

Nama : Alvin Zanu Putra

NRP : 5025231064

Kelas : Pemrograman Jaringan – D

## TUGAS 3

Link Github Folder TUGAS3 :

<https://github.com/alvinzanuaputra/PROGJAR24/tree/main/assets/src/TUGAS3>

### Ringkasan :

Menambahkan dua kemampuan baru yaitu operasi **upload file** dan **hapus file**. Pada operasi upload, isi file yang akan dikirim ke server terlebih dahulu diencode ke dalam format **Base64** untuk memastikan file dapat dikirim dalam bentuk teks melalui protokol yang ada. Sedangkan operasi hapus file berfungsi untuk menghapus file tertentu yang ada di server sesuai dengan nama file yang dikirimkan oleh client.. Perintah upload mencakup nama file dan konten file yang sudah dalam bentuk Base64, sementara perintah hapus hanya membutuhkan nama file yang akan dihapus. Selain itu, implementasi client untuk kedua operasi ini ditambahkan pada file **file\_interface.py**, dan server diperbarui di **file\_server.py** untuk menangani perintah baru tersebut. Pengujian dilakukan dengan menjalankan komunikasi client-server.

### Jawaban :

#### A. Pertama kita akan membuat fungsi HAPUS di file

```
def hapus(self, params=[]):
    try:
        filename = params[0]
        if not os.path.exists(filename):
            return dict(status='ERROR', data=f'File "{filename}" tidak ditemukan dalam direktori')
        os.remove(filename)
        return dict(status='OK', data=f'{filename} berhasil dihapus dari server')
    except Exception as e:
        return dict(status='ERROR', data=str(e))
```

Pertama-tama, fungsi memeriksa apakah file tersebut benar-benar ada di direktori dengan `os.path.exists()`. Jika file tidak ditemukan, maka fungsi langsung membalas dengan status `ERROR` dan pesan bahwa file tidak tersedia, sehingga tidak akan terjadi kesalahan lebih lanjut saat mencoba menghapus file yang tidak ada.

Jika file ditemukan, maka fungsi menggunakan `os.remove()` untuk menghapus file tersebut dari sistem file. Setelah berhasil dihapus, fungsi mengembalikan respons dengan status `OK` dan pesan bahwa file telah berhasil dihapus dari server. Sama seperti fungsi upload, fungsi hapus juga dibungkus dalam blok `try-except` untuk menangani kemungkinan kesalahan seperti error akses file

#### B. Selanjutnya membuat fungsi UPLOAD file ke server :

```
def upload(self, params=[]):
    try:
        filename, filecontent = params[0], params[1]
        if os.path.exists(filename):
            return dict(status='ERROR', data=f'File "{filename}" sudah tersedia dalam direktori')
        with open(filename, 'wb') as fp:
            fp.write(base64.b64decode(filecontent.encode()))
        return dict(status='OK', data=f'{filename} berhasil diupload ke server')
    except Exception as e:
        return dict(status='ERROR', data=str(e))
```

Menerima parameter berupa nama file (filename) dan isi file dalam bentuk string yang sudah di-*encode* ke format Base64 (filecontent). Sebelum menyimpan file, fungsi melakukan pengecekan menggunakan `os.path.exists()` untuk memastikan file nama itu ada di direktori. Jika file ada, mengembalikan respons error dengan pesan file sudah tersedia dan tidak akan menimpa file lama.

Jika file belum ada, isi file akan didekode dari format Base64 menjadi biner menggunakan `base64.b64decode()`, kemudian disimpan ke dalam file dengan nama yang diberikan. File dibuka dalam mode `wb` (write binary) agar bisa menampung data biner hasil decoding. Jika semua proses berjalan tanpa kesalahan, fungsi akan mengembalikan status OK. Namun jika terjadi kesalahan decoding gagal atau masalah lain, maka pengecualian akan ditangkap dengan status ERROR

#### C. Dokumen update protocol bertambah fungsi yaitu hapus dan upload beserta validasi nya

##### *FILE SERVER*

*TUJUAN: melayani client dalam request file server*

##### *ATURAN PROTOKOL:*

- client harus mengirimkan request dalam bentuk string
- string harus dalam format:  
*REQUEST spasi PARAMETER1 spasi PARAMETER2 dst.*
- Semua result akan diberikan dalam bentuk JSON
- Diakhiri dengan karakter ASCII "`\r\n\r\n`"

