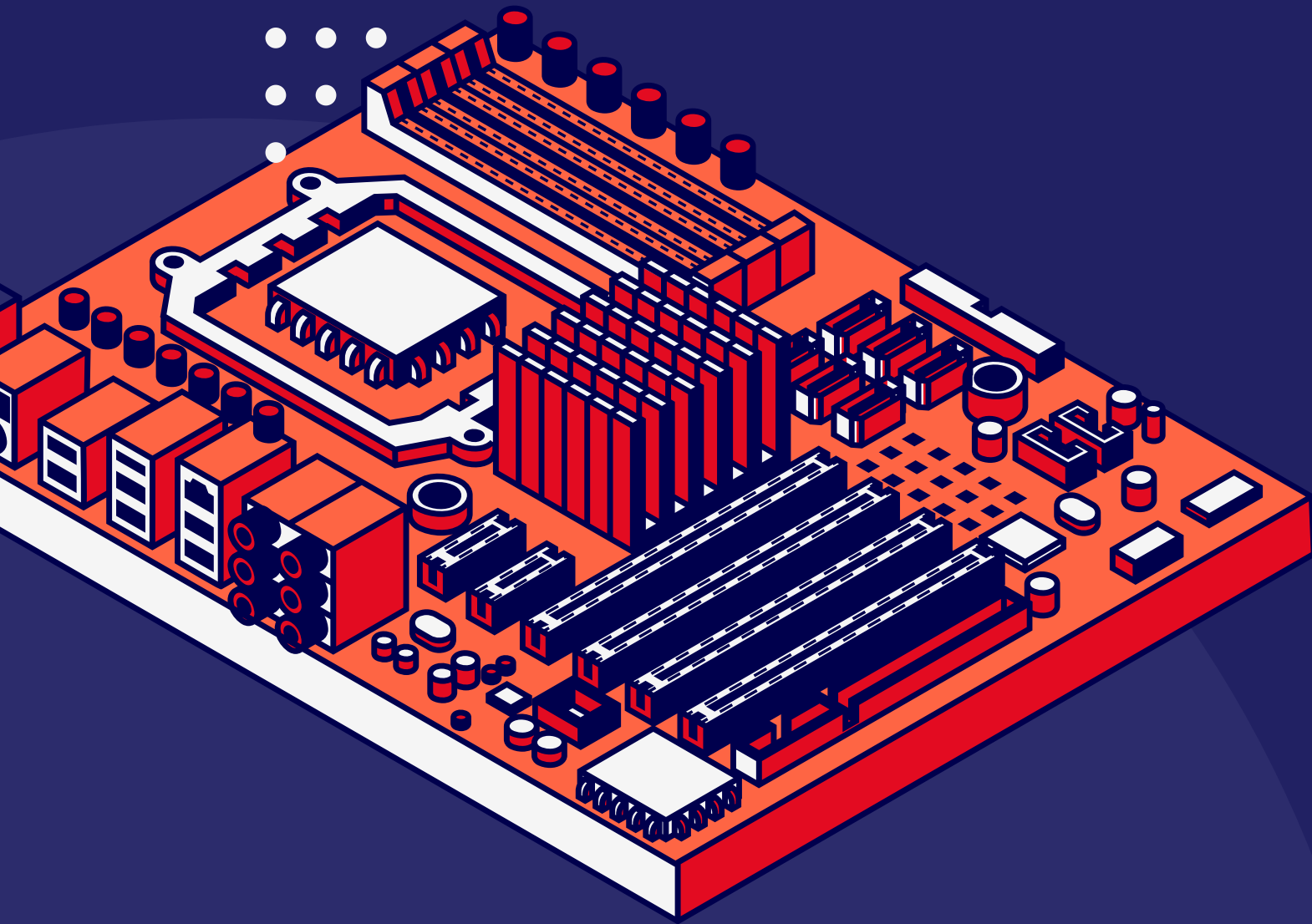


Jaringan Komputer

Kelompok YTTA





Single Thread

Hanya satu thread yang aktif.
Server melayani satu klien dalam satu waktu.

