**PES UNIVERSITY**

**(Established under Karnataka Act No. 16 of 2013)**

**100 Ft. Road, BSK III Stage, Bengaluru – 560 085**

| | |
|---|---|
| **Course Title: Computer Networks** | **Course code: UE17CS301** |
| **Semester: 5th semester** | **Section: E** |
| **SRN: PES1201700217** | **PES1201700267** |
| **Name: Madhav Agal** | **Saransh Gupta** |

# ASSIGNMENT REPORT

**Problem Statement:** Building a simple drive storage system through which multiple users can store their files on the server and can fetch them whenever necessary.

**Description:** The drive storage contains the following features:

- Users can log in using a unique username and password.
- Multiple users can log in simultaneously.
- Users can only access the files they stored there.
- Multiple types of files are supported eg. txt, doc, docx, csv etc.

Proper error handling is done for the following expected situations:

- When a new user chooses a username which is already assigned to some other user.
- If the username and password is not valid.
- If a user tries to access a file he has not uploaded.
- If the pathname of the file is not valid.

**PES UNIVERSITY**
**(Established under Karnataka Act No. 16 of 2013)**
**100 Ft. Road, BSK III Stage, Bengaluru – 560 085**

The drive storage follows the following procedure:

- Firstly, it asks whether the user is new or existing.
- If the user is new, he has to enter a new username and password.
- If he's already an existing user, he has to enter his username and password.
- If the username and password is valid the user is redirected to his storage drive.
- Now, he's asked when he wants to store a new file or fetch an already existing from the drive.
- In case the user wants to store a new file, he has to enter the full name and correct path of the file.
- In case he chooses otherwise, he will be shown a dictionary of files available in his directory from he can fetch the file he wants.

## Packages used:

- socket
- json
- pickle
- time
- sys

## Link to the demo video:

https://drive.google.com/open?id=1KIUvGDFyfWfZ1IfY1i1K6AcKARv1TdVf

## Code:

Client.py

```
import socket
import json
import pickle
import time
import sys
#import threading

def func(s,usr):
        print("Your file directory on the server is -\n")
        dirc = pickle.loads(s.recv(1024))
        print(dirc)
        print("Enter 0 for storing new file on the server or the file number for fetching
the file from the server")
        fno = int(input())
        while(fno not in range(len(list(dirc.keys()))+1)):
                        print("\nPlease enter a valid number: ")
                        fno = int(input())
        s.send(str(fno).encode('ascii'))
        if(fno==0):
                #Lock = threading.RLock()
                #Lock.acquire()
                print("Enter the name of the file")
                fname = input()
                print("Enter the full path of the file")
```

```
                fpath = input()
                sendfile(s,usr,fname,fpath)
                #Lock.release()
        else:
                fname = (s.recv(1024)).decode('ascii')
                f = open(fname,'wb')
                l = s.recv(1024)
                while (l):
                        f.write(l)
                        l = s.recv(1024)
                        print('Recieving data...')
                f.close()
                print("Data received")

def sendfile(s,usr,fname,fpath):



        s.send(fname.encode('ascii'))
        time.sleep(0.5)
        #s.send(fpath.encode('ascii'))
        try:
                f = open(fpath,'rb')
                l = f.read(1024)

                while(l):
                        s.send(l)
                        l = f.read(1024)
                        print('Sending data...')
                f.close()
                print("Data sent")

        except:
                print('Please enter the correct path')
```

```
                #sys.exit()


def Main():
        # local host IP '127.0.0.1'
        host = '127.0.0.1'

        # Define the port on which you want to connect
        port = 12346
        s = socket.socket(socket.AF_INET,socket.SOCK_STREAM)

        # connect to server on local computer
        s.connect((host,port))

        # message you send to server
        print("Connected to the drive server\n")
        while True:
                print("Enter 0 for new user, 1 for existing user: ")
                n = int(input())
                while(n not in [0,1]):
                        print("\nPlease enter 0 or 1: ")
                        n = int(input())
                s.send(str(n).encode('ascii'))
                print()
                if(n==0):
                        print("Enter username: ")
                        usr = input()
                        print("\nEnter password: ")
                        pas = input()
                        print("\nConfirm password: ")
                        pas2 = input()
                        l = [usr,pas,pas2]
                        s.send(pickle.dumps(l))
                        reply = str((s.recv(1024)).decode('ascii'))
```

```
                print()
                print(reply,'\n')
                if(reply == "Account successfully created!"):
                        print("Logged in as ",usr,'\n')
                        func(s,usr)
                        break

        elif(n==1):
                print("Enter username: ")
                usr = input()
                print("\nEnter password: ")
                pas = input()
                l = [usr,pas]
                s.send(pickle.dumps(l))
                reply = str((s.recv(1024)).decode('ascii'))
                print()
                print(reply,'\n')
                if(reply == "Login successful!"):
                        print("Logged in as ",usr,'\n')
                        func(s,usr)
                        break
        break
        # message sent to server
        #s.send(message.encode('ascii'))
        # messaga received from server
        #data = str((s.recv(1024)).decode('ascii'))

        # print the received message
        # here it would be a reverse of sent message

    # close the connection
    s.close()

if __name__ == '__main__':
```

