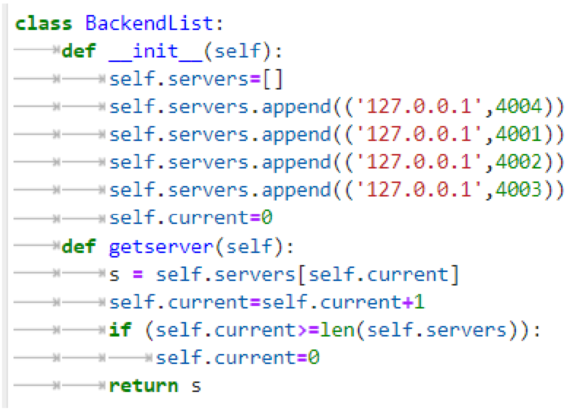
Ichlasul Hasanat

5025201091

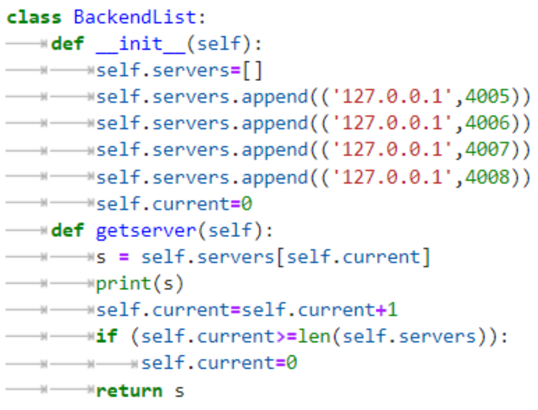
Tugas 5 Pemrograman Jaringan

- Implementasikan arsitektur load balancer
  Sebelum mengimplementasikan load balancer, disesuaikan dahulu list port yang ingin Dikoneksikan pada kedua file load balancer serta bash file untuk run server (karena run server hanya ada 1 yaitu untuk asynchronous, maka dibuat satu run server lagi yaitu ‘runserverpool.sh’ untuk server pool). Untuk port yang digunakan adalah sebagai berikut:
  - Port 8887 : Mode Asynchronous (append port 4001-4004)
  - Port 8000 : Mode Server Pool (append port 4005-4008)
  - Port 44444 : Load balancer mode Server Pool
  - Port 55555 : Load balancer mode Asynchronous

- Tampilan class “BackendList” pada file “lb_async”(kiri) dan “lb_process”(kanan):

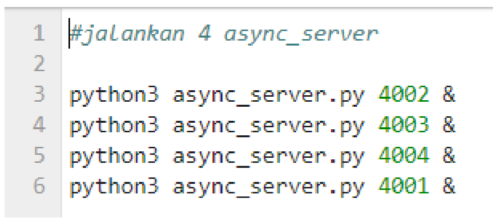


```python
class BackendList:
	def __init__(self):
		self.servers=[]
		self.servers.append(('127.0.0.1',4004))
		self.servers.append(('127.0.0.1',4001))
		self.servers.append(('127.0.0.1',4002))
		self.servers.append(('127.0.0.1',4003))
		self.current=0
	def getserver(self):
		s = self.servers[self.current]
		self.current=self.current+1
		if (self.current>=len(self.servers)):
			self.current=0
		return s
```

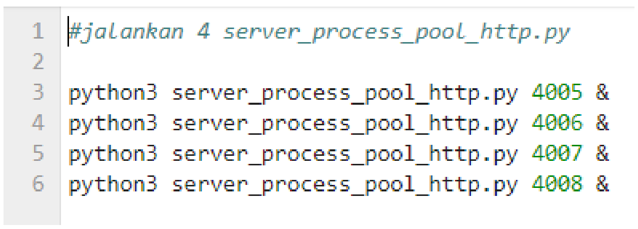


```python
class BackendList:
	def __init__(self):
		self.servers=[]
		self.servers.append(('127.0.0.1',4005))
		self.servers.append(('127.0.0.1',4006))
		self.servers.append(('127.0.0.1',4007))
		self.servers.append(('127.0.0.1',4008))
		self.current=0
	def getserver(self):
		s = self.servers[self.current]
		print(s)
		self.current=self.current+1
		if (self.current>=len(self.servers)):
			self.current=0
		return s
```

- Tampilan isi “runserverasync.sh”(kiri) dan “runserverpool.sh”(kanan):

