Muhammad Nabil Fadhil
5025221200

1. Implementasi Thread Pool HTTP Server dengan fitur Melihat daftar file pada satu direktori,Mengupload sebuah file dan Menghapus file
    a. Penjelasan Modifikasi

```
(base) jovyan@a34a93dcdc3f:~/work$ python server_thread_pool_http.py
Thread Pool HTTP Server running on port 8885
Request #1 from ('172.16.16.102', 34110)
Thread Status - Running: 0, Done: 1
Request #2 from ('172.16.16.102', 51742)
Thread Status - Running: 0, Done: 0
Request #3 from ('172.16.16.102', 52016)
Thread Status - Running: 0, Done: 1
Request #4 from ('172.16.16.102', 51004)
Thread Status - Running: 0, Done: 1
```

Saya menambahkan Peningkatan Buffer Size: Buffer size ditingkatkan dari 32 bytes menjadi 1024 bytes untuk mendukung upload file, Menambahkan sistem monitoring yang lebih detail untuk melacak status thread, Menambahkan konfigurasi port yang fleksibel via parameter, Meningkatan penanganan error dan logging

b. Screenshot Melihat daftar file pada satu direktori

Muhammad Nabil Fadhil
5025221200

```
=== HTTP Client Operations (Thread Pool Server: mesin1:8885) ===
1. List Directory
2. Upload File
3. Delete File
4. Test Connection
5. Re-detect Server
6. Exit
==============================
Select operation (1-6): 1
Listing directory...
WARNING:root:connecting to ('172.16.16.101', 8885)
WARNING:root:sending message
WARNING:root:GET /listdir HTTP/1.1
Host: localhost


WARNING:root:data received from server:
Server response:
HTTP/1.0 200 OK
Date: Sat Jul  5 13:16:01 2025
Connection: close
Server: myserver/1.0
Content-Length: 799
Content-Type:application/json

[
  {
    "name": "donalbebek.jpg",
    "size": 10729,
    "modified": 1746600804.968016
  },
  {
    "name": "gege.txt",
    "size": 10,
    "modified": 1751719922.58461
  },
  {
    "name": "pokijan.jpg",
    "size": 15702,
    "modified": 1746600804.9729257
  },
  {
    "name": "research_center.jpg",
    "size": 75007,
    "modified": 1746600804.9729257
  },
  {
    "name": "resources.txt",
    "size": 143,
    "modified": 1746600804.974937
  },
  {
    "name": "rfc2616.pdf",
    "size": 550558,
    "modified": 1746600804.9764624
```

Proses listing direktori dalam Thread Pool server dimulai ketika client mengirimkan HTTP GET request ke endpoint "/listdir". Request ini diterima oleh main server thread yang kemudian mendelegasikan pemrosesan ke salah satu worker thread yang tersedia dalam thread pool. Worker thread yang menerima tugas ini akan memanggil fungsi http_get() dalam HttpServer class yang mengidentifikasi bahwa request ditujukan untuk operasi listing direktori.

Fungsi list_directory() kemudian dieksekusi oleh worker thread tersebut. Fungsi ini pertama-tama memverifikasi keberadaan direktori target, dalam hal ini direktori "./folder". Jika direktori tidak ada, sistem akan otomatis membuatnya menggunakan os.makedirs(). Setelah direktori tersedia, fungsi melakukan iterasi melalui seluruh item dalam direktori menggunakan os.listdir().

Setiap item yang ditemukan akan diperiksa apakah merupakan file atau direktori menggunakan os.path.isfile(). Untuk setiap file yang ditemukan, sistem mengumpulkan metadata penting seperti nama file, ukuran file menggunakan os.path.getsize(), dan waktu modifikasi terakhir menggunakan os.path.getmtime(). Seluruh informasi ini dikompilasi ke dalam struktur data JSON yang kemudian dikirim kembali ke client sebagai response dengan content-type application/json.

