

Program File Transfer Protocol (FTP) menggunakan Metode XMLRPC Pada Sistem Terdistribusi

Laporan Ini Diajukan untuk Memenuhi Syarat Kelulusan Mata Kuliah Sistem Paralel dan Terdistribusi Pada Program Studi Informatika Jenjang Pendidikan S-1 Universitas Telkom



Disusun Oleh

1. Sya Raihan Heggi (1301184219)
2. Muhammad Raihan Muhith (1301184245)
3. Mohammad Dwiantara Mahardhika (1301184467)
4. Fariz Muhammad Rizky (1301184327)

Program Studi S1 Informatika

Fakultas Informatika

Universitas Telkom

2021

DAFTAR ISI

BAB I : PENDAHULUAN	3
BAB II : ANALISIS	4
Alasan Pemilihan RPC	4
Definisi RPC	4
Karakteristik RPC	4
Kelebihan RPC	4
Kekurangan RPC	4
Model Sistem	5
BAB III : PERANCANGAN	6
UPLD File_Path	9
LIST	10
CLN	11
DWLD	13
QUIT	13
LOGIN	14
BAB III : IMPLEMENTASI	15

BAB I : PENDAHULUAN

FTP (File Transfer Protocol) adalah internet service yang dirancang untuk membuat sambungan ke server internet tertentu atau komputer, sehingga user dapat mengirimkan file ke komputer (*download*) atau mengirimkan file ke server (*upload*). FTP saat ini banyak digunakan untuk melakukan pertukaran data, karena lebih mudah daripada menggunakan perangkat kabel atau fisik. Program yang kami bangun merupakan program FTP sederhana yang memiliki menu download, upload, lihat file yang ada di server, dan lihat client teraktif. Client teraktif ditentukan berdasarkan seberapa sering client tersebut melakukan aktifitas download dan upload. Agar client dapat mengakses menu-menu yang disediakan oleh server, client harus terdaftar dalam server, server memegang data-data client yang sudah dibuat manual oleh kami. Interface dan segala instruksi yang digunakan untuk mengeksekusi program menggunakan Command Line Interface. Adapun peran kami dalam membangun proyek tugas besar ini adalah sebagai berikut.

Nama Anggota	Peran dan Tanggung Jawab
Sya Raihan Heggi	Programmer dan Dokumentasi Proyek
Muhammad Raihan Muhith	Tester dan Dokumentasi Proyek
Mohammad Dwiantara Mahardhika	Presentasi dan Tester
Fariz Muhammad Rizky	Presentasi dan Dokumentasi Proyek

Table 1 Peran dan Tanggung Jawab Anggota

BAB II : ANALISIS

Alasan Pemilihan RPC

Dari permasalahan yang dihadapi yaitu membuat sebuah aplikasi FTP ada beberapa pilihan yang dapat digunakan dalam melakukan pengiriman, namun berdasarkan pertimbangan kekurangan dan kelebihan, serta mempertimbangkan uniqueness (Karena banyak contoh yang menggunakan socket saja), akhirnya kami putuskan untuk menggunakan RPC.

Remote Procedure Call dipilih untuk membangun sistem karena mengakomodasi pengaksesan prosedur/fungsi pada mesin jarak jauh (berkomunikasi menggunakan alamat IP) namun seolah-olah prosedur/fungsi tersebut ada di mesinnya sendiri RPC.

Definisi RPC

RPC (Remote Procedure Call) merupakan sebuah terobosan pada bidang komputasi terdistribusi. Tujuannya adalah membuat pemrograman sistem terdistribusi terlihat serupa. Dalam RPC, prosedur pada mesin jarak jauh dapat disebut seolah-olah merupakan prosedur yang ada pada mesin tersebut. RPC pada python dapat menggunakan library xmlrpc dengan code `import xmlrpc`.

Karakteristik RPC

- Menggunakan socket untuk berkomunikasi dengan proses lainnya
- Menggunakan paradigma procedural programming

Kelebihan RPC

- **Relatif mudah digunakan:**

Pemanggilan remote procedure tidak jauh berbeda dibandingkan pemanggilan procedure. Sehingga program dapat berkonsentrasi pada software logic, tidak perlu memikirkan low level details seperti socket, marshalling & unmarshalling.