##### *REQUEST YANG DILAYANI:*

##### *LIST*

\* *TUJUAN: Mendapatkan daftar seluruh file yang tersedia di direktori server*

\* *PARAMETER: tidak ada*

\* *RESULT:*

- *BERHASIL:*
  - *status: OK*
  - *data: list file*
- *GAGAL:*
  - *status: ERROR*
  - *data: pesan kesalahan*

##### *GET*

\* *TUJUAN: Mengambil isi file dengan nama tertentu*

\* *PARAMETER:*

- *PARAMETER1: nama file*

\* *RESULT:*

- *BERHASIL:*
  - *status: OK*
  - *data\_namafile: nama file*
  - *data\_file: isi file (dalam base64)*
- *GAGAL:*
  - *status: ERROR*
  - *data: pesan kesalahan*

##### *UPLOAD*

\* *TUJUAN: Mengunggah file ke server*

\* *PARAMETER:*

- *PARAMETER1: nama file yang akan disimpan*



```
def run(self):
    while True:
        data = self.connection.recv(8096)
        if data:
            d = data.decode()
            hasil = fp.proses_string(d)
            hasil=hasil+"\r\n\r\n"
            self.connection.sendall(hasil.encode())
        else:
            break
    self.connection.close()
```

oleh karena itu receive server harus diubah size nya yang awalnya 32 menjadi 4096 – 8096 tergantung kebutuhan.

```
(base) jovyan@9343b9693c1c:~/work/progjar/progjar4a$ python3 file_server.py
WARNING:root:server berjalan di ip address ('0.0.0.0', 56666)
WARNING:root:connection from ('172.16.16.102', 47758)
WARNING:root:string di proses: LIST
WARNING:root:memproses request: LIST
WARNING:root:string di proses: list
WARNING:root:memproses request: list
WARNING:root:string di proses: get kenekjane.txt
WARNING:root:memproses request: get
WARNING:root:string di proses: hapus oke.txt
WARNING:root:memproses request: hapus
WARNING:root:string di proses: hapus oke.txt
WARNING:root:memproses request: hapus
WARNING:root:string di proses: upload mclaren.txt YWt1IH8pbmdpbGlhbnR5YSB5YW1ib3JnaGluaSAzMDAK
(base) jovyan@9343b9693c1c:~/work/progjar/progjar4a/files$ ls -lh
total 580K
-rw-r--r-- 1 jovyan users 17 May 23 14:19 alvin.txt
-rwxr-xr-x 1 jovyan users 11K May 23 13:30 donalbebek.jpg
-rw-r--r-- 1 jovyan users 18 May 23 14:12 kenekjane.txt
-rw-r--r-- 1 jovyan users 26 May 23 14:07 oke.txt
-rwxr-xr-x 1 jovyan users 16K May 23 13:30 pokijan.jpg
-rwxr-xr-x 1 jovyan users 538K May 23 13:30 rfc2616.pdf
(base) jovyan@9343b9693c1c:~/work/progjar/progjar4a/files$ ls -lh
total 580K
-rw-r--r-- 1 jovyan users 17 May 23 14:19 alvin.txt
-rwxr-xr-x 1 jovyan users 11K May 23 13:30 donalbebek.jpg
-rw-r--r-- 1 jovyan users 18 May 23 14:12 kenekjane.txt
-rw-r--r-- 1 jovyan users 33 May 23 14:28 mclaren.txt
-rwxr-xr-x 1 jovyan users 16K May 23 13:30 pokijan.jpg
-rwxr-xr-x 1 jovyan users 538K May 23 13:30 rfc2616.pdf
(base) jovyan@9343b9693c1c:~/work/progjar/progjar4a/files$ cat mclaren.txt
aku pingin punya lamborghini 100
(base) jovyan@9343b9693c1c:~/work/progjar/progjar4a/files$
```

Server berhasil dijalankan di alamat 0.0.0.0 dan port 56666, serta menerima koneksi dari client dengan IP 172.16.16.102. Server mencatat semua perintah yang masuk dengan format log Python. Client berkomunikasi menggunakan nc (netcat) untuk mengirimkan perintah seperti list, get, hapus, dan upload. Saat client mengirim perintah LIST (huruf kapital), server merespon dengan "request tidak dikenali", menunjukkan bahwa saat string\_datafile dikirim format lower sudah dihapus. Setelah perintah list, server berhasil mengembalikan daftar file yang ada. Selanjutnya, client mencoba mengambil file kenekjane.txt dengan perintah get kenekjane.txt, dan server berhasil mengirimkan isi file dalam format base64. Ketika file oke.txt dihapus menggunakan perintah hapus oke.txt, server merespons sukses. Namun, saat perintah yang sama dikirim lagi, server menolak karena file tersebut sudah tidak ada, dan mengembalikan pesan error yang sesuai. Hal serupa juga terjadi pada perintah upload: ketika client mengunggah file mclaren.txt untuk pertama kalinya, server berhasil menyimpannya. Tetapi saat perintah yang sama diulang dengan isi file yang sama, server menolak dengan pesan bahwa file sudah tersedia.

