

Tugas 4 Tugas 4

Http server 1

- Dengan berbasis pada program http server pada
<https://github.com/rm77/progjar/tree/master/progjar5>
 - Dengan method yang ada pada spesifikasi HTTP server, tambahkan kemampuan http server untuk
 - Melihat daftar file pada satu direktori
 - Mengupload sebuah file
 - Menghapus file
 - Jalankan http server dengan mode
 - Thread pool
 - Process pool
 - Buatlah client implementation dari operasi tambahan tersebut. Jalankan operasi client server untuk kemampuan tersebut, berikanlah screenshot seperlunya, dan penjelasan dalam paragraf

Instruksi submission

- Semua dimasukkan ke satu file PDF
- Dalam satu file PDF berisi
 - Link menuju ke repository di github
 - Penjelasan tentang modifikasi yang dilakukan baik pada client ataupun pada server (minimal 2 paragraf)
 - Capture operasi client server untuk operasi upload dan hapus
 - Deskripsi dan penjelasan minimal 1 paragraf untuk capture screenshot

GITHUB REPO :

NABILAH ATIKA RAHMA – 5025221005

<https://github.com/ranabel/network-programming-5025221005>

Modifikasi pada Server

- **Modifikasi pada File [http.py](#)**

Saya menambahkan beberapa fitur utama pada HTTP server, yaitu untuk menampilkan daftar file dalam direktori, mengunggah file, dan menghapus file. Fitur unggah menggunakan metode POST, sedangkan penghapusan file menggunakan metode DELETE. Untuk tampilan daftar atau list file, server akan memberi nama file. Saya juga menambahkan validasi path untuk mencegah akses ke luar folder utama (directory traversal), dan membatasi ukuran maksimum file yang bisa diunggah. Server juga dilengkapi dengan logging yang komprehensif yang menampilkan aktivitas real time di console, sehingga administrator dapat memantau semua operasi yang dilakukan client.

- **Modifikasi pada Thread Pool dan Process Pool**

Saya menjalankan server dalam dua mode yaitu thread pool dan process pool. Untuk thread pool, saya menggunakan ThreadPoolExecutor dengan maksimal 20 thread, di mana setiap permintaan (request) ditangani oleh thread yang berbeda. Server dilengkapi dengan timeout handling, buffer optimization, dan connection management yang lebih baik. Masing-masing thread juga memiliki ID yang dicatat dalam log agar lebih mudah dilacak. Keduanya saya berikan penambahan log untuk pemantauan activity yang cukup kompleks agar mudah untuk dipahami. Error handling juga diperbaiki untuk menangani various network exceptions seperti connection reset dan timeout.

Modifikasi pada Client

Client saya modifikasi agar mendukung operasi menampilkan / listing file, mengunggah / upload, mengunduh / download, dan menghapus / delete file. Program client bisa dijalankan dalam mode interaktif dengan menu pilihan, atau

langsung lewat command line dengan argumen tertentu, hal tersebut juga sudah saya setup untuk lebih memantau dan meminimalkan ketidakmenauan bug. Operasi upload dan download menggunakan metode binary data handling, proper dengan progress tracking dan speed calculation. Client dapat menangani file dengan berbagai ukuran dan memberikan feedback real-time tentang progress upload. Saya juga menambahkan pengecekan kesalahan, seperti saat koneksi gagal atau file tidak ditemukan. Informasi aktivitas client akan tercatat di log untuk membantu proses debugging. Semua fitur dibuat sesederhana mungkin agar mudah digunakan. Semua operasi dibuat sejelas mungkin agar mudah digunakan siapa saja.

Berikut SS pada saat awal server open hingga client dan percobaan list, upload, download, delete file untuk thread dan process pool.

