

Analisis Config File

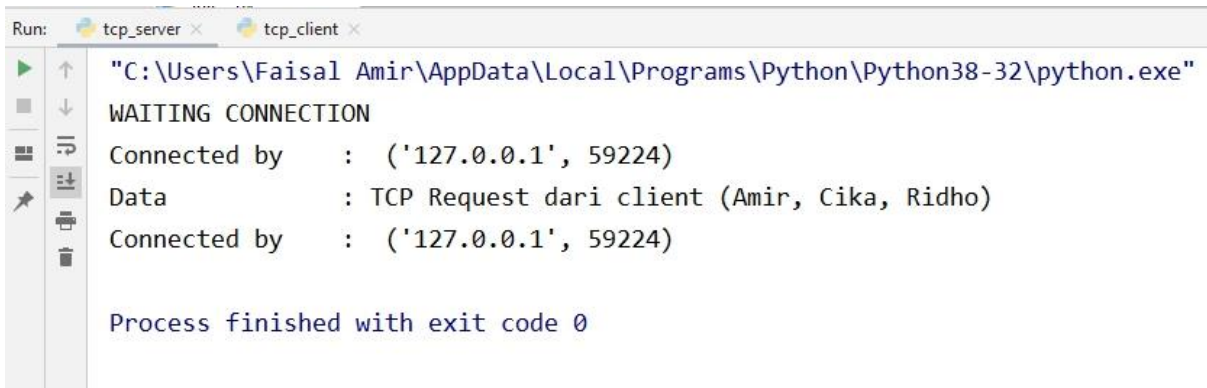
CODE	FUNGSI (PENJELASAN)
<pre> BASE_CONFIG_IP_ADDRESS = '127.0.0.1' BASE_CONFIG_PORT = 6080 BASE_CONFIG_BUFFER = 1110 BASE_MESSAGE_RESPONSE = 'RECEIVED' BASE_MESSAGE_CONNECTING = 'WAITING CONNECTION' BASE_MESSAGE_REQUEST = 'REQUEST' MESSAGE_REQUEST = 'Request dari client (Amir, Cika, Ridho)' </pre>	Deklarasi konfigurasi ip address, port, buffer, dan message

Analisis TCP Server

CODE	FUNGSI (PENJELASAN)
<pre> import socket from com.frogobox.base.config import * </pre>	Import function untuk memanggil fungsi socket
<pre> # definisikan alamat IP binding yang akan digunakan TCP_SERVER_IP = BASE_CONFIG_IP_ADDRESS # definisikan port number binding yang akan digunakan TCP_SERVER_PORT = BASE_CONFIG_PORT # definisikan ukuran buffer untuk mengirimkan pesan TCP_SERVER_BUFFER = BASE_CONFIG_BUFFER </pre>	Deklarasi IP Address, Port, Buffer dan Address.
<pre> print(BASE_MESSAGE_CONNECTING) </pre>	Menampilkan proses menunggu koneksi
<pre> # buat socket bertipe TCP with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as tcpServer: # lakukan bind tcpServer.bind((TCP_SERVER_IP, TCP_SERVER_PORT)) # server akan listen menunggu hingga ada koneksi dari client tcpServer.listen() # menerima koneksi connection, address = tcpServer.accept() with connection: # Lakukan Loop forever while True: </pre>	<p>Mengaktifkan socket yang telah di inialisasi menjadi tcp server. Kemudian mengaitkan socket pada IP dan port yang sudah di deklarasikan pada file config.py</p> <p>Kemudian code tcpServer.listen() menunggu sampai ada request dari client yang siap di jalankan.</p> <p>Jika client merequest dengan benar makan koneksi di setuju.</p> <p>Dengan koneksi yang di setuju tersebut maka akan melakukan perulangan secara terus menerus dengan menjalankan perintah menampilkan koneksi IP Adress dan pesan yang dikirim dari client.</p>