tcp.port == 56666    udp.port == 56666							tcp.port == 56666    udp.port == 56666						
No.	Time	Source	Destination	Protocol	Length	Info	No.	Time	Source	Destination	Protocol	Length	Info
43973	5126.1478788	172.16.16.101	172.16.16.102	TCP	136	56666 → 47758 [PSH, ACK] Seq=	46421	5129.2926585	172.16.16.101	172.16.16.102	TCP	136	56666 → 47758 [PSH, ACK] Seq=
43975	5126.1479403	172.16.16.102	172.16.16.101	TCP	68	47758 → 56666 [ACK] Seq=43 Ack=	46422	5129.2926788	172.16.16.102	172.16.16.101	TCP	68	47758 → 56666 [ACK] Seq=43 Ack=
43143	5137.1467642	172.16.16.102	172.16.16.101	TCP	82	47758 → 56666 [PSH, ACK] Seq=4	46534	5131.2915026	172.16.16.102	172.16.16.101	TCP	82	47758 → 56666 [PSH, ACK] Seq=4
43144	5137.1471128	172.16.16.101	172.16.16.102	TCP	151	56666 → 47758 [PSH, ACK] Seq=5	46537	5131.2918872	172.16.16.101	172.16.16.102	TCP	151	56666 → 47758 [PSH, ACK] Seq=5
43145	5137.1471695	172.16.16.102	172.16.16.101	TCP	68	47758 → 56666 [ACK] Seq=57 Ack=	46538	5131.2918966	172.16.16.102	172.16.16.101	TCP	68	47758 → 56666 [ACK] Seq=57 Ack=
43581	5209.2181466	172.16.16.102	172.16.16.101	TCP	132	47758 → 56666 [PSH, ACK] Seq=5	47889	5203.3628773	172.16.16.102	172.16.16.101	TCP	132	47758 → 56666 [PSH, ACK] Seq=5
43583	5209.2190397	172.16.16.101	172.16.16.102	TCP	139	56666 → 47758 [PSH, ACK] Seq=4	47893	5203.3628294	172.16.16.101	172.16.16.102	TCP	139	56666 → 47758 [PSH, ACK] Seq=4
43584	5209.2191265	172.16.16.102	172.16.16.101	TCP	68	47758 → 56666 [ACK] Seq=121 Ack=	47894	5203.3638468	172.16.16.102	172.16.16.101	TCP	68	47758 → 56666 [ACK] Seq=121 Ack=
43860	5237.4021736	172.16.16.102	172.16.16.101	TCP	132	47758 → 56666 [PSH, ACK] Seq=2	47839	5231.5468850	172.16.16.102	172.16.16.101	TCP	132	47758 → 56666 [PSH, ACK] Seq=2
43861	5237.4022039	172.16.16.101	172.16.16.102	TCP	154	56666 → 47758 [PSH, ACK] Seq=2	47342	5231.5476851	172.16.16.101	172.16.16.102	TCP	154	56666 → 47758 [PSH, ACK] Seq=2
43862	5237.4022763	172.16.16.102	172.16.16.101	TCP	68	47758 → 56666 [ACK] Seq=185 Ack=	47343	5231.5477034	172.16.16.102	172.16.16.101	TCP	68	47758 → 56666 [ACK] Seq=185 Ack=
[Time since reference or first frame: 5237.402889862 seconds]							[Time since reference or first frame: 5231.546885040 seconds]						
Frame Number: 43861							Frame Number: 47339						
Frame Length: 154 bytes (1232 bits)							Frame Length: 132 bytes (1056 bits)						
Capture Length: 154 bytes (1232 bits)							Capture Length: 132 bytes (1056 bits)						
[Frame is marked: False]							[Frame is marked: False]						
[Frame is ignored: False]							[Frame is ignored: False]						
[Protocols in frame: ssl:ethertype:ip:tcp:data]							[Protocols in frame: ssl:ethertype:ip:tcp:data]						
[Coloring Rule Name: TCP]							[Coloring Rule Name: TCP]						
[Coloring Rule String: tcp]							[Coloring Rule String: tcp]						
Linux cooked capture v1							Linux cooked capture v1						