The screenshot shows a Visual Studio Code interface with two terminals and a browser. The left terminal shows the setup of a Process Pool Server:

```
PS C:\Users\Abel\Network Programming\5025221005\task01> python server_process_pool_http.py --host 127.0.0.1 --port 8881 --workers 4
2025-06-28 18:14:59,840 : INFO - Max workers: 4
2025-06-28 18:14:59,840 : INFO - Server ready to accept connections...
PS C:\Users\Abel\Network Programming\5025221005\task01>
```

The right terminal shows the configuration of a Thread Pool Server:

```
PS C:\Users\Abel\Network Programming\5025221005\task01> python server_thread_pool_http.py --host 127.0.0.1 --port 8880 --workers 4
2025-06-28 18:15:05,623 : INFO - Thread pool Server started on 127.0.0.1:8880
2025-06-28 18:15:09,623 : INFO - Max workers: 4
2025-06-28 18:15:09,623 : INFO - Server ready to accept connections...
PS C:\Users\Abel\Network Programming\5025221005\task01>
```

The browser window displays an 'HTTP FILE SERVER - THREAD POOL MODE' page with the URL `http://127.0.0.1:8880`. It shows a list of available endpoints:

- GET / - Serve info
- PUT /File - Upload file
- POST /filename - Upload file
- DELETE /filename - Delete file
- GET /filename - Download file