<pre> # menampilkan koneksi berupa IP dan port client yang terhubung menggunakan print print('Connected by \t: ', address) # menerima data berdasarkan ukuran buffer messageFromClient = connection.recv(TCP_SERVER_BUFFER) if not messageFromClient: break # menampilkan pesan yang diterima oleh server menggunakan print, print('Data \t\t\t:', str(messageFromClient, 'utf-8')) # mengirim kembali data yang diterima dari client kepada client connection.send(messageFromClient) </pre>	<p>Terakhir connection.send() berfungsi untuk mengirimkan kembali ke client</p>
<pre> # tutup koneksi tcpServer.close() </pre>	<p>Koneksi di tutup</p>

Screenshoot



```

Run: tcp_server x tcp_client x
"C:\Users\Faisal Amir\AppData\Local\Programs\Python\Python38-32\python.exe"
WAITING CONNECTION
Connected by      : ('127.0.0.1', 59224)
Data              : TCP Request dari client (Amir, Cika, Ridho)
Connected by      : ('127.0.0.1', 59224)

Process finished with exit code 0

```

Analisis TCP Client

CODE	FUNGSI (PENJELASAN)
<pre># import library socket karena akan menggunakan IPC socket import socket from com.frogobox.base.config import *</pre>	Import function untuk memanggil fungsi socket
<pre># definisikan tujuan IP server TCP_SERVER_IP = BASE_CONFIG_IP_ADDRESS # definisikan port dari server yang akan terhubung TCP_SERVICE_PORT = BASE_CONFIG_PORT # definisikan ukuran buffer untuk mengirimkan pesan TCP_SERVER_BUFFER = BASE_CONFIG_BUFFER # definisikan pesan yang akan disampaikan TCP_MESSAGE = 'TCP ' + MESSAGE_REQUEST</pre>	Deklarasi IP, Port, buffer serta message yang akan di gunakan untuk menghubungkan koneksi ke server
<pre># buat socket TCP tcpClient = socket.socket(socket.AF_INET, socket.SOCK_STREAM)</pre>	Declarasi tcp Client dengan menggunakan Socket
<pre># Lakukan koneksi ke server dengan parameter IP dan Port yang telah didefinisikan tcpClient.connect((TCP_SERVER_IP, TCP_SERVICE_PORT))</pre>	Membuat koneksi TCP client
<pre># kirim pesan ke server, pesan bebas, dan ditambahkan nama anggota kelompok tcpClient.send(TCP_MESSAGE.encode())</pre>	TCP client mengirimkan pesan dengan cara mengencode terlebih dahulu
<pre># terima pesan dari server messageFromServer = tcpClient.recv(TCP_SERVER_BUFFER) # tampilkan pesan/reply dari server print('Received', messageFromServer.decode())</pre>	Mendapatkan pesan balikan dari server dan kemudian menampilkannya kembali
<pre># tutup koneksi tcpClient.close()</pre>	Menutup koneksi tcp client

Screenshoot

```

Run: tcp_server x tcp_client x
"C:\Users\Faisal Amir\AppData\Local\Programs\Python\Python38-32\python.exe"
Received TCP Request dari client (Amir, Cika, Ridho)

Process finished with exit code 0

```

Analisis UDP Server

CODE	FUNGSI (PENJELASAN)
<pre> # import library socket karena akan menggunakan IPC socket import socket from com.frogobox.base.config import * </pre>	Import function untuk memanggil fungsi socket
<pre> # Berikan tampilan (print bahwa socket berhasil dibuat) print("Socket Has Been Created") print(BASE_MESSAGE_CONNECTING) print() </pre>	Tampilan untuk bukti bahwa server telah di jalankan
<pre> # definisikan alamat IP bind dari server UDP_SERVER_IP = BASE_CONFIG_IP_ADDRESS # definisikan port number untuk bind dari server UDP_SERVER_PORT = BASE_CONFIG_PORT UDP_SERVER_BUFFER = BASE_CONFIG_BUFFER </pre>	Deklarasi IP, Port, dan Buffer untuk server
<pre> # buat socket bertipe UDP with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as udpServer: # Lakukan bind udpServer.bind((UDP_SERVER_IP, UDP_SERVER_PORT)) # Berikan tampilan (print) bahwa socket bind ke alamat IP dengan port yang telah didapatkan # Loop forever while True: # terima pesan dari client messageFromClient, address = udpServer.recvfrom(UDP_SERVER_BUFFER) if not messageFromClient: break </pre>	<p>Membuat socket bertipe UDP, kemudian socket tersebut di kaitkan pada ip dan port yang sudah di deklarasikan.</p> <p>Setelah itu dilakukan perulangan terus menerus untuk mendapatkan mengecek apakah ada pesan dari klien atau tidak, jika ada maka di tampilkan. Jika tidak maka server tetap menyala terus menerus</p>