Worker thread menyelesaikan tugasnya dengan mengirim response melalui socket connection yang telah dibuka, kemudian menutup koneksi dan kembali ke thread pool untuk menunggu tugas berikutnya.

c. Screenshot Mengupload sebuah file

Muhammad Nabil Fadhil
5025221200

```
(base) jovyan@7faed8a73447:~/work$ python client.py
Auto-detecting active server...
✓ Thread Pool Server detected on 172.16.16.101:8885

=== HTTP Client Operations (Thread Pool Server: mesin1:8885) ===
1. List Directory
2. Upload File
3. Delete File
4. Test Connection
5. Re-detect Server
6. Exit
==============================
Select operation (1-6): 2
Enter filename to upload: Upload2.txt
Enter file content: Upload
Uploading file 'Upload2.txt'...
WARNING:root:connecting to ('172.16.16.101', 8885)
WARNING:root:sending message
WARNING:root:POST /upload/Upload2.txt HTTP/1.1
Host: localhost
Content-Length: 6

Upload

WARNING:root:data received from server:
Server response:
HTTP/1.0 200 OK
Date: Sat Jul  5 13:15:02 2025
Connection: close
Server: myserver/1.0
Content-Length: 38

File Upload2.txt uploaded successfully



=== HTTP Client Operations (Thread Pool Server: mesin1:8885) ===
1. List Directory
2. Upload File
3. Delete File
4. Test Connection
5. Re-detect Server
6. Exit
==============================
Select operation (1-6):
```

Proses upload file dalam Thread Pool server diawali ketika client mengirimkan HTTP POST request ke endpoint "/upload/{filename}" dengan content yang ingin diupload disertakan dalam request body. Main server thread menerima request ini dan segera mendelegasikannya ke worker thread yang tersedia dalam pool.

Worker thread yang menerima tugas menggunakan buffer berukuran 1024 bytes untuk membaca data yang dikirim client. Proses pembacaan dilakukan secara iteratif hingga seluruh data request termasuk body content berhasil dibaca. Setelah data lengkap diterima, worker thread memanggil fungsi proses() dalam HttpServer class yang mengidentifikasi bahwa ini adalah POST request untuk upload file.

Fungsi http_post() kemudian dipanggil dengan parameter object_address yang dimulai dengan "/upload/". Sistem mengekstrak nama file dari URL path dengan menghilangkan prefix "/upload/" dan melakukan URL decoding menggunakan urllib.parse.unquote() untuk menangani karakter khusus dalam nama file.

Fungsi upload_file() dieksekusi selanjutnya, yang pertama-tama memverifikasi keberadaan direktori target "./folder". Jika direktori tidak ada, sistem otomatis membuatnya. File kemudian dibuat atau di-overwrite di lokasi yang ditentukan menggunakan operasi write dalam mode teks. Content file yang diterima dari client body akan ditulis ke file setelah dilakukan stripping untuk menghilangkan whitespace yang tidak perlu.

Setelah proses penulisan selesai, worker thread mengirim response sukses ke client dengan status 200 OK dan pesan konfirmasi yang menyatakan bahwa file telah berhasil diupload. Worker thread kemudian menutup koneksi dan kembali ke pool untuk menunggu request berikutnya.

d. Screenshot Menghapus file

Muhammad Nabil Fadhil
5025221200

```
(base) jovyan@7faed8a73447:~/work$ python client.py
Auto-detecting active server...
✓ Thread Pool Server detected on 172.16.16.101:8885

=== HTTP Client Operations (Thread Pool Server: mesin1:8885) ===
1. List Directory
2. Upload File
3. Delete File
4. Test Connection
5. Re-detect Server
6. Exit
==============================
Select operation (1-6): 2
Enter filename to upload: Upload2.txt
Enter file content: Uploat
Uploading file 'Upload2.txt'...
WARNING:root:connecting to ('172.16.16.101', 8885)
WARNING:root:sending message
WARNING:root:POST /upload/Upload2.txt HTTP/1.1
Host: localhost
Content-Length: 6

Uploat

WARNING:root:data received from server:
Server response:
HTTP/1.0 200 OK
Date: Sat Jul  5 13:15:02 2025
Connection: close
Server: myserver/1.0
Content-Length: 38

File Upload2.txt uploaded successfully



=== HTTP Client Operations (Thread Pool Server: mesin1:8885) ===
1. List Directory
2. Upload File
3. Delete File
4. Test Connection
5. Re-detect Server
6. Exit
==============================
Select operation (1-6):
```