Konsep

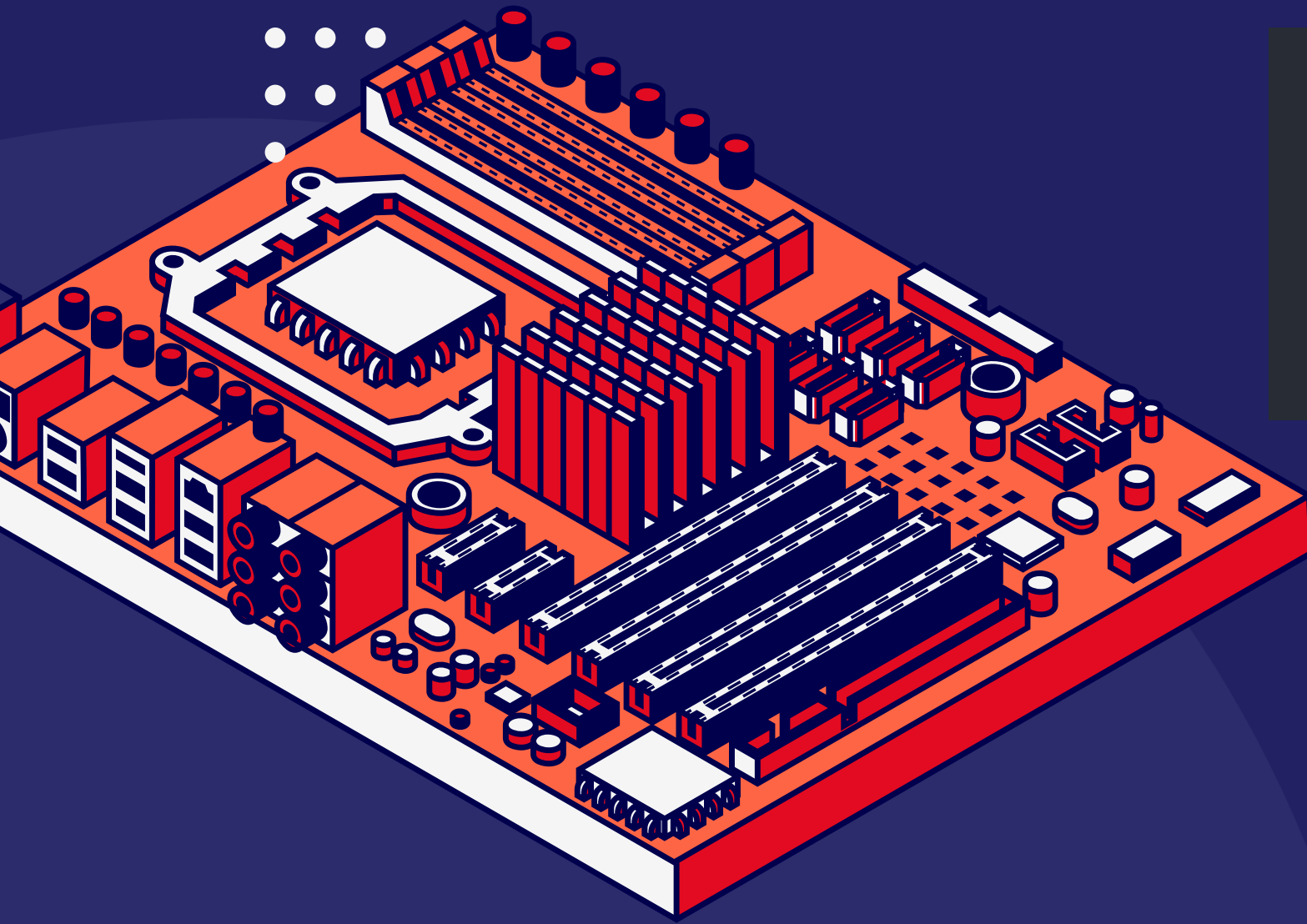
- Setiap koneksi dijalankan di thread terpisah.
- Beberapa klien bisa dilayani secara bersamaan.

Kelebihan

- Lebih cepat, efisien untuk banyak klien.
- Klien tidak harus menunggu giliran.

