Simulation Of Client Server File Sharing System

REPORT BY :

Anto Joel .V
95072012011

Simulation Of Client Server File Sharing System

Aim

To create a simulation of a client server based file transfer system using python

Objectives of the project

To create a file transfer system by implementing the networking techniques in client server based models which can transfer files from client to server.

Problem statement

The problem is the complication of sharing files among the devices which are highly looped and stimulated. Instead we create a most direct approach of file sharing within a system.

Proposed system

In this system we created a method of sharing the file among the with the client server based model which is the most direct approach and the most time efficient approach this results in making the file transfer protocol easy. The method of file sharing is encrypted which makes it secure. In this system we used Socket programming in python to implement the file transfer system. Since this is a inbuilt package dedicated for this approach we use Socket to create this client server file transfer system

Software & hardware requirements

- Python
- Socket
- Command prompt /Powershell
- WifI

Module or code

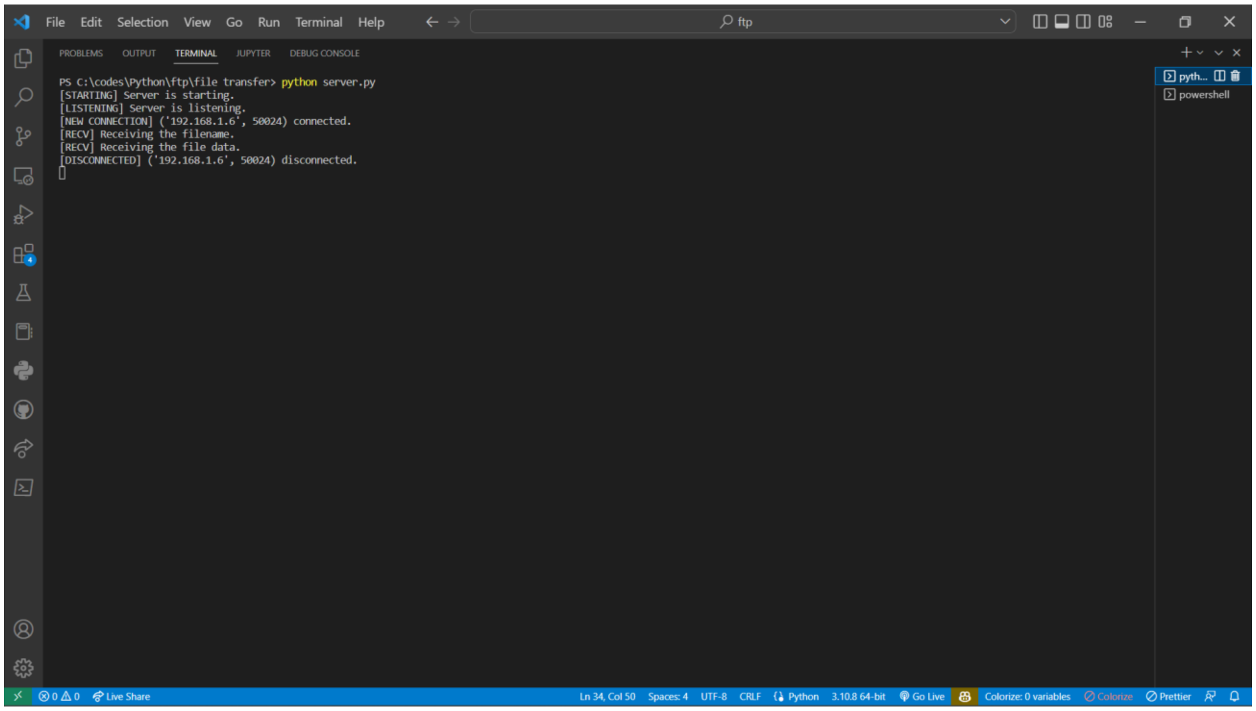
Client.py

```
import socket
IP = socket.gethostbyname(socket.gethostname())
PORT = 4455
ADDR = (IP, PORT)
FORMAT = "utf-8"
SIZE = 1024

def main():
    """ Starting a TCP socket. """
    client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    """ Connecting to the server. """
    client.connect(ADDR)

    """ Opening and reading the file data. """
    file = open("data/file.txt", "r")
    data = file.read()

    """ Sending the filename to the server. """
    client.send("file.txt".encode(FORMAT))
    msg = client.recv(SIZE).decode(FORMAT)
    print(f"[SERVER]: {msg}")

    """ Sending the file data to the server. """
    client.send(data.encode(FORMAT))
    msg = client.recv(SIZE).decode(FORMAT)
    print(f"[SERVER]: {msg}")

    """ Closing the file. """
    file.close()

    """ Closing the connection from the server. """
    client.close()

if __name__ == "__main__":
        main()
```