Operasi penghapusan file dalam Thread Pool server dimulai ketika client mengirimkan HTTP DELETE request ke endpoint "/delete/{filename}". Request ini diterima oleh main server thread dan langsung didelegasikan ke worker thread yang tersedia dalam thread pool.

Worker thread yang menerima tugas memanggil fungsi proses() dalam HttpServer class yang mengidentifikasi bahwa ini adalah DELETE request. Fungsi http_delete() kemudian dipanggil

dengan parameter object_address yang dimulai dengan "/delete/". Sistem mengekstrak nama file yang akan dihapus dari URL path dengan menghilangkan prefix "/delete/".

Fungsi delete_file() dieksekusi selanjutnya, yang pertama-tama melakukan URL decoding pada nama file menggunakan urllib.parse.unquote() untuk menangani karakter khusus. Sistem kemudian membangun path lengkap ke file yang akan dihapus dengan menggabungkan direktori base "./folder" dengan nama file yang telah di-decode.

Sebelum melakukan penghapusan, sistem memverifikasi keberadaan file menggunakan os.path.exists(). Jika file tidak ditemukan, sistem akan mengirim response 404 Not Found ke client. Jika file ditemukan, sistem melakukan penghapusan menggunakan os.remove() dan mengirim response sukses 200 OK dengan pesan konfirmasi.

Worker thread menyelesaikan tugasnya dengan mengirim response yang sesuai ke client, kemudian menutup koneksi dan kembali ke thread pool untuk menunggu request berikutnya.

2. Implementasi Process Pool HTTP Server dengan fitur Melihat daftar file pada satu direktori,Mengupload sebuah file dan Menghapus file
    a. Penjelasan Modifikasi

```
(base) jovyan@a34a93dcdc3f:~/work$ python server_process_pool_http.py
Process Pool HTTP Server running on port 8889
Request #1 from ('172.16.16.102', 42260)
Status - Running: 0, Done: 0, Pending: 0
Request #2 from ('172.16.16.102', 42264)
Submitting to worker: GET /listdir HTTP/1.1
Worker completed in 0.1204 seconds
Status - Running: 0, Done: 1, Pending: 0
Request #3 from ('172.16.16.102', 52546)
Submitting to worker: POST /upload/Upload1.txt HTTP/1.1
Worker completed in 0.0101 seconds
Status - Running: 0, Done: 1, Pending: 0
Request #4 from ('172.16.16.102', 37066)
Submitting to worker: DELETE /delete/Upload1.txt HTTP/1.1
Worker completed in 0.0099 seconds
Status - Running: 0, Done: 1, Pending: 0
Request #5 from ('172.16.16.102', 34886)
Submitting to worker: GET /listdir HTTP/1.1
Worker completed in 0.1138 seconds
Status - Running: 0, Done: 1, Pending: 0
```