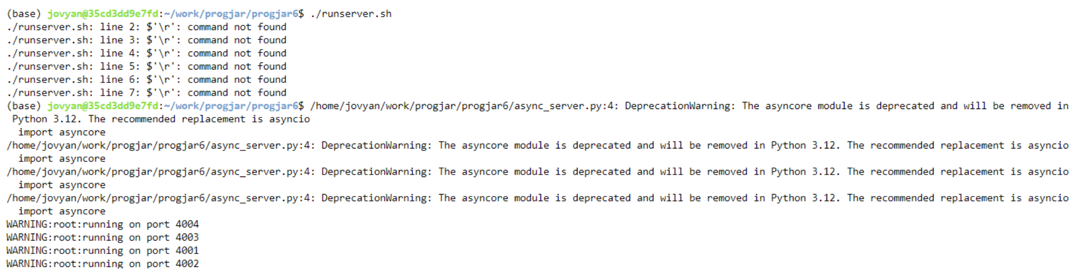


```bash
#jalankan 4 async_server

python3 async_server.py 4002 &
python3 async_server.py 4003 &
python3 async_server.py 4004 &
python3 async_server.py 4001 &
```



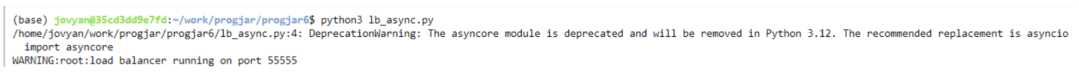
```bash
#jalankan 4 server_process_pool_http.py

python3 server_process_pool_http.py 4005 &
python3 server_process_pool_http.py 4006 &
python3 server_process_pool_http.py 4007 &
python3 server_process_pool_http.py 4008 &
```

- Mode Asynchronous
  - Jalankan “runserverasync.sh” di terminal pertama untuk menjalankan “async_server.py” pada masing-masing port yang sudah di-assign.



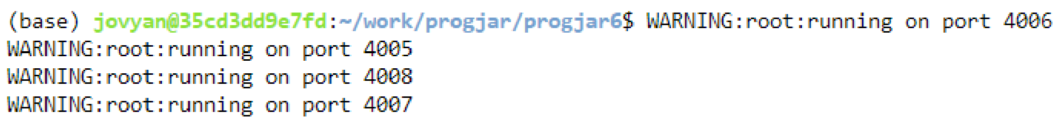
```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ ./runserver.sh
./runserver.sh: line 2: $'\r': command not found
./runserver.sh: line 3: $'\r': command not found
./runserver.sh: line 4: $'\r': command not found
./runserver.sh: line 5: $'\r': command not found
./runserver.sh: line 6: $'\r': command not found
./runserver.sh: line 7: $'\r': command not found
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ /home/jovyan/work/progjar/progjar6/async_server.py:4: DeprecationWarning: The asyncore module is deprecated and will be removed in
 Python 3.12. The recommended replacement is asyncio
  import asyncore
/home/jovyan/work/progjar/progjar6/async_server.py:4: DeprecationWarning: The asyncore module is deprecated and will be removed in Python 3.12. The recommended replacement is asyncio
  import asyncore
/home/jovyan/work/progjar/progjar6/async_server.py:4: DeprecationWarning: The asyncore module is deprecated and will be removed in Python 3.12. The recommended replacement is asyncio
  import asyncore
/home/jovyan/work/progjar/progjar6/async_server.py:4: DeprecationWarning: The asyncore module is deprecated and will be removed in Python 3.12. The recommended replacement is asyncio
  import asyncore
WARNING:root:running on port 4004
WARNING:root:running on port 4003
WARNING:root:running on port 4001
WARNING:root:running on port 4002
```

- Setelah menyalakan Asynchronous server, nyalakan load balancer pada file “lb_async.py” pada terminal yang berbeda.



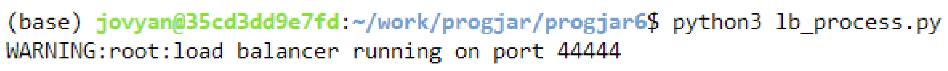
```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ python3 lb_async.py
/home/jovyan/work/progjar/progjar6/lb_async.py:4: DeprecationWarning: The asyncore module is deprecated and will be removed in Python 3.12. The recommended replacement is asyncio
  import asyncore
WARNING:root:load balancer running on port 55555
```

- Mode Server Pool
  - Jalankan “runserverpool.sh” di terminal pertama untuk menjalankan “server_process_pool_http.py” pada masing-masing port yang sudah di-assign.



