

# **LAPORAN PRAKTIKUM PEMROGRAMAN JARINGAN**

**“Client – Server (Single Thread)”**



Dibuat oleh :

Reyza Syaifullah Sully - 1203220045

**PROGRAM STUDI S1 INFORMATIKA**

**FAKULTAS INFORMATIKA**

**UNIVERSITAS TELKOM UNIVERSITY**

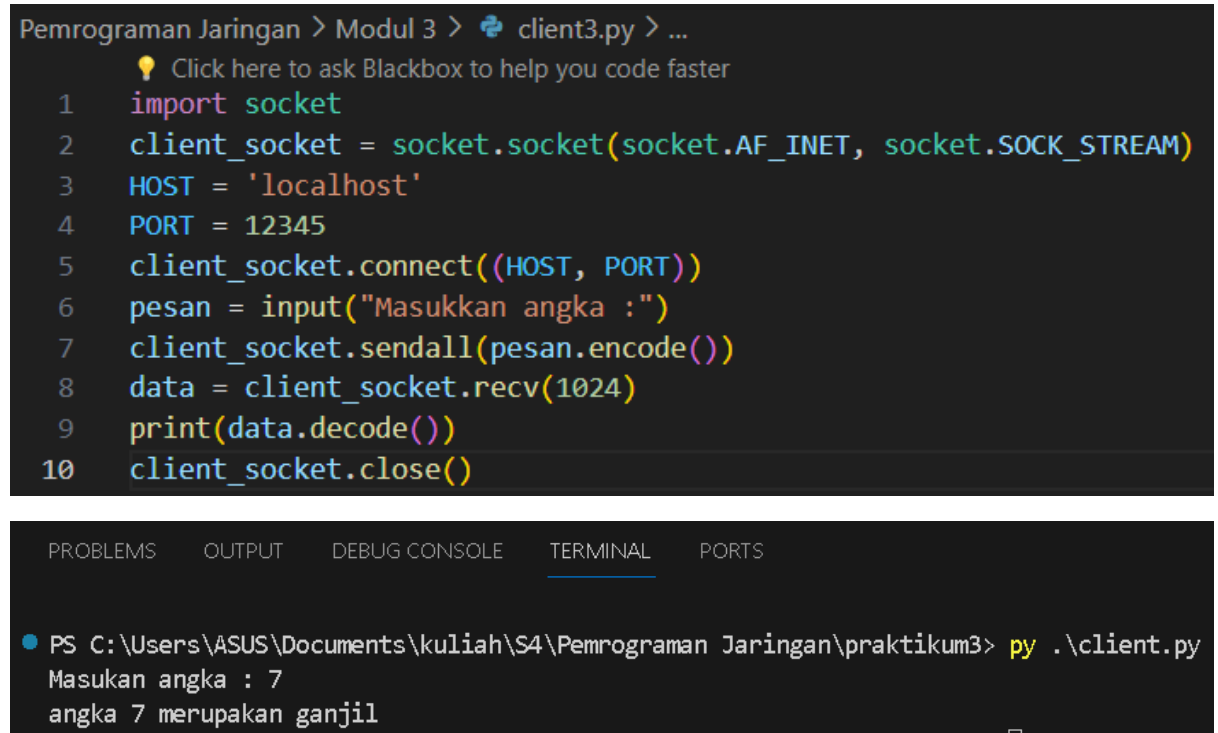
**SURABAYA**

**2024**

## TUGAS DAN LATIHAN PRAKTIKUM

1. Membuat laporan percobaan praktikum dan beri Analisa Hasil Percobaan tadi yang sudah dibuat Pembuatan Aplikasi Client-Server Sederhana (Single Thread).

Sisi Client



```
Pemrograman Jaringan > Modul 3 > client3.py > ...
  Click here to ask Blackbox to help you code faster
1  import socket
2  client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3  HOST = 'localhost'
4  PORT = 12345
5  client_socket.connect((HOST, PORT))
6  pesan = input("Masukkan angka :")
7  client_socket.sendall(pesan.encode())
8  data = client_socket.recv(1024)
9  print(data.decode())
10 client_socket.close()
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
● PS C:\Users\ASUS\Documents\kuliah\S4\Pemrograman Jaringan\praktikum3> py .\client.py
Masukan angka : 7
angka 7 merupakan ganjil
```

Menggunakan modul socket dengan fungsi 'import socket' untuk mengatur komunikasi jaringan. Pada sisi klien, sebuah objek socket dibuat dengan 'client\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)'. Parameter 'socket.AF\_INET' menandakan penggunaan alamat IPv4, sedangkan 'socket.SOCK\_STREAM' menunjukkan penggunaan protokol TCP untuk transportasi data.