- **Robust (Sempurna):**

Sejak tahun 1980-an RPC telah banyak digunakan dalam pengembangan mission- critical application yang memerlukan scalability

Kekurangan RPC

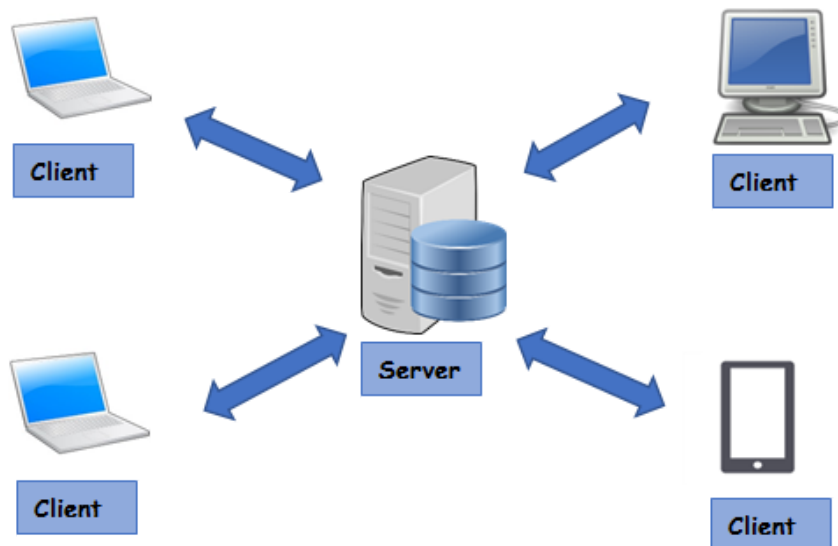
- **Tidak fleksibel terhadap perubahan:**
- **Static relationship between client & server at run-time.**
- **Berdasarkan prosedural/structured programming yang sudah ketinggalan jaman dibandingkan OOP.**

Model Sistem

Model yang digunakan dalam tugas besar ini sebenarnya ada dua yang cocok yaitu Peer to Peer maupun Client dan Server, namun karena kelompok kami lebih familiar terhadap model Client dan Server akhirnya model yang diputuskan dalam membangun adalah client server, karena disini akan dibutuhkan untuk Upload dan Download dan hal ini juga dimungkinkan dengan model Client-Server, selain itu cara kerja model ini secara sederhana client akan mengirimkan permintaan (Request) ke Server, dan Server akan membalasnya.

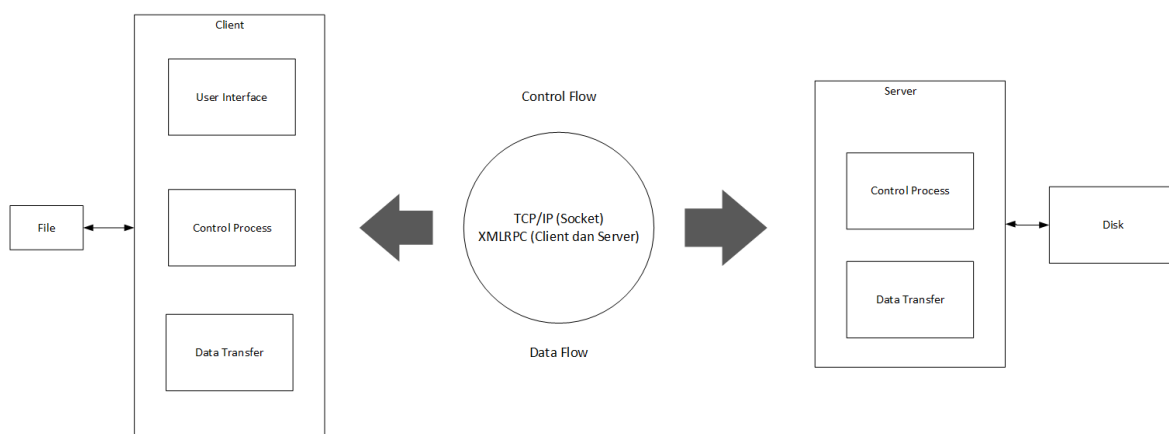
BAB III : PERANCANGAN

Pada pembuatan aplikasi FTP ini menggunakan arsitektur Client dan Server, dimana Server yang berfungsi untuk menangani dan memberikan service terhadap client, dan kemudian Client yang nantinya akan melakukan akses untuk melakukan unggah atau unduh ke dalam Server atau kurang lebihnya seperti gambar berikut ini.



