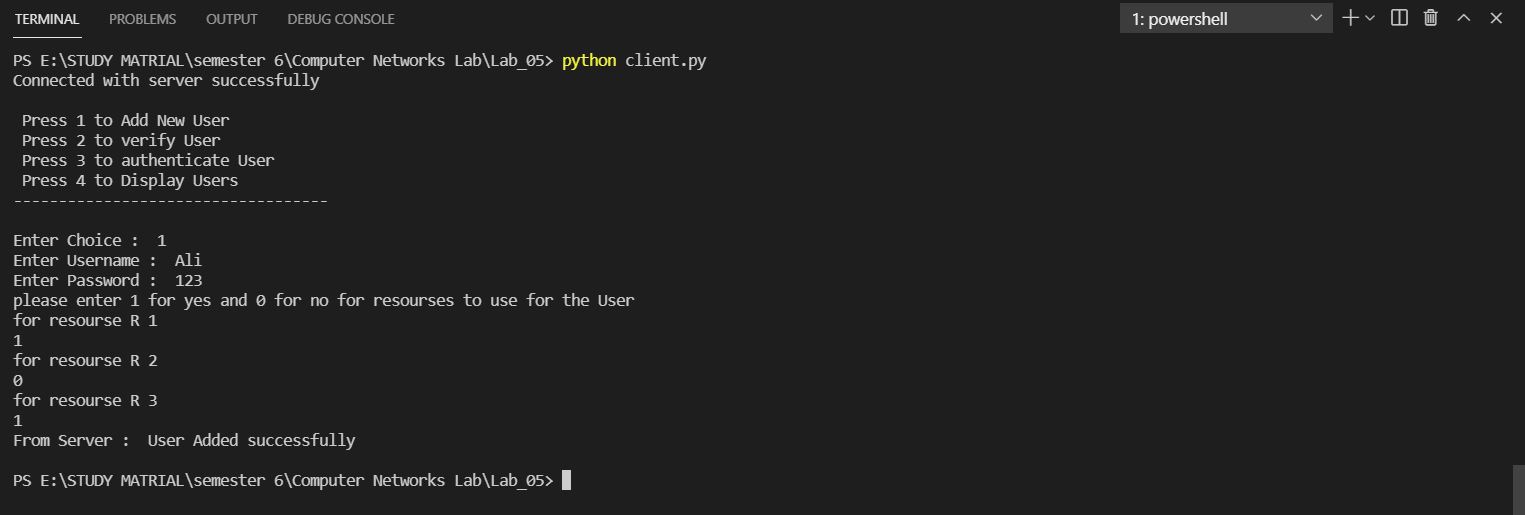


Figure 1 : Adding credentials and access to resources

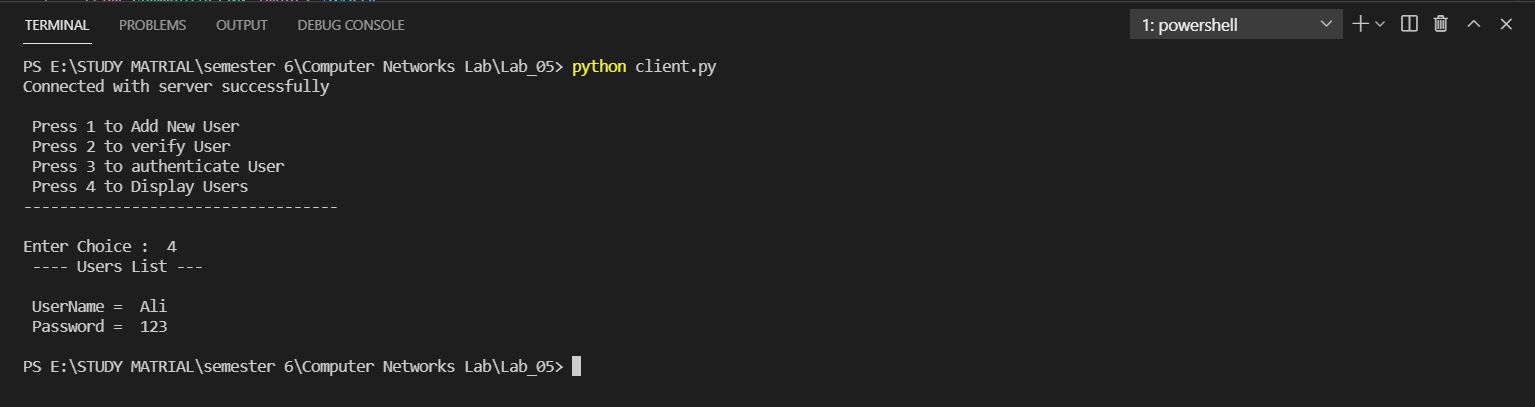


Figure 2 : Display of the users

**Task 02:-**

Validating a username and password