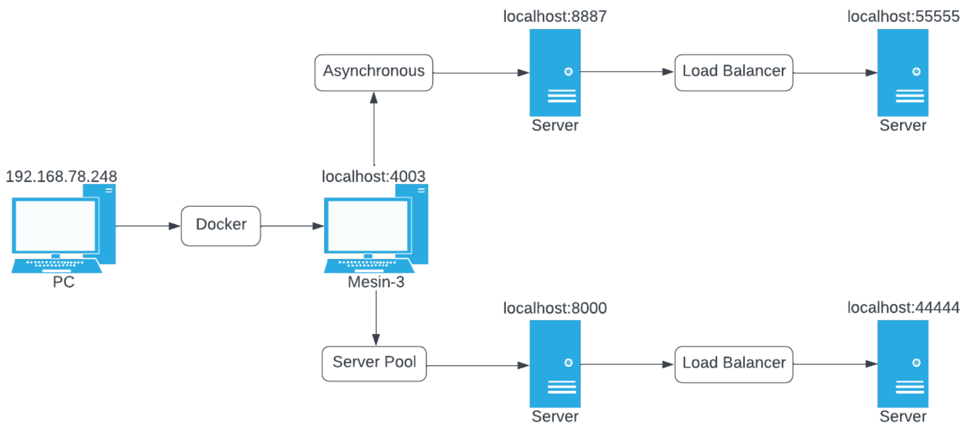
```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ WARNING:root:running on port 4006
WARNING:root:running on port 4005
WARNING:root:running on port 4008
WARNING:root:running on port 4007
```

  - Setelah menyalakan Server pool, nyalakan load balancer pada file “lb_process.py” pada terminal yang berbeda.



```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ python3 lb_process.py
WARNING:root:load balancer running on port 44444
```

- Buatlah perbandingan kinerja web server
  - Buatlah gambar dari arsitektur percobaan
    - Arsitektur percobaan

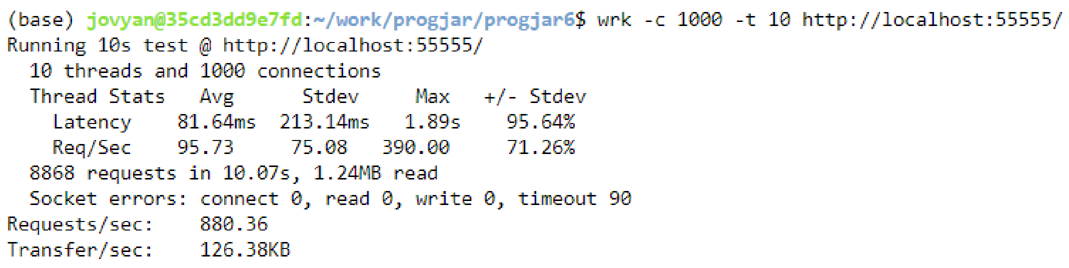


- wrk dengan jumlah request/koneksi 1000, dengan parameter concurrency 10,50,100,150,200

  Buka terminal baru lalu install library ‘wrk’ terlebih dahulu menggunakan ‘sudo apt-get install wrk’ lalu jalankan concurrency menggunakan format berikut:

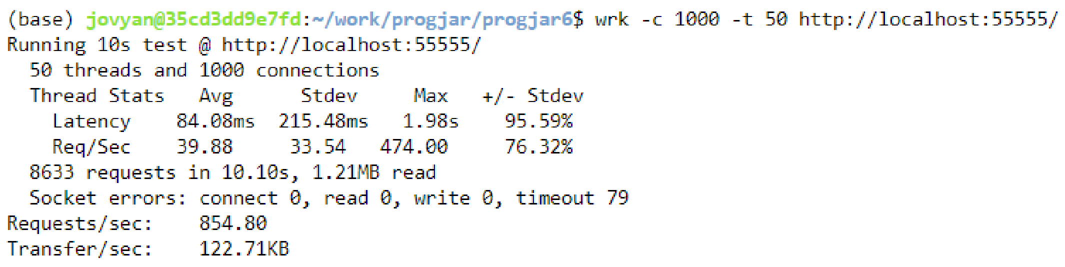
  wrk -c 1000 -t {n} http://url

- **<u>Asynchronous load balancer</u>**
    - Concurrency 10
      wrk -c 1000 -t 10 http://localhost:55555/