Berikut gambar ini menunjukkan server HTTP yang berjalan dalam mode thread pool pada port 8880, dan process pool pada port 8881 dengan tampilan awal yang menampilkan informasi server dan endpoint yang tersedia. Server log nantinya akan menampilkan aktivitas real-time termasuk thread ID dan processing time untuk setiap request. Disini juga terdapat address, max worker, dan working directory yang digunakan pada tugas ini.

```

PS C:\users\Abel\Network-Programming-5025221005\task4> python client.py --port 8880 --interactive
HTTP FILE CLIENT - INTERACTIVE MODE
-----
Server: 127.0.0.1:8880
Available operations:
1. List files
2. Upload file
3. Download file
4. Delete file
5. Exit
Select operation (1-5): 1
LISTING FILES
2025-06-28 18:15:32,880 - INFO - Connecting to ('127.0.0.1', 8880)
2025-06-28 18:15:32,881 - INFO - Sending request...
2025-06-28 18:15:32,885 - INFO - Received 15609 bytes from server
SUCCESS: File listing received
Found 20 files:
1. client.py
2. donabekb.jpg
3. http.py
4. page.html
5. perfect.sh
6. pokjan.jpg
7. resources.txt
8. research_center.jpg
9. resources.txt
10. resources.txt
11. sending.html
12. server_async_http.py
13. server_process_stream_http.py
14. server_process_http.py
15. server_thread_http.py
16. server_thread_http.py
17. server_thread_http_secure.py
18. socket_proxy.py
19. socket_proxy.py
20. testing.txt
Available operations:
1. List files
2. Upload file
3. Download file
4. Delete file
5. Exit
Select operation (1-5): 1
-----
```



```

PS C:\users\Abel\Network-Programming-5025221005\task4> python server_thread_pool_http.py --host 127.0.0.1 --port 8880 --workers 4
2025-06-28 18:15:59,840 - INFO - Thread Pool Server started on 127.0.0.1:8880
2025-06-28 18:15:59,840 - INFO - Max workers: 4
2025-06-28 18:15:59,840 - INFO - Server ready to accept connections...
-----
```



```

HTTP FILE SERVER - THREAD POOL MODE
Address: http://127.0.0.1:8880
Working Directory: C:\Users\Abel\Network-Programming-5025221005\task4
New workers: 4
Available endpoints:
  - Server Info
    □ GET /filename - List files
    □ POST /filename - Upload file
  - File Processing
    □ GET /filename - Upload file
    □ POST /filename - Download file
  - File Download
    □ GET /filename - Download file
Press Ctrl+C to stop server
-----
```



```

PS C:\users\Abel\Network-Programming-5025221005\task4> python server_process_pool_http.py --host 127.0.0.1 --port 8881 --workers 4
2025-06-28 18:15:59,623 - INFO - Process Pool Server started on 127.0.0.1:8881
2025-06-28 18:15:59,623 - INFO - Max workers: 4
2025-06-28 18:15:59,623 - INFO - Server ready to accept connections...
-----
```



```

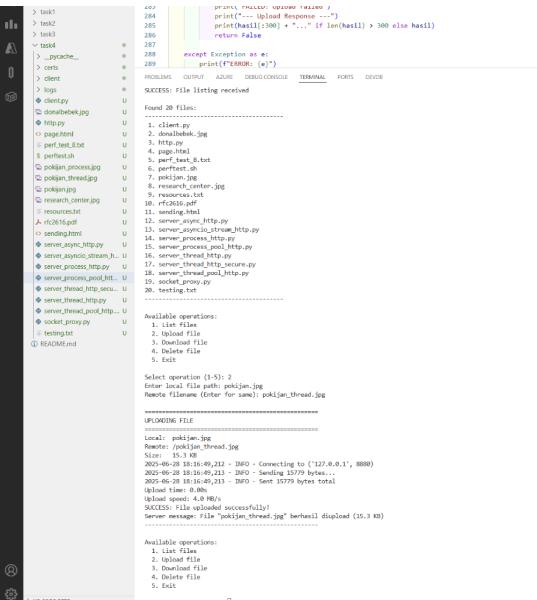
HTTP FILE SERVER - PROCESS POOL MODE
Address: http://127.0.0.1:8881
Working Directory: C:\Users\Abel\Network-Programming-5025221005\task4
Available endpoints:
  - Server Info
    □ GET /filename - List files
    □ POST /filename - Upload file
  - File Processing
    □ GET /filename - Upload file
    □ POST /filename - Download file
  - File Download
    □ GET /filename - Download file
Press Ctrl+C to stop server
-----
```



```

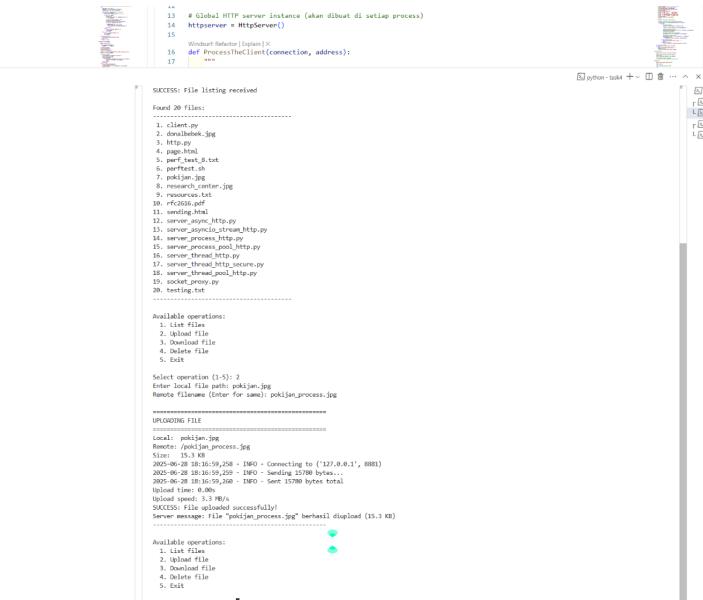
2025-06-28 18:15:34,973 - INFO - New connection from ('127.0.0.1', 55605)
2025-06-28 18:15:34,992 - INFO - Active connections: 1
2025-06-28 18:15:35,194 - INFO - [Process-2892] Processing connection from ('127.0.0.1', 55605)
2025-06-28 18:15:35,194 - INFO - [Process-2892] Received 31 bytes headers + 0 bytes body from ('127.0.0.1', 55605)
2025-06-28 18:15:35,194 - INFO - [Process-2892] Received 31 bytes headers + 8 bytes body from ('127.0.0.1', 55605)
2025-06-28 18:15:35,194 - INFO - [Process-2892] Received 31 bytes headers + 8 bytes body from ('127.0.0.1', 55605)
2025-06-28 18:15:35,194 - INFO - [Process-2892] HTTP Server processing: GET /
2025-06-28 18:15:35,194 - INFO - [Process-2892] Processing GET requests until /
2025-06-28 18:15:35,194 - INFO - [Process-2892] Scanning directory: /
2025-06-28 18:15:35,194 - INFO - [Process-2892] Found 20 files in directory
2025-06-28 18:15:35,194 - INFO - [Process-2892] Generating response: 200 OK (15777 bytes)
2025-06-28 18:15:35,194 - INFO - [Process-2892] Completed ("127.0.0.1", 55605) in 0.009s
2025-06-28 18:15:35,200 - INFO - [Process-2892] Completed ("127.0.0.1", 55605) in 0.014s
2025-06-28 18:15:35,208 - INFO - [Process-2892] Connection ('127.0.0.1', 55605) closed (total: 0.014s)
-----
```