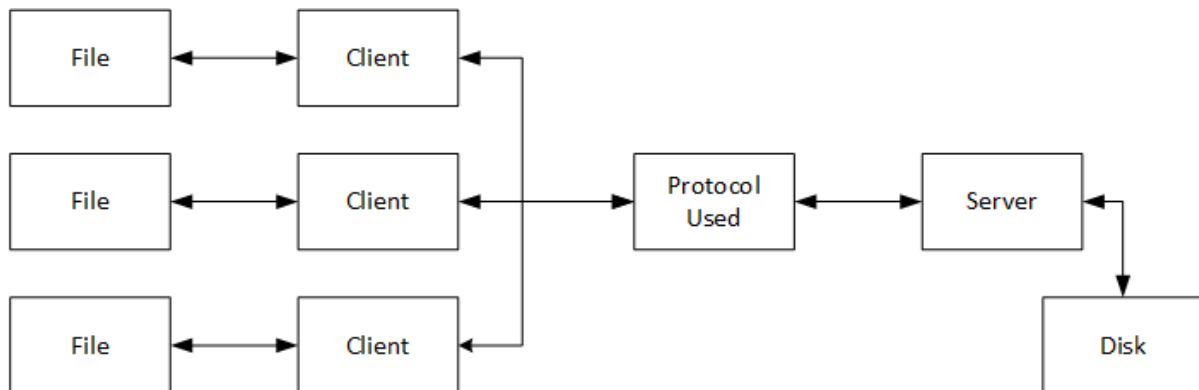
Sehingga setiap client akan melakukan proses melalui perantara Server, dan untuk penggunaan jenis komunikasi yang digunakan akan menggunakan Interprocess Communication (IPC) sehingga setiap client mempunyai identifikasi yang berbeda dan IPC yang digunakan adalah RPC, aplikasi kurang lebih dirancang seperti berikut ini.

SKEMA KERJA FTP

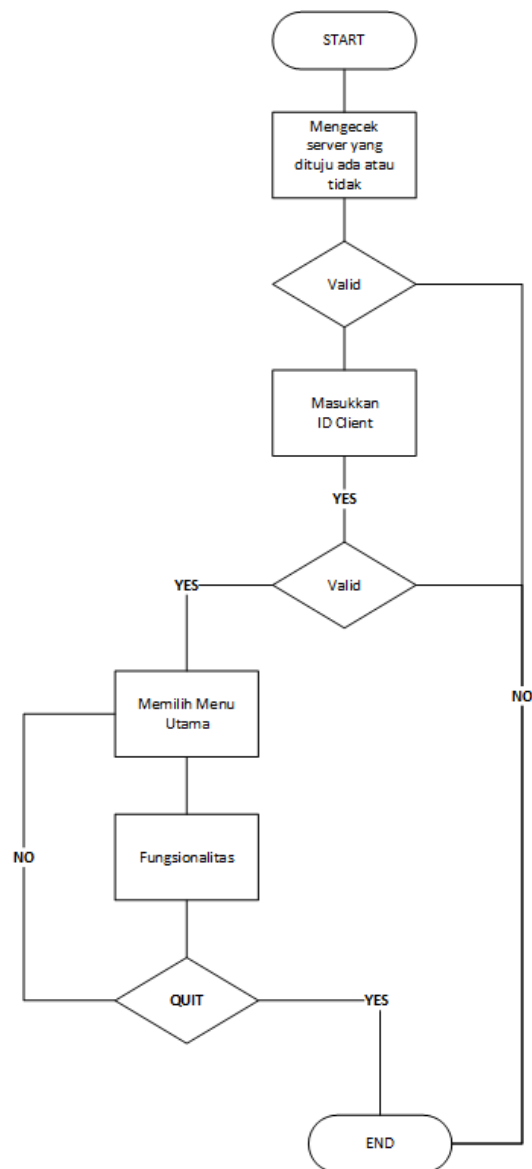


Karena pada aplikasi ini menggunakan RPC maka prosedur-prosedur yang digunakan dalam FTP akan disimpan didalam server yang didalamnya kurang lebih akan berisi prosedur unggah, unduh, menampilkan data list, dan menghitung aktivitas unggah-unduh yang dilakukan oleh client, selanjutnya prosedur itu akan di panggil melalui komunikasi RPC oleh

client sehingga dapat dijalankan oleh client berikut ini gambaran bila menggunakan banyak client didalam sistem.