Disini saya menambahkan, Process pool tidak lagi men-share socket object ke worker process, melainkan membaca data di main process dan mengirim string ke worker, Menambahkan timeout dan error handling yang lebih robust, Menambahkan timing dan monitoring untuk worker process, MengImplementasi pembersihan future objects yang sudah selesai

b. Screenshot Melihat daftar file pada satu direktori

Muhammad Nabil Fadhil
5025221200

```
=== HTTP Client Operations (Process Pool Server: mesin1:8889) ===
1. List Directory
2. Upload File
3. Delete File
4. Test Connection
5. Re-detect Server
6. Exit
==============================
Select operation (1-6): 1
Listing directory...
WARNING:root:connecting to ('172.16.16.101', 8889)
WARNING:root:sending message
WARNING:root:GET /listdir HTTP/1.1
Host: localhost


WARNING:root:data received from server:
Server response:
HTTP/1.0 200 OK
Date: Sat Jul  5 13:13:36 2025
Connection: close
Server: myserver/1.0
Content-Length: 799
Content-Type:application/json

[
  {
    "name": "donalbebek.jpg",
    "size": 10729,
    "modified": 1746600804.968016
  },
  {
    "name": "gege.txt",
    "size": 10,
    "modified": 1751719922.58461
  },
  {
    "name": "pokijan.jpg",
    "size": 15702,
    "modified": 1746600804.9729257
  },
  {
    "name": "research_center.jpg",
    "size": 75007,
    "modified": 1746600804.9729257
  },
  {
    "name": "resources.txt",
    "size": 143,
    "modified": 1746600804.974937
  },
  {
    "name": "rfc2616.pdf",
    "size": 550558,
    "modified": 1746600804.9764624
  },
  {
```

Proses listing direktori dalam Process Pool server dimulai ketika client mengirimkan HTTP GET request ke endpoint "/listdir". Main process menerima koneksi dari client dan membaca seluruh data request menggunakan buffer berukuran 1024 bytes. Proses pembacaan dilakukan secara iteratif hingga seluruh request data berhasil dibaca, yang diidentifikasi dengan keberadaan sequence "\r\n" di akhir request.

Setelah data request lengkap diterima, main process mem-parsing request untuk mengekstrak informasi dasar seperti method dan path untuk keperluan logging. Main process kemudian men-submit tugas ke ProcessPoolExecutor dengan memanggil fungsi ProcessTheClient() dan mengirim data request sebagai string parameter.

Worker process yang menerima tugas membuat instance HttpServer baru di dalam process tersebut, karena object tidak dapat di-share antar process. Worker process kemudian memanggil fungsi proses() untuk memproses request string yang diterima. Fungsi ini mengidentifikasi bahwa request adalah GET request untuk "/listdir" dan memanggil fungsi http_get().

Dalam worker process, fungsi list_directory() dieksekusi yang melakukan operasi yang sama seperti Thread Pool: verifikasi direktori, iterasi melalui file-file, pengumpulan metadata, dan kompilasi dalam format JSON. Perbedaannya adalah bahwa seluruh operasi ini terjadi di dalam process terpisah yang terisolasi dari main process.

Setelah pemrosesan selesai, worker process mengembalikan hasil dalam bentuk bytes ke main process melalui future object. Main process menunggu hasil dengan timeout 10 detik menggunakan p.result(timeout=10). Jika hasil berhasil diterima, main process mengirim response ke client melalui socket connection yang telah dibuka, kemudian menutup koneksi.

c. Screenshot Mengupload sebuah file

Muhammad Nabil Fadhil
5025221200

```
=== HTTP Client Operations (Process Pool Server: mesin1:8889) ===
1. List Directory
2. Upload File
3. Delete File
4. Test Connection
5. Re-detect Server
6. Exit
================================
Select operation (1-6): 2
Enter filename to upload: Upload1.txt
Enter file content: Upload
Uploading file 'Upload1.txt'...
WARNING:root:connecting to ('172.16.16.101', 8889)
WARNING:root:sending message
WARNING:root:POST /upload/Upload1.txt HTTP/1.1
Host: localhost
Content-Length: 6

Upload

WARNING:root:data received from server:
Server response:
HTTP/1.0 200 OK
Date: Sat Jul  5 13:12:11 2025
Connection: close
Server: myserver/1.0
Content-Length: 38

File Upload1.txt uploaded successfully



=== HTTP Client Operations (Process Pool Server: mesin1:8889) ===
1. List Directory
2. Upload File
3. Delete File
4. Test Connection
5. Re-detect Server
6. Exit
================================
Select operation (1-6):
```

Proses upload file dalam Process Pool server dimulai ketika client mengirimkan HTTP POST request ke endpoint "/upload/{filename}" dengan file content dalam request body. Main process menerima koneksi dan membaca seluruh request data termasuk body content menggunakan buffer berukuran 1024 bytes yang telah ditingkatkan untuk mendukung file upload.