<pre> # menampilkan hasil pesan dari client print('Data \t\t\t:', messageFromClient.decode()) # print (addr) print('Received from \t: ', address) print() </pre>	
<pre> udpServer.close() </pre>	Koneksi server udp di tutup

Screenshoot

```

Run: udp_server x udp_client x
"C:\Users\Faisal Amir\AppData\Local\Programs\Python\Python38-32\python.exe" E:/PI
WAITING CONNECTION

Data      : UDP Request dari client (Amir, Cika, Ridho) pesan ke - 1
Received from : ('127.0.0.1', 49770)

Data      : UDP Request dari client (Amir, Cika, Ridho) pesan ke - 2
Received from : ('127.0.0.1', 49770)

Data      : UDP Request dari client (Amir, Cika, Ridho) pesan ke - 3
Received from : ('127.0.0.1', 49770)

Data      : UDP Request dari client (Amir, Cika, Ridho) pesan ke - 4
Received from : ('127.0.0.1', 49770)

Data      : UDP Request dari client (Amir, Cika, Ridho) pesan ke - 5
Received from : ('127.0.0.1', 49770)

Data      : UDP Request dari client (Amir, Cika, Ridho) pesan ke - 6
Received from : ('127.0.0.1', 49770)

Data      : UDP Request dari client (Amir, Cika, Ridho) pesan ke - 7
Received from : ('127.0.0.1', 49770)

Data      : UDP Request dari client (Amir, Cika, Ridho) pesan ke - 8
Received from : ('127.0.0.1', 49770)

Data      : UDP Request dari client (Amir, Cika, Ridho) pesan ke - 9
Received from : ('127.0.0.1', 49770)

Data      : UDP Request dari client (Amir, Cika, Ridho) pesan ke - 10
Received from : ('127.0.0.1', 49770)

```

Analisis UDP Client

CODE	FUNGSI (PENJELASAN)
------	---------------------

<pre># import library socket karena akan menggunakan IPC socket import socket from com.frogobox.base.config import *</pre>	Import function untuk memanggil fungsi socket
<pre># definisikan target IP server yang akan dituju UDP_SERVER_IP = BASE_CONFIG_IP_ADDRESS # definisikan target port number server yang akan dituju UDP_SERVER_PORT = BASE_CONFIG_PORT UDP_MESSAGE = 'UDP ' + MESSAGE_REQUEST</pre>	Deklarasi IP, Port dan Message dari klien
<pre>print('Sending Message ' + UDP_MESSAGE)</pre>	Menampilkan apa yang sedang dilakukan oleh client yaitu mengirim pesan
<pre># buat socket bertipe UDP with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as udpClient: udpClient.connect((UDP_SERVER_IP, UDP_SERVER_PORT)) for x in range(10): udpClient.sendto(str(UDP_MESSAGE + ' pesan ke - ' + str(x + 1)).encode(), (UDP_SERVER_IP, UDP_SERVER_PORT))</pre>	Mendeklarasikan udp client dengan menggunakan socket yang dikaitkan pada ip dan port udp server yang sudah di buat. Kemudian mengirimkan pesan sebanyak 10 kali dengan perulangan. Sesuai ip dan port server yang tepat
<pre>udpClient.close()</pre>	Menutup koneksi udp client

Screenshoot

```
Run: udp_server x udp_client x
"C:\Users\Faisal Amir\AppData\Local\Programs\Python\Python38-32\python.exe"
Sending Message UDP Request dari client (Amir, Cika, Ridho)
Process finished with exit code 0
```

Telkom University / IFX - 43 – GAB

Friskadini Ismayanti (1301198496)

Muhammad Faisal Amir (1301198497)

Ridho Maulana Cahyudi (1301198515)