Pada gambar diatas ditunjukkan sisi client dan dibawah ditunjukkan di sisi server. Pada awalnya kita diberi pilihan ingin melakukan apa, list, upload, download, delete, atau exit dan terlihat bahwasannya jika kita memilih list, maka function akan mencari dan akan ditampilkan listing files yang ada dalam directory saat ini (pada directory tersebut terdapat 20 files mulai dari client.py sampai testing.txt), disini juga dapat kita lihat kita sedang di port dan address mana.



```

> task1
  285     print("Success: %s" % file)
  286     print("Upload Response -->")
  287     print(hasil1[1500] + "...")
  288     return False
  289
  290 except Exception as e:
  291     print("ERROR: (%s)" % e)
  292
  293
  294 # Global HTTP server instance (akan dibuat di setup process)
  295 httpserver = HttpServer()
  296
  297 # Handler untuk menangani koneksi
  298 def ProcessClient(connection, address):
  299     ***
  
```



```

PROBLEMS OUTPUT AZURE DEBUG CONSOLE TERMINAL PORTS DEVOB
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

Select operation (1-5): 2
Enter local file path: pokjan_thread.jpg
Remote filename (Enter for same): pokjan_process.jpg
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

Select operation (1-5): 2
Enter local file path: pokjan.jpg
Remote filename (Enter for same): pokjan_thread.jpg
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

Select operation (1-5): 1
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

Select operation (1-5): 1
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

Select operation (1-5): 1
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

Select operation (1-5): 1
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

Select operation (1-5): 1
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

Select operation (1-5): 1
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

Select operation (1-5): 1
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

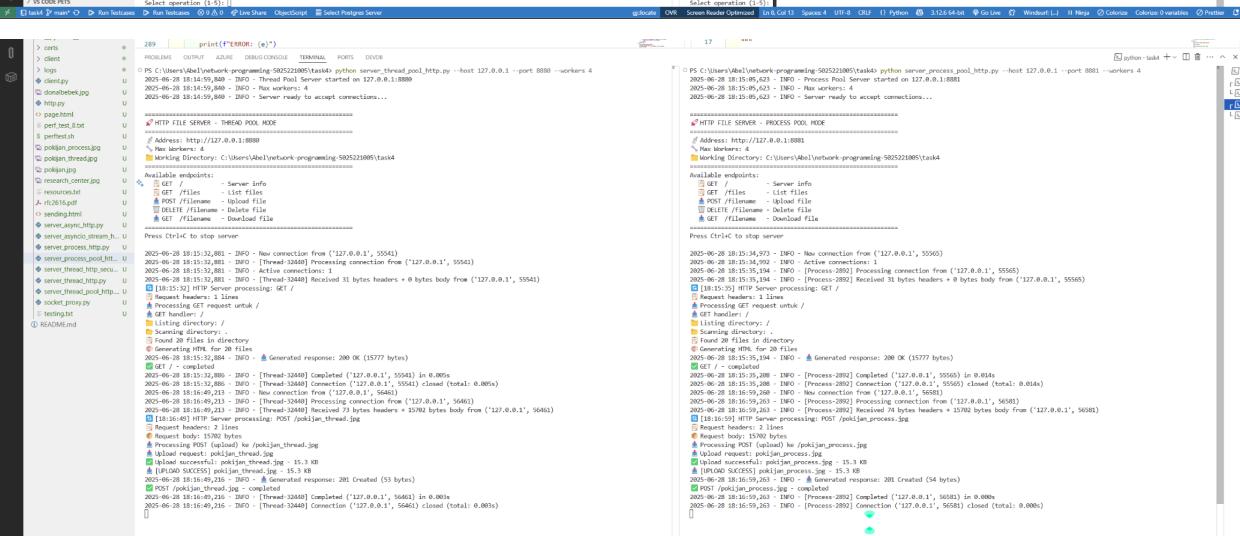
Select operation (1-5): 1
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

Select operation (1-5): 1
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

Select operation (1-5): 1
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

Select operation (1-5): 1
Available operations:
1. List file
2. Upload file
3. Download file
4. Delete file
5. Exit