Server.py

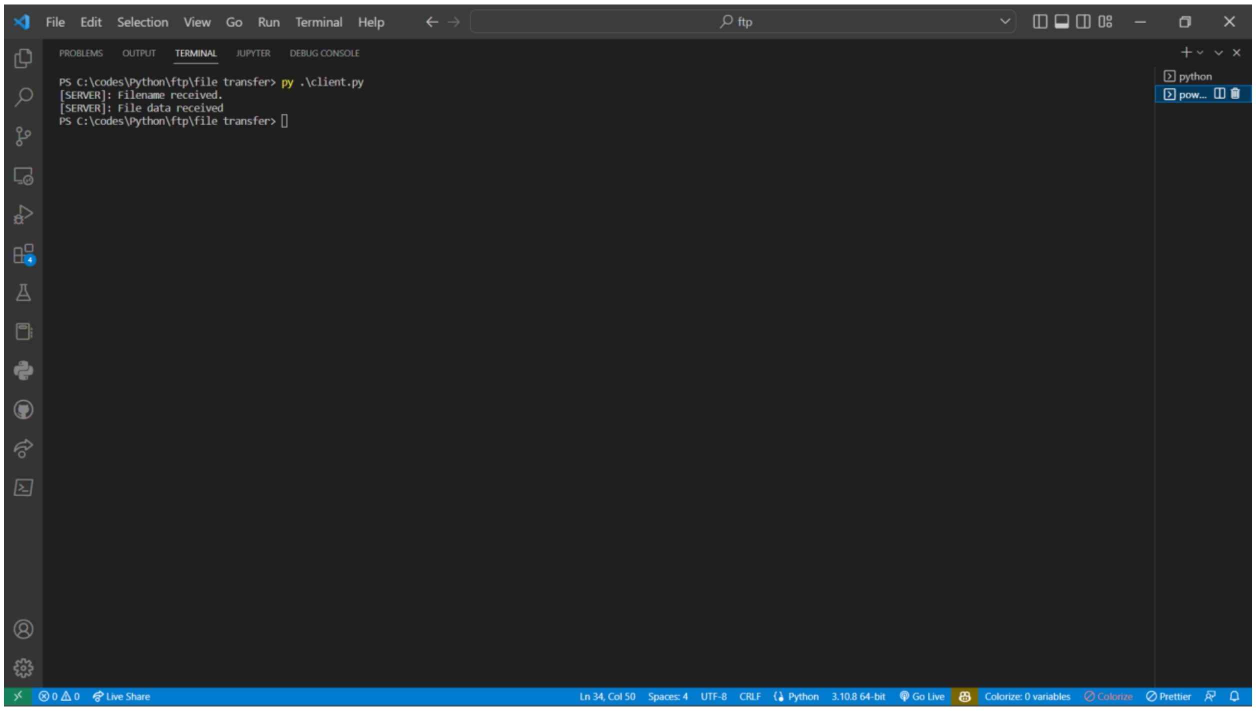
```
import socket

IP = socket.gethostbyname(socket.gethostname())
PORT = 4455
ADDR = (IP, PORT)
SIZE = 1024
FORMAT = "utf-8"

def main():
    print("[STARTING] Server is starting.")
    """ Staring a TCP socket. """
    server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    """ Bind the IP and PORT to the server. """
    server.bind(ADDR)

    """ Server is listening, i.e., server is now waiting for the client to connected. """
    server.listen()
    print("[LISTENING] Server is listening.")

    while True:
        """ Server has accepted the connection from the client. """
        conn, addr = server.accept()
        print(f"[NEW CONNECTION] {addr} connected.")

        """ Receiving the filename from the client. """
        filename = conn.recv(SIZE).decode(FORMAT)
        print(f"[RECV] Receiving the filename.")
        file = open(filename, "w")
        conn.send("Filename received.".encode(FORMAT))

        """ Receiving the file data from the client. """
        data = conn.recv(SIZE).decode(FORMAT)
        print(f"[RECV] Receiving the file data.")
        file.write(data)
        conn.send("File data received".encode(FORMAT))

        """ Closing the file. """
        file.close()

        """ Closing the connection from the client. """
        conn.close()
        print(f"[DISCONNECTED] {addr} disconnected.")

if __name__ == "__main__":
    main()
```

Screenshot



```
PS C:\codes\Python\ftp\file transfer> python server.py
[STARTING] Server is starting.
[LISTENING] Server is listening.
[NEW CONNECTION] ('192.168.1.6', 50024) connected.
[RECV] Receiving the filename.
[RECV] Receiving the file data.
[DISCONNECTED] ('192.168.1.6', 50024) disconnected.
```

File Edit Selection View Go Run Terminal Help
ftp
PROBLEMS OUTPUT TERMINAL JUPYTER DEBUG CONSOLE
pyth...
powershell
Live Share
Ln 34, Col 50 Spaces: 4 UTF-8 CRLF Python 3.10.8 64-bit Go Live Colorize: 0 variables Colorize Prettier


File Edit Selection View Go Run Terminal Help
ftp
PROBLEMS OUTPUT TERMINAL JUPYTER DEBUG CONSOLE
python
pow...
Live Share
Ln 34, Col 50 Spaces: 4 UTF-8 CRLF Python 3.10.8 64-bit Go Live Colorize: 0 variables Colorize Prettier

```
PS C:\codes\Python\ftp\file transfer> py .\client.py
[SERVER]: Filename received.
[SERVER]: File data received
PS C:\codes\Python\ftp\file transfer> []
```

Conclusion future enhancement

Thus the project is developed as the first phase in the future. We are planning to develop this into a full fledged application which includes GUI, Database and which will be also used in all form devices like laptops, desktops and mobile which also includes android ios with high security standards.