Figure 3 : Verification of a User

**Task 03:-**

Authorization of a user

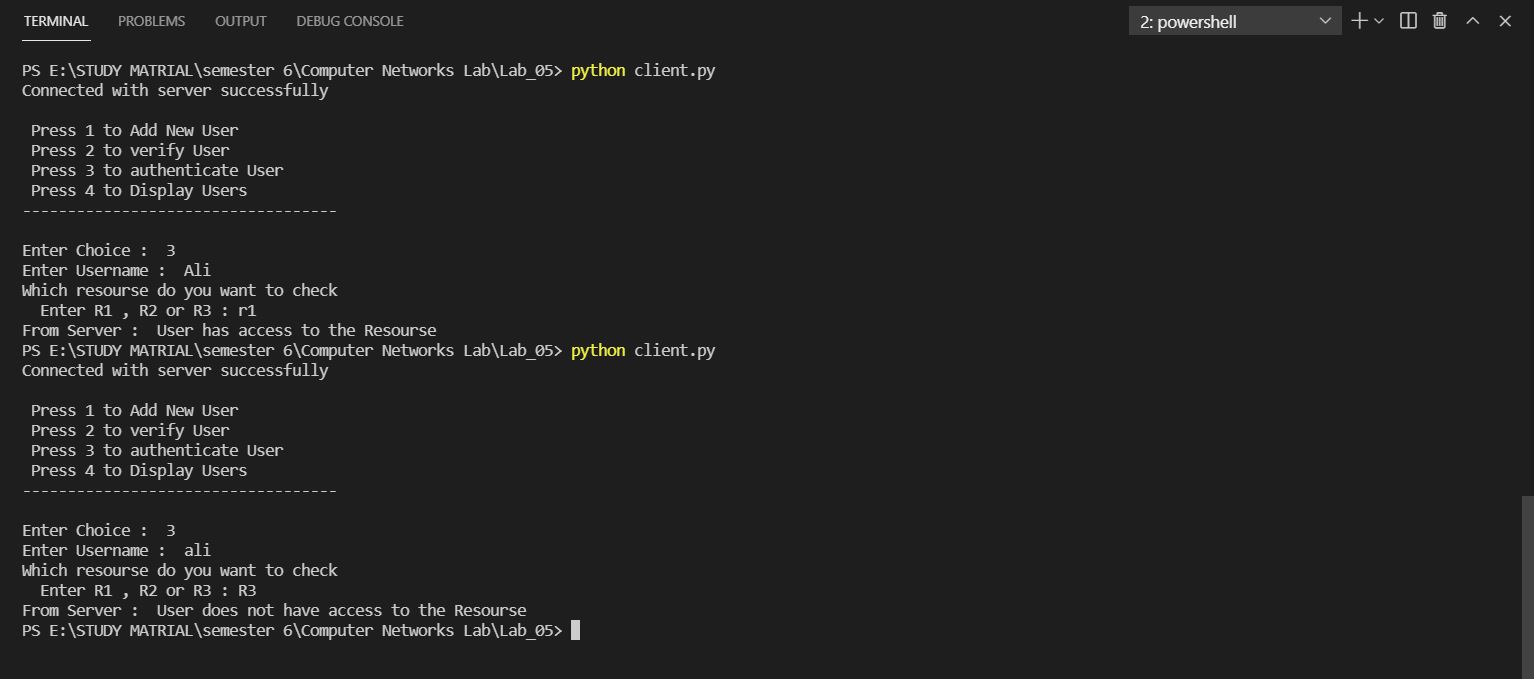


Figure 4 : Authorization of the users for a given resource

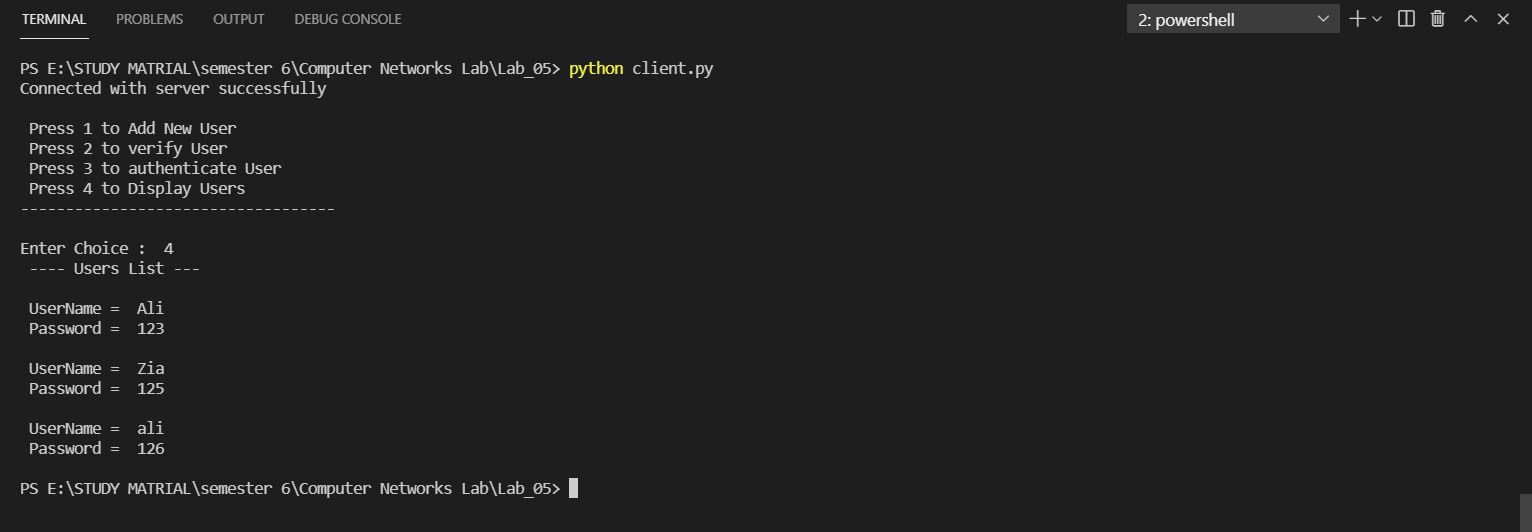


Figure 5 : Display of users