```



```

> task1
  289     print("ERROR: (%s)" % e)
  290
  291 # Global HTTP server instance (akan dibuat di setup process)
  292 httpserver = HttpServer()
  293
  294 # Handler untuk menangani koneksi
  295 def ProcessClient(connection, address):
  296     ***
  
```

Pada gambar diatas ditunjukkan sisi client dan dibawah ditunjukkan di sisi server. Sedangkan pada gambar upload menunjukkan client berhasil mengupload file ke server menggunakan metode POST ditampilkan lengkap dengan ukuran file, kecepatan transfer, dan durasi. Server mencatat file yang diterima, ukurannya, dan lokasi penyimpanan. Client menerima respons sukses dari server.

The screenshot displays two terminal windows. The left window shows a client-side file manager and operations:

- Uploading File:** A file named `pokjan_thread.jpg` is selected for upload. The status shows "Selected operation (1-5): 2" and "Enter local file path: pokjan.jpg". The log indicates "INFO - Sending request..." and "INFO - Received 15799 bytes from server".
- Downloading File:** A file named `pokjan_process.jpg` is selected for download. The status shows "Selected operation (1-5): 3" and "Enter remote filename: pokjan_process.jpg". The log indicates "INFO - Received 15798 bytes from server".
- File Operations:** Other operations like list, upload, download, delete, and exit are shown.

The right window shows a server-side log (Windows Reflect) for the process:

- Uploading File:** A file named `pokjan_thread.jpg` is uploaded. The log shows "INFO - Connecting to ('127.0.0.1', 8888)", "INFO - Sending 15799 bytes...", and "INFO - Sent 15799 bytes total".
- Processing File:** The file is processed. The log shows "INFO - File uploaded successfully!" and "INFO - File uploaded successfully!".
- File Operations:** Other operations like list, upload, download, delete, and exit are shown.

Pada gambar diatas ditunjukkan sisi client dan dibawah ditunjukkan di sisi server. Tidak jauh berbeda seperti upload yang menunjukkan client berhasil mengupload file ke server menggunakan metode POST ditampilkan lengkap dengan ukuran file, kecepatan transfer, dan durasi. Server mencatat file yang diterima, ukurannya, dan lokasi penyimpanan. Client menerima respons sukses dari server. Begitupun juga sama dengan download.

```

VS CODE PETES Select operation (1-5): 1
1. List files
2. Upload file
3. Download file
4. Delete file
5. Exit

VS CODE PETES Select operation (1-5): 1
1. List files
2. Upload file
3. Download file
4. Delete file
5. Exit

VS CODE PETES Select operation (1-5): 1
1. List files
2. Upload file
3. Download file
4. Delete file
5. Exit

```

The terminal window shows two sessions of file listing operations. The first session is for 'client.py' and the second is for 'testing.txt'. Both sessions show the same operations: List files, Upload file, Download file, Delete file, and Exit.

```

VS CODE PETES Select operation (1-5): 1
1. List files
2. Upload file
3. Download file
4. Delete file
5. Exit

VS CODE PETES Select operation (1-5): 1
1. List files
2. Upload file
3. Download file
4. Delete file
5. Exit

VS CODE PETES Select operation (1-5): 1
1. List files
2. Upload file
3. Download file
4. Delete file
5. Exit

```

This screenshot shows more detailed operations for both 'client.py' and 'testing.txt'. It includes additional options like 'Scanning directory /' and 'Generating HTML for 20 files'. The logs show various HTTP requests and responses, such as 'Processing GET request untuk /', 'POST /upload completed', and 'POST /download completed'.