Main()

## Server.py

```
# import socket programming library
import socket
import pickle
# import thread module
from _thread import *
import threading
import time
import sys
#print_lock = threading.Lock()
try:
        users_database = pickle.load( open( "udb.p", "rb" ) )
except:
        users_database = dict()
try:
        users_files = pickle.load( open( "ufiles.p", "rb" ) )
except:
        users_files = dict()
# thread fuction
def login(c):
        n = int((c.recv(24)).decode('ascii'))
        if(n==0):
                l = pickle.loads(c.recv(1024))
                usr = str(l[0])
                pas = str(l[1])
                pas2 = str(l[2])
                if((pas == pas2) and (usr not in users_database.keys())):
                        message = "Account successfully created!"
```

```
                        c.send(message.encode('ascii'))
                        users_database[usr] = pas
                        users_files[usr] = dict()
                        return usr
                elif(usr in users_database.keys()):
                        message = "Username already taken.\nClosing the connection.."
                        c.send(message.encode('ascii'))
                        c.close()
                elif(pas != pas2):
                        message = "The passwords do not match.\nClosing the
connection.."
                        c.send(message.encode('ascii'))
                        c.close()

        elif(n==1):
                l = pickle.loads(c.recv(1024))
                usr = str(l[0])
                pas = str(l[1])
                if(usr not in users_database.keys()):
                        message = "User does not exist!\nClosing the connection..."
                        c.send(message.encode('ascii'))
                        c.close()
                elif(users_database[usr] != pas):
                        message = "Wrong password!\nClosing the connection..."
                        c.send(message.encode('ascii'))
                        c.close()
                elif(users_database[usr] == pas):
                        message = "Login successful!"
                        c.send(message.encode('ascii'))
                        return usr


def func(usrn,c):
        time.sleep(1)
```

```
            c.send(pickle.dumps(users_files[usrn]))
            fno = int((c.recv(24)).decode('ascii'))
            if(fno == 0):
                    fname = (c.recv(1024)).decode('ascii')
                    #time.sleep(1)
                    file = usrn+fname
                    f = open(file,'wb')
                    l = c.recv(1024)
                    while (l):
                            f.write(l)
                            l = c.recv(1024)
                    f.close()
                    m = max(len(list(users_files[usrn].keys()))+1,1)
                    users_files[usrn][m] = fname
                    print("Recieved file ",fname," from user - ",usrn)
                    c.close()
            else:
                    fname = users_files[usrn][fno]
                    c.send(fname.encode('ascii'))
                    time.sleep(0.5)
                    f = open(fname,'rb')
                    l = f.read(1024)
                    while (l):
                            c.send(l)
                            l = f.read(1024)
                    f.close()
                    print("Sent file ",fname," to user - ",usrn)
                    c.close()


            #print('fno - ',fno,' fname - ',fname,' fpath - ',fpath)

    def threaded(c,addr):
            usrn = login(c)
```

```
        if(usrn != None):
                print(addr[0],':',addr[1],' logged in as ',usrn)
                pickle.dump( users_database, open( "udb.p", "wb" ) )
                pickle.dump( users_files, open( "ufiles.p", "wb" ) )
                res = func(usrn,c)
        else:
                print(addr[0],':',addr[1],' failed to login')

def Main():
        host = ""

        # reverse a port on your computer
        # in our case it is 12345 but it
        # can be anything
        port = 12346
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        s.bind((host, port))
        print("socket binded to port", port)

        # put the socket into listening mode
        s.listen(5)
        print("socket is listening")

        # a forever loop until client wants to exit
        while True:

                # establish connection with client
                c, addr = s.accept()

                # lock acquired by client
                #print_lock.acquire()
                print('Connected to :', addr[0], ':', addr[1])

                # Start a new thread and return its identifier
```

```
                start_new_thread(threaded, (c,addr))
        s.close()


if __name__ == '__main__':
Main()
```