Mungkin untuk lebih jelasnya akan dijelaskan dari metode aplikasi berjalan secara rinci pada flowchart berikut ini.



untuk mempermudah proses perhitungan client maka client harus mempunyai id unique oleh karena itu disiapkan id yang harus dimasukkan saat menjalankan aplikasi, kemudian bila valid akan menuju ke tahapan utama dan melakukan fungsionalitas, program tidak akan berakhir hingga fungsionalitas QUIT dijalankan, untuk menu utama akan memiliki fitur seperti pada baris kode berikut ini.

```
1 # fungsi yang digunakan untuk menampilkan menu dari aplikasi
2 def menu(clientName):
3     # clear command prompt
4     clear()
5     # print kata sambutan
6     print("\nSelamat Datang FTP client.\n")
7     # lakukan looping terus hingga kondisi menghentikan aplikasi dan menu
8     while True:
9         # bersihkan command prompt
10        clear()
11        # print tampilan mempercantik menu
12        print("+++++++ MENU ++++++")
13        # penjelasan perintah yang dapat digunakan
14        print("\nGunakan Perintah Berikut Ini:")
15        # untuk melakukan upload bisa menggunakan UPLD dan file_pathnya
16        print("UPLD file_path : Upload file")
17        # untuk melihat list dapat menggunakan LIST
18        print("LIST          : List files")
19        # untuk melihat keaktifan client
20        print("CLN          : Client Activity Record")
21        # untuk melakukan download bisa menggunakan DWLD dan file_pathnya
22        print("DWLD file_path : Download file")
23        # untuk menutup menggunakan QUIT
24        print("QUIT         : Exit from Application")
25        # input menu
26        prompt = input("\nSilahkan Masukkan Command : ")
27        # kita ambil 4 kalimat awal yang dimasukkan dan kita uppcase untuk perintah sesuai keperluan menu
28        if prompt[:4].upper() == "UPLD":
29            # Upload file dengan file_path yang berada pada kalimat 5 keatas
30            upld(prompt[5:], clientName)
31        elif prompt[:4].upper() == "LIST":
32            # buka fungsi listFile
33            listFile()
34        elif prompt[:4].upper() == "DWLD":
35            # buka fungsi download
36            dwnl(prompt[5:], clientName)
37        elif prompt[:4].upper() == "CLN":
38            cln()
39        elif prompt[:4].upper() == "QUIT":
40            # Tutup Aplikasi
41            clear()
42            break
43        else:
44            # bila tidak ditemukan command print pemberitahuan
45            clear()
46            print("Command not recognised; please try again")
```

Pada menu utama ini akan terdapat lima menu yang dapat digunakan yaitu UPLD, LIST, CLN, DWLD, dan QUIT yang akan lebih dijelaskan lebih lanjut berdasarkan point-point berikut ini.

1. UPLD File_Path

Menu yang disediakan ini adalah sebuah menu yang berfungsi untuk melakukan Upload file ke server sehingga bila diperhatikan pada menu ini Client akan mengirim dan Server akan menerima, berikut ini baris kode yang digunakan.

```
1 # Fungsi Untuk Mengirimkan File Ke Server
2 def upld(file_name, clientName):
3     # kita bersihkan command prompt agar rapi
4     clear()
5     # print untuk menandakan sedang berada di fungsi ana
6     print("Uploading {}".format(file_name))
7     try:
8         # buka file yang ingin dikirim dan baca perbaris
9         with open(file_name, "rb") as handle:
10             # data yang dikirim merupakan baris data yang dibaca
11             data = xmlrpclib.Binary(handle.read())
12             # file dikirim menggunakan fungsi upload yang ada di server
13             server.file_upload(data, file_name, clientName)
14     except Exception as e:
15         # jika terjadi eksepsi print eksepsinya
16         print(e)
```

Seperti biasa yang akan dilakukan adalah client akan membuka file_path yang ingin dikirimkan dan kemudian akan dibaca baris perbaris, yang nantinya akan diterima oleh server dan di simpan, jangan lupa dibuat try dan except untuk mencegah terjadi eksepsi pada sistem, karena menggunakan RPC maka kita dapat memanggil fungsi yang kita butuhkan, dan di sisi server akan seperti berikut ini.

```
1 # Fungsi yang digunakan untuk menerima file yang dikirimkan atau bila dilihat dari client melakukan proses upload
2 def receive_file(filedata, filename, clientName):
3     # kita buat try dan except untuk mencegah jika terjadi eksepsi
4     try:
5         # pertama kita buka fileUpload di server yang menerima pengiriman dari client
6         with open("upload_{}_{}".format(clientName, filename), "wb") as handle:
7             # filedata diterima dengan nama variabel json
8             json = filedata.data
9             # kemudian fileUpload.txt diupdate line data yang digunakan
10            handle.write(json)
11            # increment data di database
12            counter_data(clientName)
13            # kita return True untuk mengakhiri proses
14            return True
15    except Exception as e:
16        # print eksepsi yang terjadi untuk mengetahui kesalahan yang terjadi
17        print(e)
```

Server akan menerima data yang dikirim server dan melakukan counter telah melakukan unggah, kemudian data akan disimpan dengan format **upload_namaClient_namaFile**.