Pada gambar diatas ditunjukkan sisi client dan dibawah ditunjukkan di sisi server. Gambar diatas tidak jauh berbeda dengan listing diawal, namun ddisini saya memberikannya untuk membedakan setelah dan sebelum proses upload download, sehingga disini menambah 2 file. Function akan mencari dan akan ditampilkan listing files yang ada dalam directory saat ini (pada directory tersebut terdapat 22 files mulai dari client.py sampai testing.txt), disini juga dapat kita lihat kita sedang di port dan address mana.

```

task4                                287
> _pycache_                          288 |         except Exception as e:
> certs                                289 |             print(f"Error: {e}")
> client                                290 |
> client                               291 |             os.system("clear")
> client                                292 |             print("   O U T P U T   A R C H I V E S   T E R M I N A L   P O R T S   D E V O")
> client                                293 |             print("-----")
> client                                294 |             print("1. Client")
> client                                295 |             print("2. downloadBeck.jpg")
> client                                296 |             print("3. upload_file.py")
> client                                297 |             print("4. page.html")
> client                                298 |             print("5. port_test.py.txt")
> client                                299 |             print("6. port_test.py")
> client                                300 |             print("7. pokijan.jpg")
> client                                301 |             print("8. pokijan_process.jpg")
> client                                302 |             print("9. pokijan_thread.jpg")
> client                                303 |             print("10. research_center.jpg")
> client                                304 |             print("11. rfc3164.pdf")
> client                                305 |             print("12. rfc6360.pdf")
> client                                306 |             print("13. server_error.jpg")
> client                                307 |             print("14. server_error.http.py")
> client                                308 |             print("15. server_asyncio_stream_http.py")
> client                                309 |             print("16. server_thread.http.py")
> client                                310 |             print("17. server_process_pool.py")
> client                                311 |             print("18. server_thread.http.py")
> client                                312 |             print("19. server_thread.poolhttp.py")
> client                                313 |             print("20. server_thread_poolhttp.py")
> client                                314 |             print("21. socket_process.py")
> client                                315 |             print("22. testing.txt")
> client                                316 |
> client                                317 |             print("-----")
> client                                318 |             print("Available operations:")
> client                                319 |             print("1. list files")
> client                                320 |             print("2. Upload file")
> client                                321 |             print("3. Download file")
> client                                322 |             print("4. Delete file")
> client                                323 |             print("5. Exit")
> client                                324 |
> client                                325 |             Select operation (1-5): 4
> Enter filenameto delete: pokijan_thread.jpg
> Delete "pokijan_thread.jpg"? (y/n): y
> DELETING FILE
> File: pokijan_thread.jpg
2025-06-28 18:15:23,296  INFO - Connecting to ('127.0.0.1', 8080)
2025-06-28 18:15:23,297  INFO - Sending request...
2025-06-28 18:15:23,297  INFO - Received 187 bytes from server
SUCCESS: File deleted successfully!
Server message: File "pokijan_thread.jpg" berhasil dihapus

Available operations:
1. list files
2. Upload file
3. Download file
4. Delete file
5. Exit
Select operation (1-5): 1