Kekurangan:

- Lebih kompleks (perlu manajemen thread).
- Konsumsi memori lebih tinggi.



```
HOST = '0.0.0.0'  
PORT = 6789
```

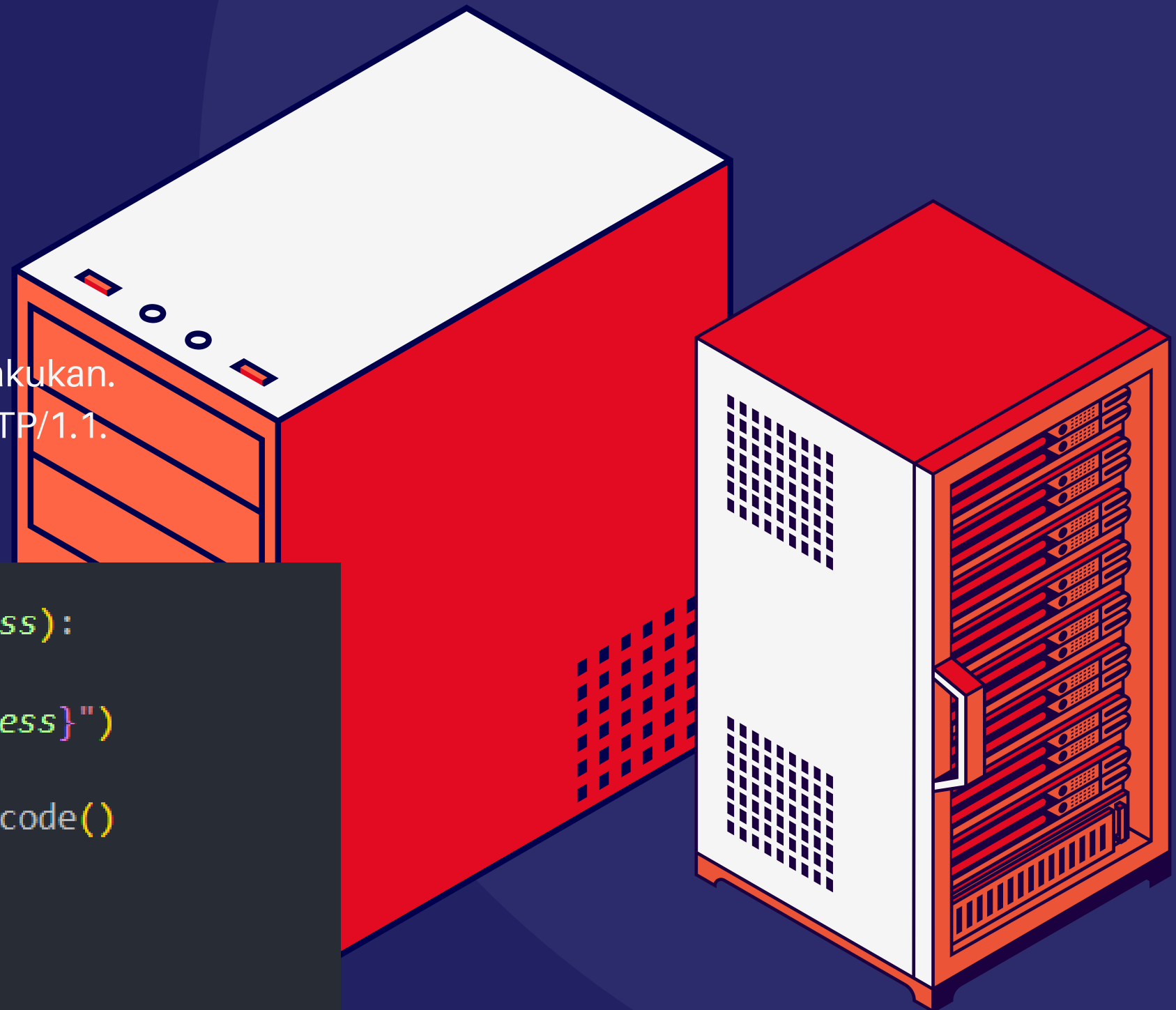
- HOST = '0.0.0.0' untuk server menerima koneksi dari semua alamat IP (localhost & jaringan).
- PORT = 6789 untuk port server. Bisa diakses dari browser: <http://localhost:6789/>.