Variabel HOST = 'localhost' dan PORT = 12345 menetapkan alamat host yang akan terhubung dengan server, dengan 'localhost' menandakan bahwa server berjalan pada mesin yang sama dengan klien. Koneksi antara socket klien dan alamat server ditetapkan dengan 'client\_socket.connect((HOST, PORT))'.

Input dari pengguna berupa angka yang akan dikirim ke server diambil melalui terminal dengan perintah 'Pesan = input("Masukkan angka :")'. Kemudian, pesan tersebut dikirim ke server menggunakan 'Client\_socket.sendall(pesan.encode())', dimana 'encode' digunakan untuk mengubah string (pesan) menjadi bytes yang dapat dikirim melalui jaringan.

Respon dari server diterima dengan 'Data = client\_socket.recv(1024)', dimana 1024 adalah ukuran maksimum byte yang diterima dalam satu waktu. Pesan yang diterima dari server kemudian ditampilkan kepada pengguna dengan 'print(data.decode())', dengan 'decode' digunakan untuk mengubah byte kembali menjadi string.

Koneksi socket ditutup dengan 'Client\_socket.close()' setelah selesai.

#### Sisi Server

```
1 import socket
2 server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
3 HOST = 'localhost'
4 PORT = 12345
5 server_socket.bind((HOST, PORT))
6 server_socket.listen(1)
7 print("Waiting...")
8 client_socket, client_address = server_socket.accept()
9 data = client_socket.recv(1024)
10 angka = int(data.decode())
11 print("Request dari client :", angka, "IP client :", client_address)
12 if angka % 2 == 0:
13     response = "angka " + str(angka) + " merupakan genap"
14 else:
15     response = "angka " + str(angka) + " merupakan ganjil"
16 client_socket.sendall(response.encode())
17 client_socket.close()
18 server_socket.close()
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\ASUS\Documents\kuliah\S4\Pemrograman Jaringan\praktikum3> py .\server.py
PS C:\Users\ASUS\Documents\kuliah\S4\Pemrograman Jaringan\praktikum3> py .\server.py
Waiting...
Request dari client : 7 IP client : ('127.0.0.1', 64903)
```

Menggunakan modul socket dengan fungsi 'import socket' untuk menciptakan objek socket dan melaksanakan operasi jaringan. Objek socket server dibuat dengan 'server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)' dengan penggunaan alamat IPv4 dan jenis socket stream.

Variabel HOST = 'localhost' dan PORT = 12345 digunakan untuk menentukan host dan port server. Socket server diikat ke alamat dan port tertentu untuk menerima koneksi dari klien dengan 'server\_socket.bind((HOST, PORT))'. Server akan mendengarkan koneksi yang masuk dengan 'server\_socket.listen(1)', dimana parameter 1 menunjukkan jumlah koneksi yang dapat diterima secara simultan.

"Pesan "waiting..." dicetak untuk memberi tahu bahwa server sedang menunggu koneksi dari klien. Koneksi dari klien diterima dengan 'client\_socket, client\_address =

`server_socket.accept()`), mengembalikan objek socket baru untuk berkomunikasi dengan klien serta alamat IP klien.

Data yang dikirim oleh klien diterima dengan '`Data = client_socket.recv(1024)`', dimana maksimal 1024 byte dapat diterima sekaligus. Data tersebut kemudian dikonversi menjadi bilangan bulat dengan '`Angka = int(data.decode())`'.

"Pesan "request dari client :", angka, "IP client :", `client_address`" ditampilkan untuk menampilkan permintaan dari klien beserta alamat IP klien.

Hasil dari pengecekan apakah angka input adalah genap atau ganjil ditentukan dengan menggunakan fungsi '`if angka%2 == 0`' dan '`else`', dengan '`response = "angka"+str(angka)+"merupakan genap /(ganjil)"`' untuk menyiapkan pesan respons yang akan dikirim kembali kepada klien berdasarkan hasil pengecekan tersebut.

