

**LAPORAN PRAKTIKUM**  
**Chat Sederhana Berbasis Socket di Python**



Nama : L Hafidl Alkhair  
Nim : 2023903430060  
Kelas : TRKJ-2C  
Jurusan : Teknologi Informasi dan Komputer  
Progam Studi : Teknologi Rekayasa Komputer dan Jaringan  
Dosem Pengampu : Afla Nevrisa S.Kom, M.Kom



**PROGRAM STUDI TEKNOLOGI REKAYASA KOMPUTER JARINGAN**  
**JURUSAN TEKNOLOGI INFORMASI DAN KOMPUTER**  
**POLITEKNIK NEGERI LHOKSEUMAWE 2025**

## LEMBAR PENGESAHAN

No. Praktikum : 02 /TIK/TRKJ-2C/ Pemrograman Jaringan  
Judul : Laporan Praktikum  
Nama : L Hafidl Alkhair  
Nim : 202390343060  
Kelas : TRKJ-2C  
Jurusan : Teknologi Informasi Dann Komputer  
Program Studi : Teknologi Rekayasa Komputer Jaringan  
Tanggal Praktikum : 20 Mei 2025  
Tanggal Penyerahan : 1 Juni 2025

Buketrata, 1 Juni 2025

Dosen Pembimbing,

**Afla Nevrisa S.Kom, M.Kom**

NIP. 199211172022032007



## BAB I PENDAHULUAN

### A. Dasar teori

#### 1. Socket Programming

Socket adalah antarmuka yang memungkinkan komunikasi antara dua node dalam jaringan. Dalam pemrograman Python, socket digunakan untuk membuat aplikasi jaringan seperti server dan client. Socket menyediakan dua endpoint: socket server dan socket client, yang saling bertukar data.

#### 2. TCP/IP

Transmission Control Protocol (TCP) adalah protokol komunikasi yang andal dan banyak digunakan di internet. Protokol ini menjamin bahwa data yang dikirim akan sampai ke tujuan secara utuh dan berurutan. Program chat ini menggunakan socket TCP untuk memastikan pesan antar client terkirim dengan benar.

#### 3. Multithreading

Multithreading memungkinkan aplikasi untuk menjalankan beberapa bagian kode secara bersamaan. Dalam aplikasi server, threading digunakan untuk menangani banyak client secara paralel tanpa membuat server menunggu satu client selesai.

#### 4. Client-Server Architecture

Arsitektur ini memisahkan sistem menjadi dua komponen utama: client (pengguna) dan server (penyedia layanan). Server akan terus mendengarkan permintaan dari client dan menanggapi sesuai logika aplikasi.

### B. Tujuan Praktikum

- Menerapkan konsep dasar pemrograman socket di Python.
- Membuat program server dan client yang dapat berkomunikasi menggunakan socket.
- Mengimplementasikan multithreading pada sisi server untuk menangani banyak client secara bersamaan.
- Memahami prinsip komunikasi jaringan berbasis TCP/IP.

### C. Alat dan bahan

- Laptop/PC
- Python 3.x
- Text editor (seperti VS Code, Sublime Text, atau lainnya)
- Terminal atau Command Prompt
- Jaringan lokal (opsional)