Konfigurasi ke Server

Fungsi handle_client()

- Fungsi ini bertanggung jawab melayani 1 klien.
- Di sinilah proses membaca request dan membalas file dilakukan.
- Ambil method dan path dengan contoh GET /file.html HTTP/1.1.
- Buang tanda / di depan path menjadi nama file lokal.

```
def handle_client(client_socket, client_address):  
    try:  
        print(f"[+] Koneksi dari {client_address}")  
  
        request = client_socket.recv(1024).decode()  
        print(f"[REQUEST] {request}")  
  
        lines = request.splitlines()  
        if len(lines) > 0:  
            method, path, _ = lines[0].split()  
            filename = path.lstrip('/')  
            # ... (rest of the function code) ...
```



Pengangan

Request

```
if method != 'GET':  
    response = "HTTP/1.1 405 Method Not Allowed\r\n\r\nMethod Not Allowed"  
    client_socket.sendall(response.encode())  
elif not os.path.isfile(filename):  
    response = "HTTP/1.1 404 Not Found\r\n\r\n404 File Not Found"  
    client_socket.sendall(response.encode())
```

Cek apakah method GET, dan file benar-benar ada.

Simulasi Delay

- Delay 3 detik untuk mensimulasikan server sedang sibuk.

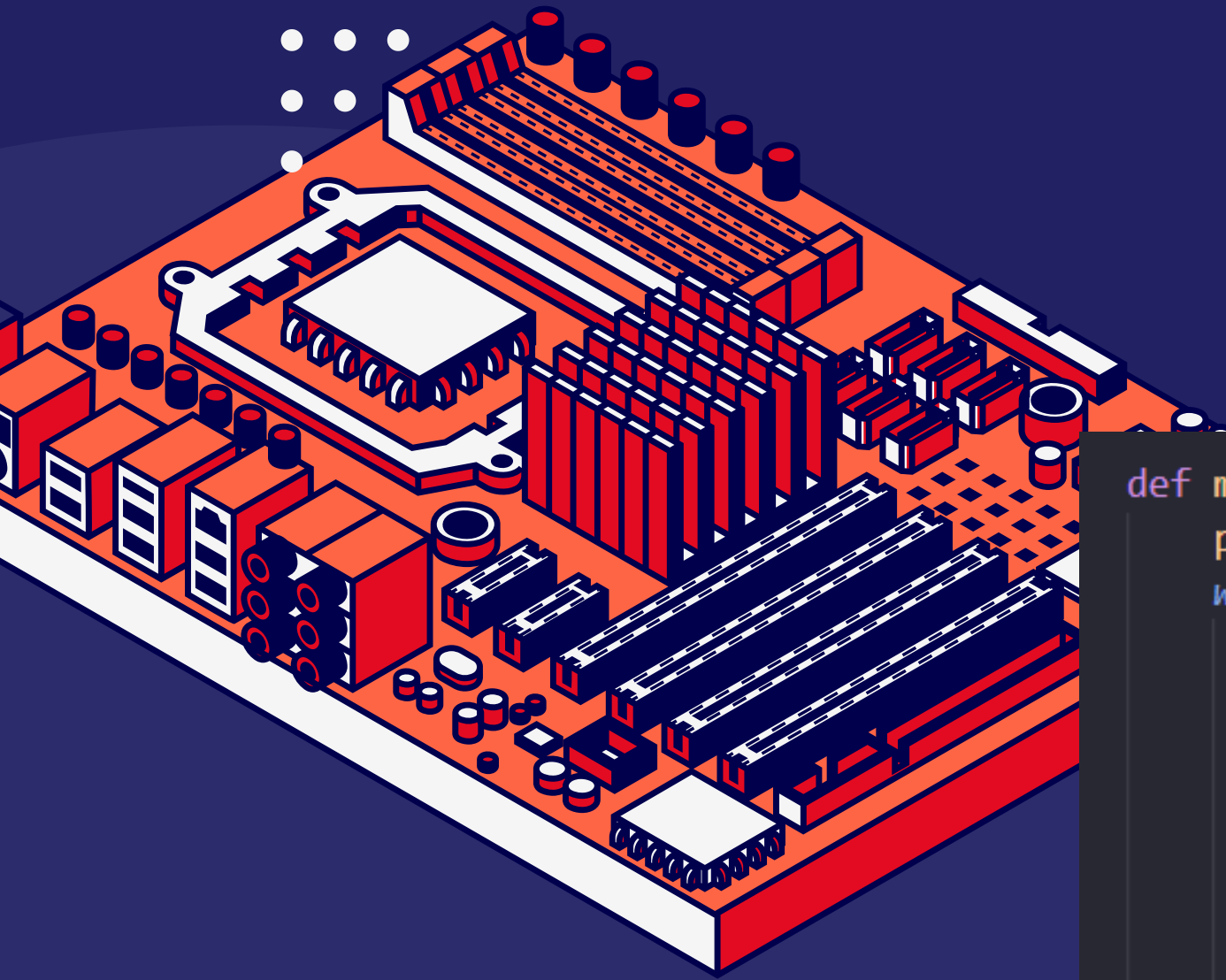
```
else:
```

```
    print("[DEBUG] Delay 3 detik dimulai...")
```

```
    time.sleep(3)
```

```
    print("[DEBUG] Delay selesai. Mengirim file.")
```