```

```

task4                                288 |         except Exception as e:
> _pycache_                          289 |             print(f"Error: {e}")
> certs                                290 |
> client                                291 |             os.system("clear")
> client                                292 |             print("   O U T P U T   A R C H I V E S   T E R M I N A L   P O R T S   D E V O")
> client                                293 |             print("-----")
> client                                294 |             print("1. Client")
> client                                295 |             print("2. downloadBeck.jpg")
> client                                296 |             print("3. upload_file.py")
> client                                297 |             print("4. page.html")
> client                                298 |             print("5. port_test.py.txt")
> client                                299 |             print("6. port_test.py")
> client                                300 |             print("7. pokijan.jpg")
> client                                301 |             print("8. pokijan_process.jpg")
> client                                302 |             print("9. pokijan_thread.jpg")
> client                                303 |             print("10. research_center.jpg")
> client                                304 |             print("11. rfc3164.pdf")
> client                                305 |             print("12. rfc6360.pdf")
> client                                306 |             print("13. server_error.jpg")
> client                                307 |             print("14. server_error.http.py")
> client                                308 |             print("15. server_asyncio_stream_http.py")
> client                                309 |             print("16. server_thread.http.py")
> client                                310 |             print("17. server_process_pool.py")
> client                                311 |             print("18. server_thread.http.py")
> client                                312 |             print("19. server_thread.poolhttp.py")
> client                                313 |             print("20. server_thread_poolhttp.py")
> client                                314 |             print("21. socket_process.py")
> client                                315 |             print("22. testing.txt")
> client                                316 |
> client                                317 |             print("-----")
> client                                318 |             print("Available operations:")
> client                                319 |             print("1. list files")
> client                                320 |             print("2. Upload file")
> client                                321 |             print("3. Download file")
> client                                322 |             print("4. Delete file")
> client                                323 |             print("5. Exit")
> client                                324 |
> client                                325 |             Select operation (1-5): 1
> DELETING FILE
> File: pokijan_process.jpg
2025-06-28 18:15:35,298  INFO - Connecting to ('127.0.0.1', 8081)
2025-06-28 18:15:35,298  INFO - [Thread-32480] Completed ("127.0.0.1", 55565) in 0.016s
2025-06-28 18:15:35,298  INFO - [Thread-32480] Connection ("127.0.0.1", 55565) closed (total: 0.016s)
2025-06-28 18:15:35,299  INFO - [Thread-32480] From ('127.0.0.1', 55565) to ('127.0.0.1', 8081)
2025-06-28 18:15:35,299  INFO - [Thread-32480] From ('127.0.0.1', 8081) to ('127.0.0.1', 55568)
2025-06-28 18:15:35,299  INFO - [Thread-32480] Received 137 bytes from server
SUCCESS: File deleted successfully!
Server message: File "pokijan_process.jpg" berhasil dihapus

Available operations:
1. list files
2. Upload file
3. Download file
4. Delete file
5. Exit
Select operation (1-5): 1

```

Pada gambar diatas ditunjukkan sisi client dan dibawah ditunjukkan di sisi server. Pada gambar diatas menunjukkan client menghapus file menggunakan metode DELETE. Sebelum menghapus, client meminta konfirmasi. Server memeriksa keberadaan file, lalu menghapusnya. Respons dari server menyatakan file berhasil dihapus, dan log mencatat waktu serta nama file yang dihapus.

The screenshot displays two terminal windows side-by-side. The left window shows the client's perspective, and the right window shows the server's perspective. Both terminals are running Python scripts and outputting logs related to file operations and network connections.

Client Terminal Output:

```

> _precache_
  288     except Exception as e:
  289         print("ERROR: (%s)" % e)
> client
  e  PROBLEMS OUTPUT AZURE DEBUG CONSOLE TERMINAL PORTS DEVR
> logs
  e  SUCCESS: File listing received
> http
  U  Found 22 files:
  1. client.py
  2. download.jpg
  3. index.html
  4. page.html
  5. perf_test_8.txt
  6. pokjan.jpg
  7. pokjan.jpg
  8. pokjan_process.jpg
  9. pokjan_process.pdf
  10. research_center.jpg
  11. sending.html
  12. rfc0565.pdf
  13. sending.html
  14. server_async_http.py
  15. server_async_stream_http.py
  16. server_process_http.py
  17. server_process_pool_http.py
  18. server_thread_http.py
  19. server_thread_pool_http.py
  20. server_thread_pool_http.py
  21. socket_proxy.py
  22. testing.txt
> README.md
Available operations:
  1. List files
  2. Upload file
  3. Download file
  4. Delete file
  5. Exit
Select operation (1-5): 4
Enter filename to delete: pokjan_thread.jpg (y/N): y
DELETE FILE
File: pokjan_thread.jpg
  15  [Windart Reactor] [explain] X
  16  def ProcessTheClient(connection, address):
  17  ***
  18  except Exception as e:
  19  print("ERROR: (%s)" % e)
  20  e  PROBLEMS OUTPUT AZURE DEBUG CONSOLE TERMINAL PORTS DEVR
  21  logs
  22  SUCCESS: File listing received
  23  Found 22 files:
  24  1. client.py
  25  2. download.jpg
  26  3. index.html
  27  4. page.html
  28  5. perf_test_8.txt
  29  6. pokjan.jpg
  30  7. pokjan.jpg
  31  8. pokjan_process.jpg
  32  9. pokjan_process.pdf
  33  10. research_center.jpg
  34  11. sending.html
  35  12. rfc0565.pdf
  36  13. sending.html
  37  14. server_async_http.py
  38  15. server_async_stream_http.py
  39  16. server_process_http.py
  40  17. server_process_pool_http.py
  41  18. server_thread_http.py
  42  19. server_thread_pool_http.py
  43  20. server_thread_pool_http.py
  44  21. socket_proxy.py
  45  22. testing.txt
  46  47. README.md
Available operations:
  1. List files
  2. Upload file
  3. Download file
  4. Delete file
  5. Exit
Select operation (1-5): 4
Enter filename to delete: pokjan_process.jpg (y/N): y
Delete: pokjan_process.jpg (y/N): y
  15  [Windart Reactor] [explain] X
  16  def ProcessTheClient(connection, address):
  17  ***
  18  except Exception as e:
  19  print("ERROR: (%s)" % e)
  20  e  PROBLEMS OUTPUT AZURE DEBUG CONSOLE TERMINAL PORTS DEVR
  21  logs
  22  SUCCESS: File listing received
  23  Found 22 files:
  24  1. client.py
  25  2. download.jpg
  26  3. index.html
  27  4. page.html
  28  5. perf_test_8.txt
  29  6. pokjan.jpg
  30  7. pokjan.jpg
  31  8. pokjan_process.jpg
  32  9. pokjan_process.pdf
  33  10. research_center.jpg
  34  11. sending.html
  35  12. rfc0565.pdf
  36  13. sending.html
  37  14. server_async_http.py
  38  15. server_async_stream_http.py
  39  16. server_process_http.py
  40  17. server_process_pool_http.py
  41  18. server_thread_http.py
  42  19. server_thread_pool_http.py
  43  20. server_thread_pool_http.py
  44  21. socket_proxy.py
  45  22. testing.txt
  46  47. README.md
Available operations:
  1. List files
  2. Upload file
  3. Download file
  4. Delete file
  5. Exit
Select operation (1-5): 5
Goodbye!
```