Main process melakukan pembacaan data secara iteratif hingga seluruh request termasuk body content berhasil dibaca. Setelah data lengkap diterima, main process mengekstrak informasi request dan mem-submit tugas ke

ProcessPoolExecutor dengan mengirim seluruh data request sebagai string parameter ke fungsi ProcessTheClient().

Worker process yang menerima tugas membuat instance HttpServer baru di dalam process tersebut. Worker process kemudian memproses request string untuk mengidentifikasi bahwa ini adalah POST request untuk upload file. Fungsi http_post() dipanggil dengan parameter yang telah diekstrak dari request.

Dalam worker process, fungsi upload_file() dieksekusi yang melakukan operasi yang sama seperti Thread Pool: verifikasi direktori, URL decoding nama file, dan penulisan file content ke disk. Perbedaannya adalah bahwa seluruh operasi ini terjadi di dalam process terpisah yang terisolasi, memberikan keamanan dan stabilitas yang lebih baik.

Setelah file berhasil diupload, worker process mengembalikan response sukses ke main process melalui future object. Main process menunggu hasil dengan timeout dan mengirim response final ke client melalui socket connection, kemudian menutup koneksi.

d. Screenshot Menghapus file

Muhammad Nabil Fadhil
5025221200

```
=== HTTP Client Operations (Process Pool Server: mesin1:8889) ===
1. List Directory
2. Upload File
3. Delete File
4. Test Connection
5. Re-detect Server
6. Exit
==============================
Select operation (1-6): 3
Enter filename to delete: Upload1.txt
Deleting file 'Upload1.txt'...
WARNING:root:connecting to ('172.16.16.101', 8889)
WARNING:root:sending message
WARNING:root:DELETE /delete/Upload1.txt HTTP/1.1
Host: localhost


WARNING:root:data received from server:
Server response:
HTTP/1.0 200 OK
Date: Sat Jul  5 13:12:53 2025
Connection: close
Server: myserver/1.0
Content-Length: 37

File Upload1.txt deleted successfully



=== HTTP Client Operations (Process Pool Server: mesin1:8889) ===
1. List Directory
2. Upload File
3. Delete File
4. Test Connection
5. Re-detect Server
6. Exit
==============================
Select operation (1-6):
```

Operasi penghapusan file dalam Process Pool server dimulai ketika client mengirimkan HTTP DELETE request ke endpoint "/delete/{filename}". Main process menerima koneksi dan membaca seluruh request data dengan mekanisme yang sama seperti operasi lainnya.

Setelah data request lengkap diterima, main process mem-submit tugas ke ProcessPoolExecutor dengan mengirim data request sebagai string parameter ke worker process. Worker process yang menerima tugas membuat instance HttpServer baru dan memproses request untuk mengidentifikasi bahwa ini adalah DELETE request.

Dalam worker process, fungsi http_delete() dipanggil yang kemudian mengeksekusi fungsi delete_file(). Fungsi ini melakukan URL decoding nama file, verifikasi keberadaan file, dan penghapusan file jika ditemukan. Seluruh operasi ini terjadi di dalam process terpisah yang terisolasi dari main process.

Worker process mengembalikan response yang sesuai (sukses atau error) ke main process melalui future object. Main process menunggu hasil dengan timeout dan mengirim response final ke client, kemudian menutup koneksi.

3. Link Github.
   https://github.com/Papavero30/Progjar-Activity.git