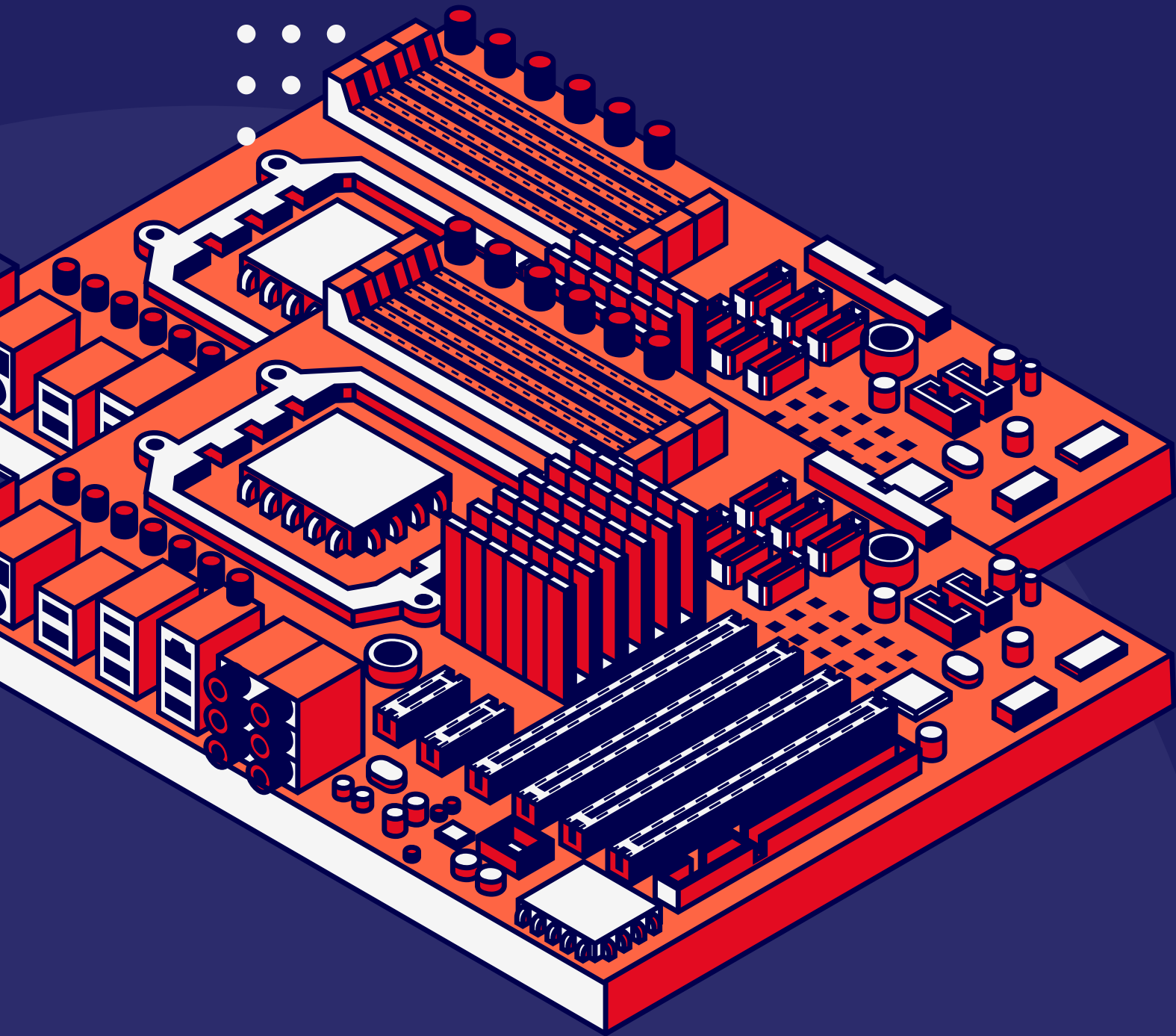




Loop Utama main()

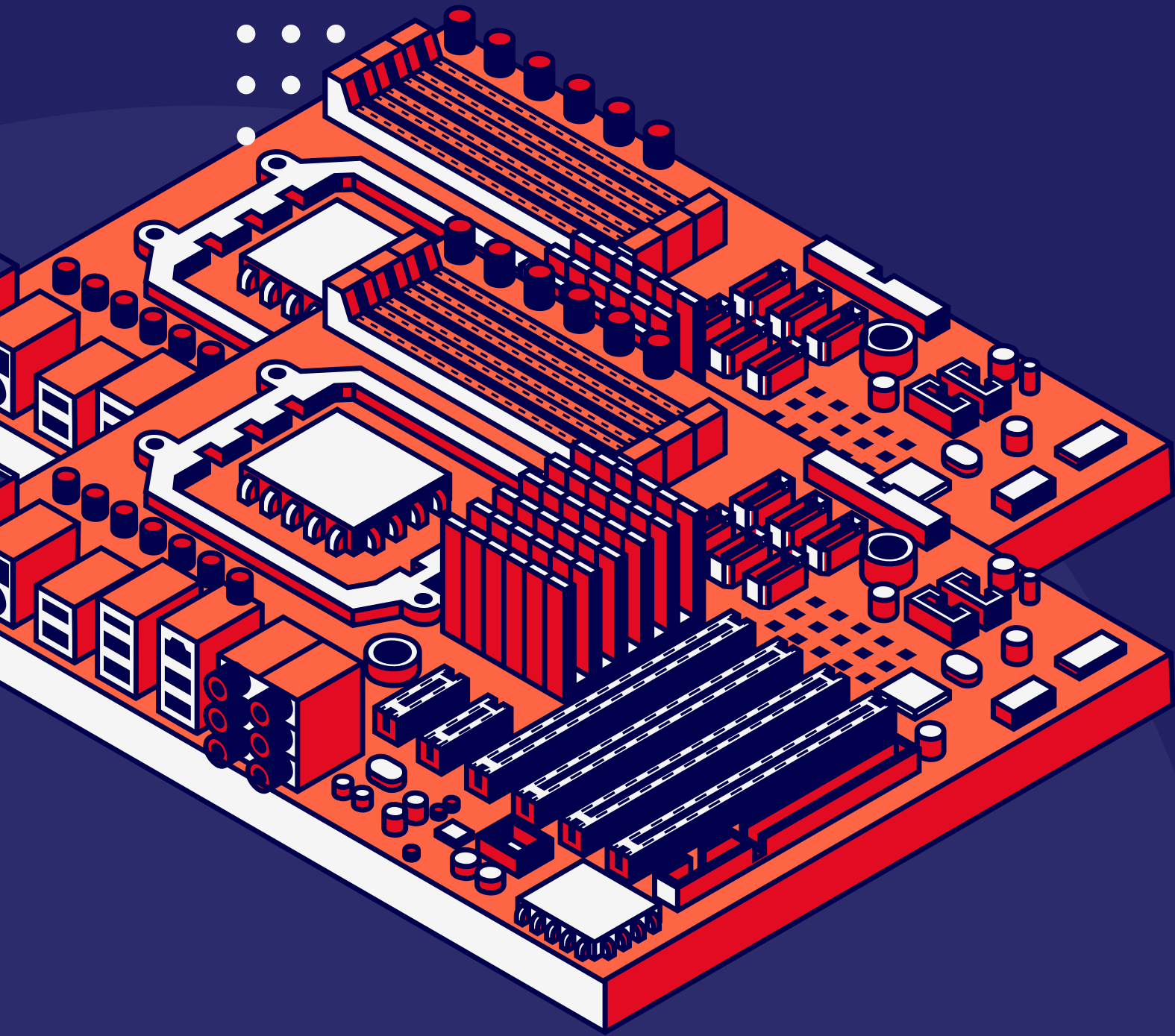
```
def main():  
    print("[*] Server SINGLE THREAD sedang berjalan")  
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:  
        server_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)  
        server_socket.bind((HOST, PORT))  
        server_socket.listen(1)  
        print(f"[*] Listening di {HOST}:{PORT}")  
  
    while True:  
        client_socket, client_address = server_socket.accept()  
        handle_client(client_socket, client_address)
```