Server Terminal Output:

```

> taskkill
  286     return False
> task4
  287
> _precache_
  288     except Exception as e:
  289         print("ERROR: (%s)" % e)
> cors
  e  PROBLEMS OUTPUT AZURE DEBUG CONSOLE TERMINAL PORTS DEVR
> logs
  e  SUCCESS: File listing received
> http
  U  Found 22 files:
  1. client.py
  2. download.jpg
  3. index.html
  4. page.html
  5. perf_test_8.txt
  6. pokjan.jpg
  7. pokjan.jpg
  8. pokjan_process.jpg
  9. pokjan_process.pdf
  10. research_center.jpg
  11. sending.html
  12. rfc0565.pdf
  13. sending.html
  14. server_async_http.py
  15. server_async_stream_http.py
  16. server_process_http.py
  17. server_process_pool_http.py
  18. server_thread_http.py
  19. server_thread_pool_http.py
  20. server_thread_pool_http.py
  21. socket_proxy.py
  22. testing.txt
  23. README.md
Available operations:
  1. List files
  2. Upload file
  3. Download file
  4. Delete file
  5. Exit
Select operation (1-5): 5
Goodbye!
```

The client terminal shows the user interacting with the application, selecting options to list files, delete files, and finally exit. The server terminal shows the application processing these requests and responding with file listings and success messages.

Pada gambar diatas ditunjukkan sisi client dan dibawah ditunjukkan di sisi server. Disini karena semua sudah dicoba maka ada menu nomer 5 yaitu exit. Jika client memilih exit maka akan say goodbye dan server akan close connection sebelumnya.

Server HTTP ini juga memiliki beberapa fitur tambahan untuk meningkatkan kenyamanan dan keamanan penggunaan. Beberapa di antaranya adalah path validation, file size limits, dan input sanitization untuk mencegah penyalahgunaan. Server juga mencatat semua aktivitas melalui log, sehingga mudah dipantau. Dari sisi performa, saya menyesuaikan buffer dan koneksi agar server bisa menangani banyak request secara bersamaan. Penanganan error juga diperhatikan, misalnya untuk masalah jaringan, file tidak ditemukan, atau request yang tidak valid. Untuk client, saya menyediakan beberapa mode seperti interaktif, demo, dan command-line agar lebih fleksibel digunakan dan mudah diuji.