## BAB II PRAKTIKUM

### 1. Membuat Program Server

Buat file server.py dengan kode berikut:

```
#!/usr/bin/env python3
```

```
import socket
import threading
```

```
# Konfigurasi Server
```

```
IP = '127.0.0.1'
```

```
PORT = 3606
```

```
MAX_CLIENTS = 5
```

```
# Daftar client yang terhubung
```

```
clients = []
```

```
# Fungsi broadcast pesan ke semua client kecuali pengirim
```

```
def broadcast(message, sender_socket):
```

```
    for client in clients:
```

```
        if client != sender_socket:
```

```
            try:
```

```
                client.send(message)
```

```
            except:
```

```
                client.close()
```

```
                if client in clients:
```

```
                    clients.remove(client)
```

```
# Fungsi handle untuk masing-masing client
```

```
def handle_client(client_socket, addr):
```

```
    print(f"[+] {addr} terhubung.")
```

```
    broadcast(f"[INFO] {addr} bergabung ke server.\n".encode('utf-8'), client_socket)
```

```
while True:
```

```
    try:
```

```
        message = client_socket.recv(1024)
```

```
        if not message:
```

```
            break
```

```
        print(f"[{addr}] {message.decode('utf-8')}")
```

```
        broadcast(f"[{addr}] {message.decode('utf-8')}".encode('utf-8'),
```

```
client_socket)
```

```
    except:
```

```

        break

    # Jika client keluar
    print(f"[-] {addr} putus koneksi.")
    broadcast(f"[INFO] {addr} keluar dari server.\n".encode('utf-8'), client_socket)
    if client_socket in clients:
        clients.remove(client_socket)
    client_socket.close()

def main():
    server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server.bind((IP, PORT))
    server.listen()
    print(f"[*] Server berjalan di {IP}:{PORT}")

    while True:
        client_socket, addr = server.accept()
        if len(clients) < MAX_CLIENTS:
            clients.append(client_socket)
            thread = threading.Thread(target=handle_client, args=(client_socket, addr))
            thread.start()
            print(f"[Aktif] Jumlah koneksi: {len(clients)}")
        else:
            print(f"[!] Menolak {addr}: server penuh.")
            client_socket.send("Server penuh. Silakan coba lagi nanti.\n".encode('utf-8'))
            client_socket.close()

if __name__ == "__main__":
    main()

```

## 2. Membuat Program client

Buat file client.py dengan kode berikut:

```
#!/usr/bin/env python3
```

```
import socket
import threading
```

```
# Konfigurasi Client
SERVER_IP = '127.0.0.1'
SERVER_PORT = 3606
```

```
def receive_messages(client):
```

```

while True:
    try:
        message = client.recv(1024).decode('utf-8')
        if not message:
            break
        print(message)
    except:
        print("[!] Terputus dari server.")
        client.close()
        break

def main():
    client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    try:
        client.connect((SERVER_IP, SERVER_PORT))
    except:
        print("[!] Gagal terhubung ke server.")
        return

    print("[*] Terhubung ke server. Ketik 'exit' untuk keluar.")

    # Thread untuk menerima pesan
    thread = threading.Thread(target=receive_messages, args=(client,))
    thread.start()

    # Kirim pesan
    while True:
        message = input()
        if message.lower() == 'exit':
            break
        try:
            client.send(message.encode('utf-8'))
        except:
            print("[!] Koneksi putus.")
            break

    client.close()

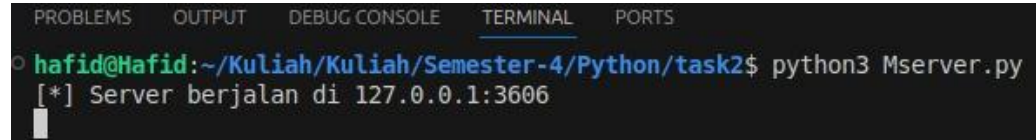
if __name__ == "__main__":
    main()

```



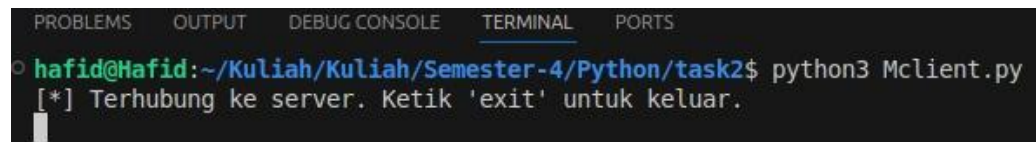
### 3. Output program

#### 1. Server dijalankan

A terminal window with tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is active. The prompt is hafid@Hafid:~/Kuliah/Kuliah/Semester-4/Python/task2\$. The command python3 Mserver.py has been executed, and the output is [\*] Server berjalan di 127.0.0.1:3606.