- Server menerima koneksi satu per satu.
- Tidak ada thread klien berikutnya menunggu.



Multi Thread

- Satu program bisa punya banyak thread yang jalan paralel.
- Masing-masing thread bisa menjalankan tugas yang berbeda atau menangani klien yang berbeda.



Konfigurasi Server

```
HOST = '0.0.0.0'  
PORT = 6789
```

Fungsi handle_client()

- Fungsi ini bertanggung jawab melayani banyak klien.
- Di sinilah proses membaca request dan membalas file dilakukan.
- Ambil method dan path dengan contoh GET /file.html HTTP/1.1.
- Buang tanda / di depan path menjadi nama file lokal.

```
def handle_client(client_socket, client_address):
    try:
        print(f"[+] Koneksi dari {client_address}")

        request = client_socket.recv(1024).decode()
        print(f"[REQUEST] {request}")

        lines = request.splitlines()
        if len(lines) > 0:
            method, path, _ = lines[0].split()
            filename = path.lstrip('/')

            if method != 'GET':
                response = "HTTP/1.1 405 Method Not Allowed\r\n\r\nMethod Not Allowed"
                client_socket.sendall(response.encode())
            elif not os.path.isfile(filename):
                response = "HTTP/1.1 404 Not Found\r\n\r\n404 File Not Found"
                client_socket.sendall(response.encode())
            else:
                with open(filename, 'rb') as f:
                    content = f.read()

                header = (
                    "HTTP/1.1 200 OK\r\n"
                    f"Content-Length: {len(content)}\r\n"
                    "Content-Type: text/html\r\n"
                    "\r\n"
                ).encode()

                client_socket.sendall(header + content)

    except Exception as e:
        print(f"[ERROR] {e}")
    finally:
        client_socket.close()
        print(f"[-] Koneksi ditutup dari {client_address}")
```