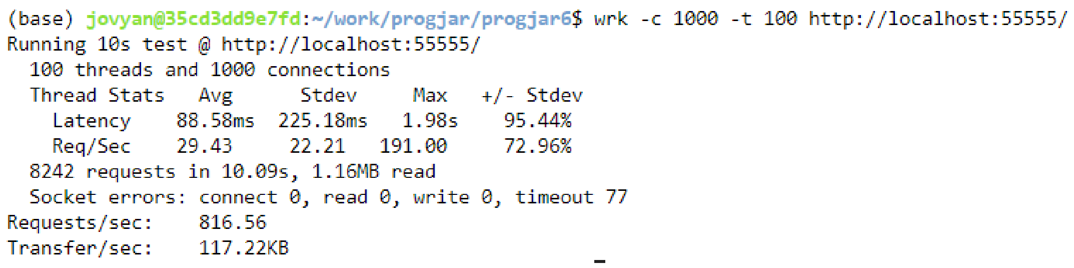
```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ wrk -c 1000 -t 10 http://localhost:55555/
Running 10s test @ http://localhost:55555/
  10 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency    81.64ms  213.14ms   1.89s    95.64%
    Req/Sec    95.73     75.08   390.00     71.26%
  8868 requests in 10.07s, 1.24MB read
  Socket errors: connect 0, read 0, write 0, timeout 90
Requests/sec:    880.36
Transfer/sec:    126.38KB
```

    - Concurrency 50
      wrk -c 1000 -t 50 http://localhost:55555/



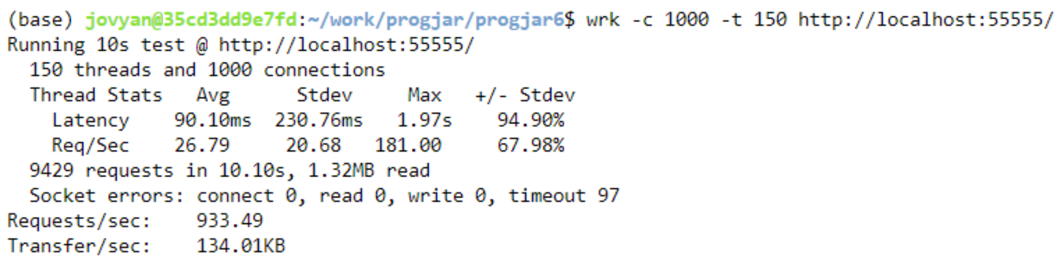
```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ wrk -c 1000 -t 50 http://localhost:55555/
Running 10s test @ http://localhost:55555/
  50 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency    84.08ms  215.48ms   1.98s    95.59%
    Req/Sec    39.88     33.54   474.00     76.32%
  8633 requests in 10.10s, 1.21MB read
  Socket errors: connect 0, read 0, write 0, timeout 79
Requests/sec:    854.80
Transfer/sec:    122.71KB
```

    - Concurrency 100
      wrk -c 1000 -t 100 http://localhost:55555/



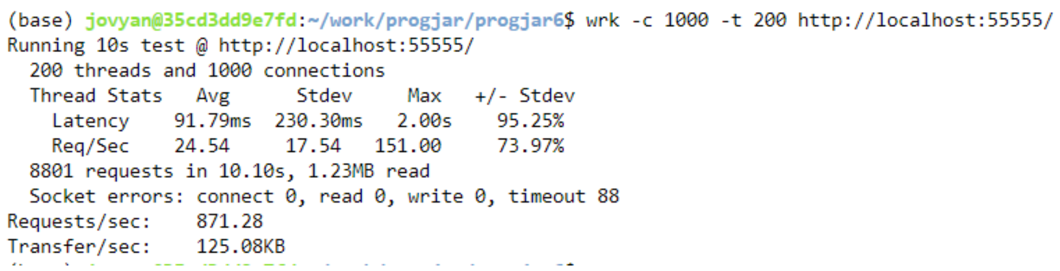
```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ wrk -c 1000 -t 100 http://localhost:55555/
Running 10s test @ http://localhost:55555/
  100 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency    88.58ms  225.18ms   1.98s    95.44%
    Req/Sec    29.43     22.21   191.00     72.96%
  8242 requests in 10.09s, 1.16MB read
  Socket errors: connect 0, read 0, write 0, timeout 77
Requests/sec:    816.56
Transfer/sec:    117.22KB
```