2. LIST

Untuk fungsionalitas ini untuk melihat file apa saja yang ada di server sehingga pengguna dapat mengunduh file yang diinginkan yang ada di server, sebenarnya yang perlu dilakukan adalah server akan membuat data yang berisi list data yang ada di direktorinya dengan menggunakan `os.listdir()`, kemudian data tersebut yang akan dikirim dan di tampilkan oleh client implementasi kodenya akan seperti berikut ini.

```
1 # Fungsi untuk menampilkan list data yang berada di server
2 def listFile():
3     # kita buat try dan except untuk mencegah jika terjadi eksepsi
4     try:
5         # pertama kita get data file yang berada di direktori menggunakan os.listdir()
6         arr = os.listdir()
7         # kemudian kita buat file yang menuliskan data byte yang dibaca sebelumnya agar bisa dikirim
8         with open("listDirektori.txt", "w+") as f:
9             # setiap item yang berada di arr nanti akan di tuliskan ke file direktori tersebut
10            for item in arr:
11                # lakukan write setiap item
12                f.write("%s\n" % item)
13            # kita tutup bila sudah di tulis
14            f.close()
15        # selanjutnya kita buka file List Direktori tersebut untuk mengirimkan data Direktori di server
16        with open("listDirektori.txt", "rb") as handle:
17            # Lakukan Pengiriman data tersebut perbaris data yang ada di file listdirektori
18            return xmlrpc.client.Binary(handle.read())
19        # kita tutup proses pengiriman dan pembacaan
20        handle.close()
21    except Exception as e:
22        # print eksepsi yang terjadi untuk mengetahui kesalahan yang terjadi
23        print(e)
```

```
1 # getting server direktori file
2 def listFile():
3     # print untuk menandai menu yang digunakan
4     print("List File")
5     try:
6         # lakukan looping agar data bisa ditampilkan
7         while True:
8             # bersihkan command prompt
9             clear()
10            # panggil data yang diterima dari server yang berupa list nama barang di server
11            data = server.list_file().data
12            # print kalimat untuk mempercantik aplikasi
13            print("Data Di Server \n")
14            # print data byte yang diterima dengan kita decode dulu ke format yang diketahui
15            print(data.decode("utf-8"))
16            # input untuk menu
17            prompt = input("Ingin Menutup Menu (Y/N) : ")
18            # jika Y maka kembali ke menu utama
19            if prompt == "Y" or prompt == "y":
20                break
21    except Exception as e:
22        # jika terjadi eksepsi maka print eksepsinya
23        print(e)
```

3. CLN

Fungsi ini berguna untuk menampilkan data dari keaktifan client dalam melakukan unggah dan unduh, proses ini berkaitan dengan proses pencatatan namun pada fungsi ini hanya menampilkan datanya saja, karena pencatatan ditangani oleh fungsi lainnya yang intinya fungsi pencatatan akan melakukan pengecekan dan menambah data kedalam database, kemudian data database inilah yang akan dikirim ke client.

```
1 # Print Data Keaktifan User Dengan mengembalikan data Client yang disimpan
2 def most_active_client():
3     try:
4         # panggil fungsi untuk mendapatkan dictionary data client
5         accnts = get_accounts_data()
6         # jika terjadi eksepsi
7     except Exception as e:
8         # print kesalahannya dimana
9         print(e)
10    # mengembalikan dictionary client
11    return accnts
```

```
1 # getting most active client
2 def cln():
3     try:
4         # looping menu
5         while True:
6             clear()
7             print("List Client")
8             # get data dari server mengenai client
9             data = server.client_active()
10            # sort max value dari dictionary dan mengembalikan key
11            maxValue = max(data, key=data.get)
12            # hasil data client dan nama client teraktif
13            print("Akun yang terdaftar : {}".format(data))
14            print("Akun yang teraktif : {}".format(maxValue))
15            # input untuk menu
16            prompt = input("Ingin Menutup Menu (Y/N) : ")
17            # jika Y maka kembali ke menu utama
18            if prompt == "Y" or prompt == "y":
19                break
20        # jika ada eksepsi
21    except Exception as e:
22        # print kesalahan
23        print(e)
```