Internet Protocol Version 4, Src: 172.16.16.101, Dst: 172.16.16.102							Internet Protocol Version 4, Src: 172.16.16.102, Dst: 172.16.16.101						
Transmission Control Protocol, Src Port: 56666, Dst Port: 47758, Seq: 494, Ack: 185, Len: 86							Transmission Control Protocol, Src Port: 56666, Dst Port: 47758, Seq: 121, Ack: 494, Len: 64						
Data (86 bytes)							Data (64 bytes)						
Data: 7b22737461747573223a2022455252223a2022264617461223a202246696c65205223							Data: 75706c6164206d636c61726556e26747874206957743149484270626d64706269427764						
0000 00 04 00 01 00 06 02 42 ac 10 10 65 3a 20 08 00	-----B---e:--						0000 00 04 00 01 00 06 02 42 ac 10 10 66 02 42 08 00	-----B---f:B-					
0010 45 00 00 8a f6 d1 40 00 40 06 ca b0 ac 10 10 65	E-----@ @-----e						0010 45 00 00 74 3b c9 40 00 40 06 88 cf ac 10 10 66	E-t8@ @-----f					
0020 ac 10 10 66 dd 5a ba 8e 18 7e 1a 79 63 58 2d bf	---f-Z---yck---						0020 ac 10 10 65 ba 8e dd 5a 63 58 2d 7f 18 7e 1a 79	---e---Z cX-----y					
0030 80 18 61 fe 79 68 00 00 01 01 08 0a 0b a4 f7 a4	---yh-----						0030 80 18 01 f6 79 52 00 00 01 01 08 0a 97 2b 8a e4	---yR-----					
0040 97 2b 8a e4 f7 2d 7f 18 7e 1a 79 63 58 2d bf	---f-Z---yck---						0040 0b a4 89 8c 75 70 6c 67 61 64 20 60 83 6c 61 72	---bD10 ad mclar					
0050 45 92 52 4f 52 22 2c 20 22 64 61 74 61 22 3a 20	ERROR- "data"						0050 57 35 35 59 53 42 70 65	---b-t8 Y w11n3pa					
0060 22 46 69 6c 65 20 5c 22 6d 83 6c 61 72 65 6a 20	"File \"mclaren.						0060 6d 64 70 62 69 42 77 64 57 35 35 59 53 42 70 65	---mphiBwd w55Y5B-v					
0070 74 70 74 5c 22 20 72 75 64 61 68 20 74 65 72 72	txt\" su dah ters						0070 57 31 69 62 33 4a 6e 61 47 6c 75 61 53 41 78 40	---w11n3Jna G1u5A-M					
0080 65 64 69 61 20 64 61 6c 61 00 20 64 69 72 65 6b	edia dal an direk						0080 24 41 4b 0a	DAK-					
0090 74 6f 72 69 22 7d 0d 0a 0d 0a	tori\" ..												

## Analisis wireshark :

Hasil tangkapan paket menggunakan Wireshark yang memperlihatkan komunikasi TCP antara client (172.16.16.102) dan server (172.16.16.101) pada port 56666. Pada sisi kanan gambar, terlihat bahwa client mengirimkan request upload mclaren.txt beserta isi file dalam format base64. Data ini dikirim dari port client 47758 menuju port server 56666 dan terlihat dengan jelas dalam bentuk payload string di bagian bawah (upload mclaren.txt <data base64>). Di sisi kiri gambar, server memberikan respon dengan pesan JSON yang menunjukkan bahwa file mclaren.txt sudah ada: "status":"ERROR", "data":"File \"mclaren.txt\" sudah tersedia dalam direktori". Respon ini memperlihatkan bahwa validasi pada fungsi upload berjalan dengan baik, dan protokol aplikasi berhasil menangani kondisi di mana file sudah pernah diunggah sebelumnya. Paket TCP tersebut mengandung payload berformat string JSON, yang merupakan bagian dari protokol aplikasi berbasis teks yang dikembangkan. Proses Push (PSH, ACK) menunjukkan bahwa data dikirim langsung tanpa menunggu buffer penuh.