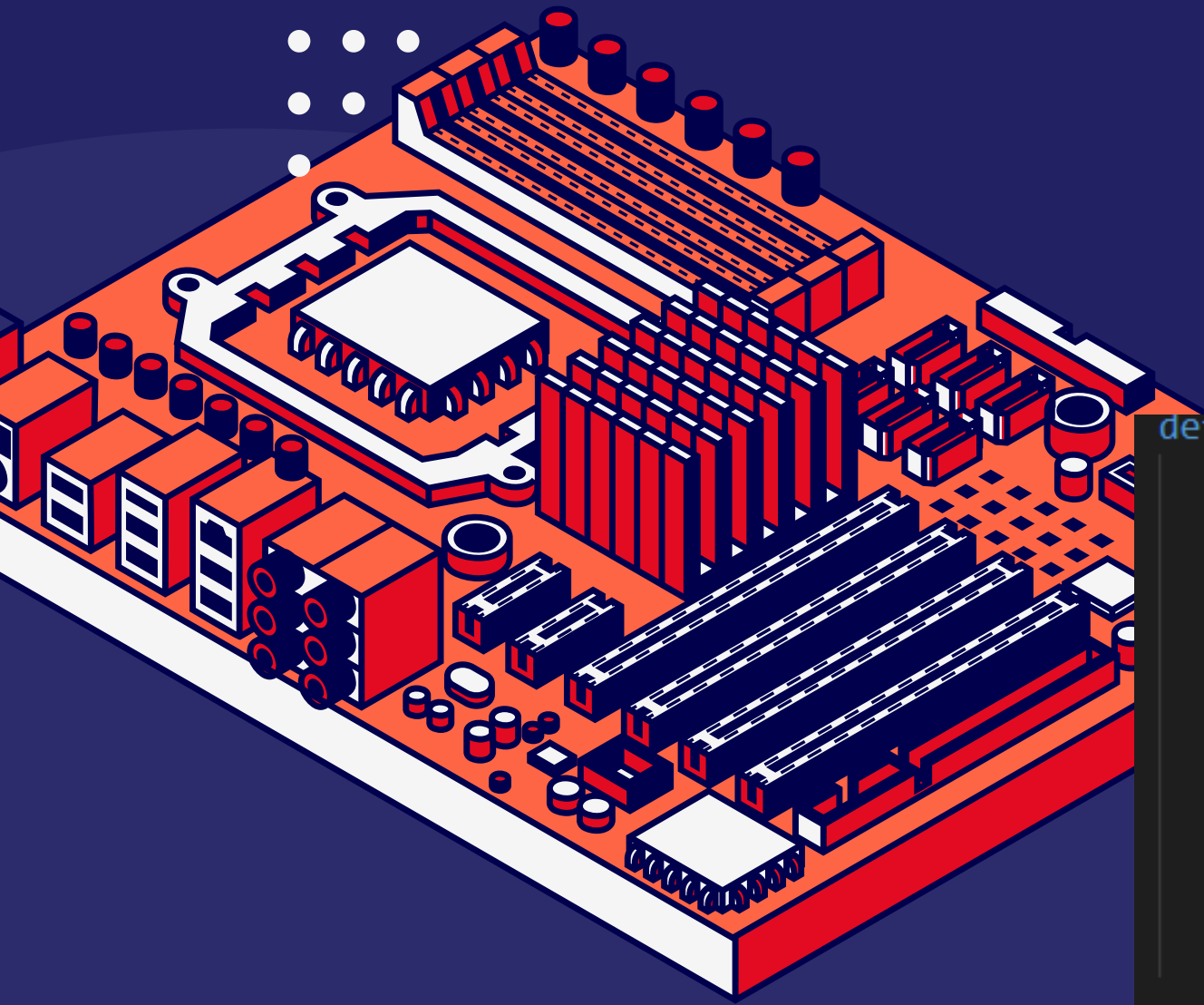
Pengangan Request

```
if method != 'GET':
    response = "HTTP/1.1 405 Method Not Allowed\r\n\r\nMethod Not Allowed"
    client_socket.sendall(response.encode())
elif not os.path.isfile(filename):
    response = "HTTP/1.1 404 Not Found\r\n\r\n404 File Not Found"
    client_socket.sendall(response.encode())
else:
    with open(filename, 'rb') as f:
        content = f.read()

    header = (
        "HTTP/1.1 200 OK\r\n"
        f"Content-Length: {len(content)}\r\n"
        "Content-Type: text/html\r\n"
        "\r\n"
    ).encode()

    client_socket.sendall(header + content)
```

Cek apakah method GET, dan file benar-benar ada.

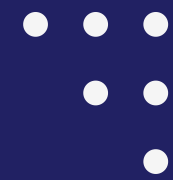


Loop Utama main()

```
def main():
    print("[*] Server MULTI THREAD sedang berjalan")
    with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:
        server_socket.bind((HOST, PORT))
        server_socket.listen(5)
        print(f"[*] Listening di {HOST}:{PORT}")

        while True:
            client_socket, client_address = server_socket.accept()
            thread = threading.Thread(target=handle_client, args=(client_socket, client_address))
            thread.start()

if __name__ == "__main__":
    main()
```



MKSH

(> _ <)

+62 896-0416-2102

<https://github.com/Hiiroga/Tubes-Jaringan-Komputer>

////////////////////////////////////