Dan Fungsi Pencatatan akan seperti pada berikut ini, intinya akan melakukan validasi dan pengambilan terlebih dahulu dan kemudian mengaksesnya.

```
1 # melakukan counter_data untuk setiap upload dan download client
2 def counter_data(account):
3     # memanggil change value
4     return change_value(account)
5
6
7 # merubah value counter
8 def change_value(account):
9     # dapatkan dictionary keseluruhan data
10    akun = get_accounts_data()
11    try:
12        # increment nilai counter yang dikandung
13        akun[account] += 1
14        # kemudian tuliskan kembali kedalam database
15        write_to_database(akun)
16        # return True bila sudah tidak dibutuhkan
17        return True
18    # bila ada eksepsi key yang error
19    except (KeyError):
20        # tidak lakukan apapun tapi kembalikan false
21        return False
22
23
24 # mengambil keseluruhan data untuk diupdate
25 def get_accounts_data():
26     # siapkan dictionary untuk menampung data
27     accnts = {}
28     try:
29         # buka database dari sistem
30         fileData = open(ACCOUNT_FILE, "r")
31         # looping data yang ada di database
32         for line in fileData:
33             # ambil value dari database dan dibuat menjadi client5, ",", 0
34             value = line.rstrip().partition(",")
35             # buat menjadi dictionary {client5: value[2]}
36             accnts[value[0]] = int(value[2])
37         # tutup pembacaan
38         fileData.close()
39     # bila terjadi error input output
40     except IOError:
41         # Print notifikasi gagal
42         print("Gagal membuka {}".format(ACCOUNT_FILE))
43     # kembalikan dictionary akun untuk digunakan pada proses yang membutuhkan
44     return accnts
45
46
47 # update data di file penyimpanan akun
48 def write_to_database(akun):
49     try:
50         # buka database
51         f = open(ACCOUNT_FILE, "w")
52         # looping data pada dictionary key => keyvaluedict, val => valuedict {client5 (key) : 0 (value)}
53         for key, val in akun.items():
54             # write data kedalam database dengan format key,value atau client5,value
55             f.write("{}{}\n".format(key, val))
56         # tutup pembacaan
57         f.close()
58     # bila terjadi kesalahan input output
59     except IOError:
60         # print notifikasi gagal
61         print("Gagal menambahkan counter {}".format(ACCOUNT_FILE))
62     # mengembalikan true karena fungsi dari def hanya menuliskan dan tidak mendapatkan apapun
63     return True
```

4. DWLD

Perintah ini merupakan perintah untuk melakukan download file dari server, oleh karena itu dari sisi client akan menerima dan server mengirimkan dan hasilnya akan disimpan dalam format `download_namaFile`, untuk implementasi kode akan seperti berikut ini.

```
1 # Fungsi untuk Mengunduh File Dari Server ke Client dan kemudian akan disimpan dengan nama hasilDownload.txt
2 def dwnl(file_name, clientName):
3     # membersihkan command prompt
4     clear()
5     print(clientName)
6     # print untuk menandai proses download
7     try:
8         # buka file yang baru hasilDownload untuk menerima file yang dikirim server
9         print("Downloading {}".format(file_name))
10        with open("download_{}".format(file_name), "wb") as handle:
11            # tuliskan perbaris apa yang data yang diterima dari server
12            handle.write(server.file_download(file_name, clientName).data)
13            # penulisan di tutup
14            handle.close()
15    except Exception as e:
16        # jika terjadi eksepsi print eksepsinya
17        print(e)
18
```

```
1 # Fungsi yang digunakan untuk mengirimkan file dari server atau bila dilihat dari client melakukan proses download
2 def sendFile(fileDownload, clientName):
3     # kita buat try dan except untuk mencegah jika terjadi eksepsi
4     try:
5         # kita baca file yang ingin kita download yang berada di server
6         with open(fileDownload, "rb") as handle:
7             # increment data didatabase bila file ada
8             counter_data(clientName)
9             # kita akan mengirimkan setiap apa yang dibaca pada file tersebut
10            return xmlrpc.client.Binary(handle.read())
11            # kemudian pembacaan kita tutup
12            handle.close()
13    except Exception as e:
14        # print eksepsi yang terjadi untuk mengetahui kesalahan yang terjadi
15        print(e)
16
```

5. QUIT

Fungsionalitas ini digunakan untuk mengakhiri sesi client penerapannya hanya melakukan **break** pada menu utama sehingga aplikasi terhenti.