Respons tersebut dikirim kembali kepada klien setelah dikodekan dalam format byte dengan '`Client_socket.sendall(response.encode())`'.

Koneksi dengan klien ditutup dengan '`client_socket.close()`', dan socket server ditutup dengan '`server_socket.close()`' setelah selesai berkomunikasi.

2. Membuat sebuah program server yang dapat menerima koneksi dari klien menggunakan protokol TCP. Server ini akan menerima pesan dari klien dan mengirimkan pesan balasan berisi jumlah karakter pada pesan tersebut. Gunakan port 12345 untuk server. Membuat analisa dari hasil program tersebut

Screenshot :

```
21 import socket
22
23 server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
24
25 HOST = 'localhost'
26 PORT = 12345
27
28 server_socket.bind((HOST, PORT))
29 server_socket.listen(1)
30
31 print("Waiting...")
32
33 client_socket, client_address = server_socket.accept()
34
35 pesan = client_socket.recv(1024).decode()
36 jumlah_karakter = len(pesan)
37 pesan_balasan = f"Jumlah karakter pesan: {jumlah_karakter}"
38
39 client_socket.sendall(pesan_balasan.encode())
40
41 client_socket.close()
42 server_socket.close()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
● PS C:\Users\ASUS\Documents\kuliah\S4\Pemrograman Jaringan\praktikum3> py .\server.py
○ PS C:\Users\ASUS\Documents\kuliah\S4\Pemrograman Jaringan\praktikum3> py .\server.py
Waiting...
```

Membuat sebuah server menggunakan modul socket pada bahasa pemrograman Python. Server akan mengikat socket ke alamat IP 'localhost' dengan port '12345' dan kemudian akan mulai mendengarkan koneksi. Saat ada koneksi masuk dari klien, server akan menerima koneksi tersebut dan menerima pesan yang dikirim oleh klien. Pesan yang diterima oleh server berupa data dalam format string yang akan diubah menjadi sebuah bilangan bulat dengan proses decode.

Setelah menerima pesan dari klien, server akan menghitung jumlah karakter dalam pesan tersebut. Selanjutnya, server akan membuat sebuah pesan balasan yang berisi jumlah karakter dari pesan yang diterima, dan mengirimkannya kembali kepada klien. Setelah mengirim pesan balasan, server akan menutup koneksi dengan klien dan kemudian menutup socket.

3. Langkah kedua adalah membuat program klien yang dapat terhubung ke server yang telah dibuat dalam soal nomor 1. Klien ini akan mengirimkan pesan ke server berupa inputan yang diberikan oleh pengguna, dan kemudian menampilkan pesan balasan yang berisi jumlah karakter yang diterima dari server. Selanjutnya, kita akan melakukan analisis terhadap hasil dari program tersebut.

Screenshot :

```
12
13  import socket
14
15  client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
16
17  HOST = 'localhost'
18  PORT = 12345
19
20  client_socket.connect((HOST, PORT))
21  pesan = input("Masukkan pesan: ")
22
23  client_socket.sendall(pesan.encode())
24  pesan_balasan = client_socket.recv(1024).decode()
25
26  print("Pesan balasan dari server:", pesan_balasan)
27
28  client_socket.close()
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
● PS C:\Users\ASUS\Documents\kuliah\S4\Pemrograman Jaringan\praktikum3> py .\client.py
Masukkan pesan: Reyza Syaifullah Sully
Pesan balasan dari server: Jumlah karakter pesan: 22
```

Membuat sebuah objek klien socket untuk berkomunikasi dengan server menggunakan modul 'socket'. Klien akan terhubung ke server menggunakan alamat 'localhost' dan port '12345'. Setelah itu, klien akan meminta input dari pengguna berupa pesan yang akan dikirimkan ke server menggunakan metode 'sendall()'.

Ketika klien menerima respons dari server menggunakan metode 'recv()', respons tersebut akan berisi pesan balasan yang mencakup jumlah karakter dari pesan yang dikirim oleh klien. Klien akan menampilkan pesan balasan dari server. Setelah selesai berkomunikasi, klien akan menutup koneksi dengan server.