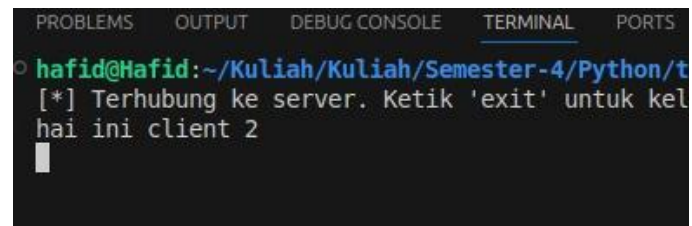
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
hafid@Hafid:~/Kuliah/Kuliah/Semester-4/Python/task2$ python3 Mserver.py
[*] Server berjalan di 127.0.0.1:3606
```

#### 2. Client 1 dijalankan

A terminal window with tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is active. The prompt is hafid@Hafid:~/Kuliah/Kuliah/Semester-4/Python/task2\$. The command python3 Mclient.py has been executed, and the output is [\*] Terhubung ke server. Ketik 'exit' untuk keluar.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
hafid@Hafid:~/Kuliah/Kuliah/Semester-4/Python/task2$ python3 Mclient.py
[*] Terhubung ke server. Ketik 'exit' untuk keluar.
```

#### 3. Client mengirimkan pesan

A terminal window with tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The TERMINAL tab is active. The prompt is hafid@Hafid:~/Kuliah/Kuliah/Semester-4/Python/t. The output shows [\*] Terhubung ke server. Ketik 'exit' untuk kel, followed by the user input hai ini client 2.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
hafid@Hafid:~/Kuliah/Kuliah/Semester-4/Python/t
[*] Terhubung ke server. Ketik 'exit' untuk kel
hai ini client 2
```

#### 4. Log pada server

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

[*] Server berjalan di 127.0.0.1:3606
[+] ('127.0.0.1', 59582) terhubung.
[Aktif] Jumlah koneksi: 1
[('127.0.0.1', 59582)] hai
[('127.0.0.1', 59582)] ini client 1
[+] ('127.0.0.1', 39488) terhubung.
[Aktif] Jumlah koneksi: 2
[('127.0.0.1', 39488)] hai ini client 2
[-] ('127.0.0.1', 39488) putus koneksi.
```

#### 4. Penjelasan dan Analisa

##### Server (server.py)

Program server dibuat untuk menerima koneksi dari banyak client. Setiap kali ada client terhubung, server membuat thread baru untuk menangani komunikasi dengan client tersebut. Server juga melakukan broadcast ke seluruh client untuk menyampaikan pesan yang diterima dari salah satu client.

Fitur utama:

- Menggunakan TCP socket.
- Menyimpan semua koneksi client dalam list clients.
- Setiap pesan yang diterima dari satu client akan dikirim ke client lain melalui fungsi broadcast.
- Menangani koneksi maksimum 5 client secara simultan.

##### Client (client.py)

Client menghubungkan diri ke server dan dapat mengirimkan serta menerima pesan. Pesan dari server akan ditampilkan di terminal.

Fitur utama:

- Terhubung ke server melalui IP dan PORT yang ditentukan.
- Menggunakan thread untuk menerima pesan secara paralel dari input pengguna.
- Dapat keluar dari chat dengan mengetik exit.

Analisa Output:

- Saat server dijalankan, ia mendengarkan pada 127.0.0.1:3606.

- Saat client terhubung, muncul pesan pada server bahwa client terhubung.
- Pesan yang diketik oleh satu client akan muncul di terminal client lain, menunjukkan bahwa broadcast bekerja.
- Jika client keluar, server dan client lain menerima informasi tersebut.

Program ini berhasil menunjukkan komunikasi dua arah (atau lebih) antar client melalui perantara server, sesuai dengan konsep dasar chat room.

### BAB III PENUTUP

#### A. Kesimpulan

Melalui praktikum ini, mahasiswa mampu memahami dan mengimplementasikan komunikasi antar komputer menggunakan socket di Python. Program chat sederhana ini menunjukkan bagaimana server dapat menangani banyak client dengan multithreading, dan bagaimana pesan dapat ditransmisikan secara efisien melalui jaringan lokal menggunakan protokol TCP.