6. LOGIN

Untuk fungsionalitas ini perlu dilakukan pengecekan apakah data yang digunakan ada atau tidak di database oleh karena itu pada server akan mengambil dan mengirimkan nama client bila hanya string kosong maka dianggap tidak valid, implementasinya akan seperti berikut ini.

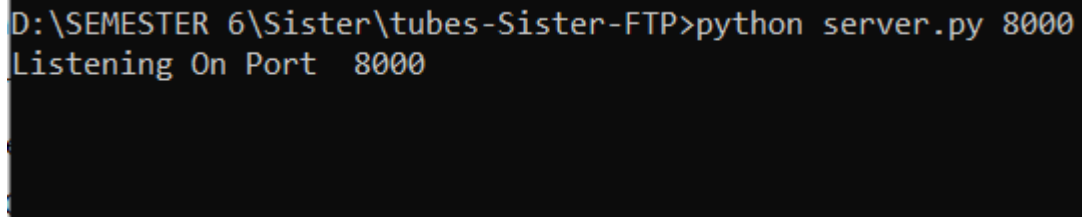
```
1 # fungsionalitas utama
2 def main():
3     # deklarasi global variabel yang dapat digunakan
4     global server
5     global clear
6     global f
7     global counterDownload
8     global clientName
9     # buat try and except mencegah eksepsi
10    try:
11        # buat fungsi untuk membersihkan command prompt windows dengan cls
12        clear = lambda: os.system("cls")
13        # Connect to Server
14        server = xmlrpcclib.ServerProxy("http://127.0.0.1:8000/")
15        # lakukan login untuk memastikan data client ada dan dapat digunakan
16        clientName = server.login_client(input("Silahkan Masukan ID anda : "))
17        # buat kondisi bila data ada
18        if clientName != "":
19            # Open Main Menu
20            menu(clientName)
21        # jika data tidak ada
22        else:
23            # naikan jadi Exception untuk menghentikan program
24            raise Exception
25        # jika socket connection error
26    except socket.error as e:
27        # print pemberitahuan
28        print("Socket Error")
29        print(e)
30        # jika terjadi eksepsi lainnya
31    except Exception as e:
32        print("Terjadi Kesalahan")
```

```
1 # get data akun untuk pertama kali saat login
2 def get_accounts_name(akun_name):
3     # siapkan string kosong untuk menampung id client yang nantinya akan dipastikan ada atau tidak
4     accnts = ""
5     try:
6         # buka database account di dataClient.txt / variable constant ACCOUNT_FILE
7         fileData = open(ACCOUNT_FILE, "r")
8         # data kita looping
9         for line in fileData:
10            # ambil valuenya dan kita pisahnya client5,0 => client5 , ",", 0
11            value = line.rstrip().partition(",")
12            # kondisi jika value ada
13            if value[0] == akun_name:
14                # setting value id client untuk dikembalikan
15                accnts = value[0]
16        # tutup bila sudah selesai
17        fileData.close()
18        # eksepsi bila ada input/output error
19    except IOError:
20        # print notifikasi
21        print("Gagal membuka {}".format(ACCOUNT_FILE))
22        # kembalikan account
23    return accnts
24
```

BAB III : IMPLEMENTASI

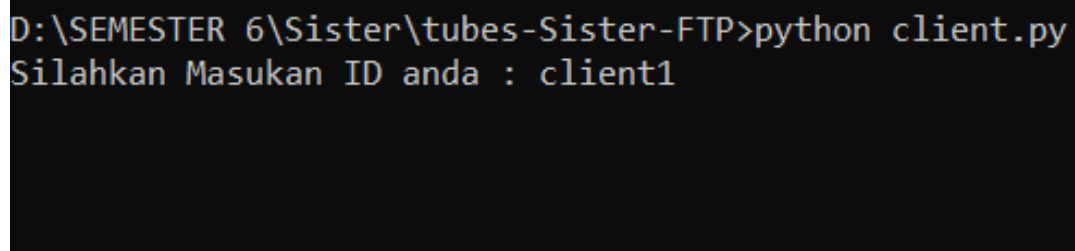
Bab ini merupakan bagian pada laporan yang memberikan gambaran pada pembaca bagaimana hasil projek program FTP sederhana yang kami bangun jika dijalankan.

1. Menjalankan server menggunakan perintah `>python server.py [port_number]`



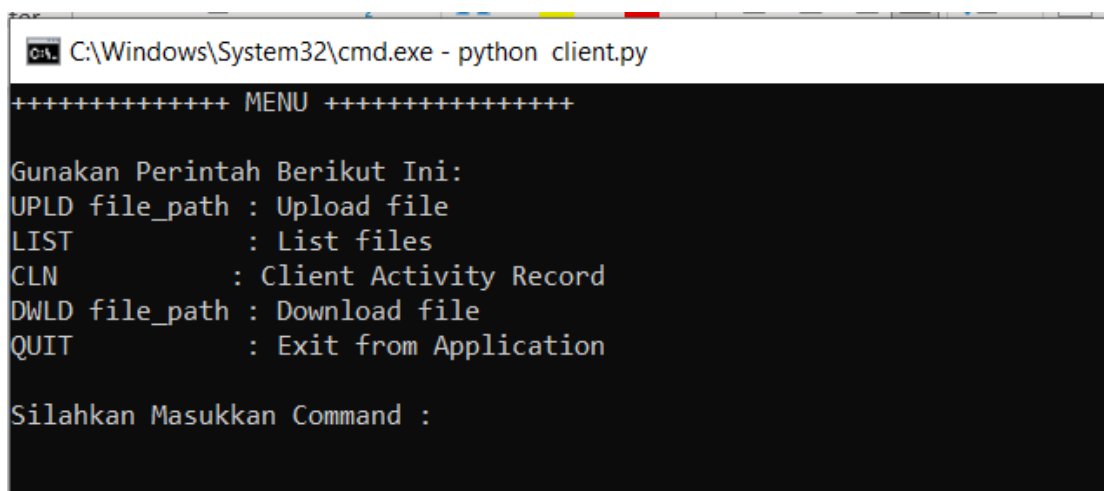
```
D:\SEMESTER 6\Sister\tubes-Sister-FTP>python server.py 8000
Listening On Port 8000
```

2. Menjalankan client menggunakan perintah `>python client.py` serta memasukkan id client dengan id yang sudah terdaftar dari server.



```
D:\SEMESTER 6\Sister\tubes-Sister-FTP>python client.py
Silahkan Masukan ID anda : client1
```

3. Tampilan Menu sederhana pada client agar memudahkan dalam proses yang akan dilakukan.



```
C:\Windows\System32\cmd.exe - python client.py

+++++ MENU +++++

Gunakan Perintah Berikut Ini:
UPLD file_path : Upload file
LIST           : List files
CLN            : Client Activity Record
DWLD file_path : Download file
QUIT          : Exit from Application

Silahkan Masukkan Command :
```

4. Melihat file-file yang ada di server menggunakan perintah LIST

```
C:\Windows\System32\cmd.exe - python client.py

Data Di Server

.git
client.py
dataClient.txt
kendaraan_train.csv
listDirektori.txt
README.md
server.py
test.py

Ingin Menutup Menu (Y/N) :
```

5. Mendownload file dari server menggunakan perintah DWLD [file_path]
6. Mengupload file dari client ke server menggunakan perintah UPLD [file_path]
7. Melihat client teraktif yang terdaftar pada server menggunakan perintah CLN

```
C:\Windows\System32\cmd.exe - python client.py

List Client
Akun yang terdaftar : {'client5': 3, 'client4': 1, 'client3': 2, 'client2': 0, 'client1': 5}
Akun yang teraktif : client1
Ingin Menutup Menu (Y/N) :
```

Adapun keterbatasan/permasalahan yang kami temukan dalam proses pembuatan program FTP ini antara lain :

- Client tidak dapat melakukan registrasi mandiri, yang mengakibatkan program ini menjadi statis.
- Untuk menjalankan aplikasi masih belum dinamis dalam menentukan port dan address sehingga masih perlu untuk mengubah baris kode.
- Sistem penamaan mungkin masih kurang baik yang merupakan file yang dikirim atau tidak
- Jika file terlalu kecil notifikasi tidak muncul karena proses pengiriman cukup cepat

Link Video Presentasi dan Demo :

- <http://bit.ly/tubesSisterFTP4203> (Google Drive View)
- <https://youtu.be/WaXuHFXhajk> (Youtube)