    - Concurrency 150
      wrk -c 1000 -t 150 http://localhost:55555/



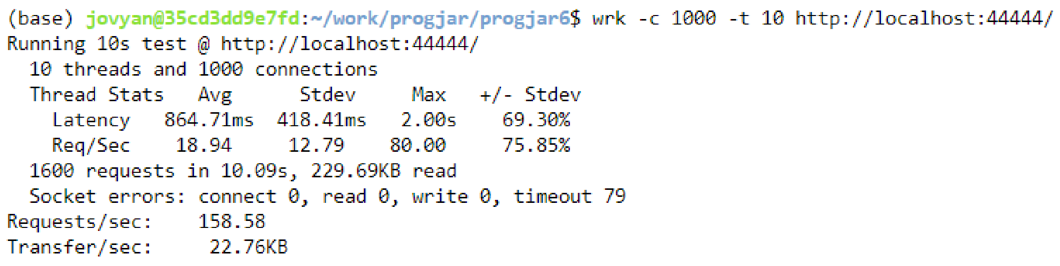
```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ wrk -c 1000 -t 150 http://localhost:55555/
Running 10s test @ http://localhost:55555/
  150 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency    90.10ms  230.76ms   1.97s    94.90%
    Req/Sec    26.79     20.68   181.00     67.98%
  9429 requests in 10.10s, 1.32MB read
  Socket errors: connect 0, read 0, write 0, timeout 97
Requests/sec:    933.49
Transfer/sec:    134.01KB
```

- Concurrency 200
  wrk -c 1000 -t 200 http://localhost:55555/



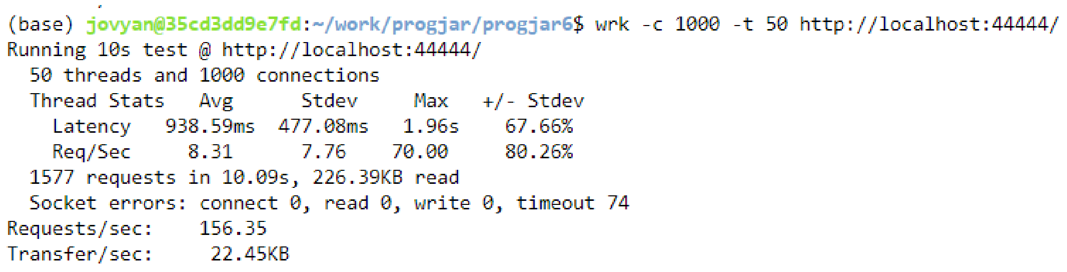
```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ wrk -c 1000 -t 200 http://localhost:55555/
Running 10s test @ http://localhost:55555/
  200 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency    91.79ms  230.30ms   2.00s    95.25%
    Req/Sec    24.54     17.54   151.00     73.97%
  8801 requests in 10.10s, 1.23MB read
  Socket errors: connect 0, read 0, write 0, timeout 88
Requests/sec:    871.28
Transfer/sec:    125.08KB
```

- **Server pool load balancer**
  - Concurrency 10
    wrk -c 1000 -t 10 http://localhost:44444/



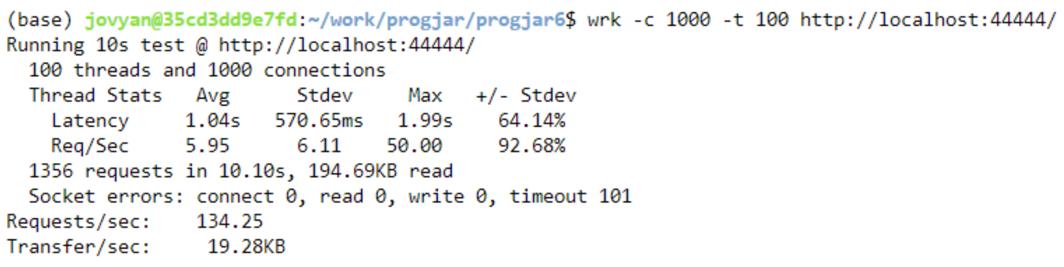
```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ wrk -c 1000 -t 10 http://localhost:44444/
Running 10s test @ http://localhost:44444/
  10 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency   864.71ms  418.41ms   2.00s    69.30%
    Req/Sec    18.94     12.79    80.00     75.85%
  1600 requests in 10.09s, 229.69KB read
  Socket errors: connect 0, read 0, write 0, timeout 79
Requests/sec:    158.58
Transfer/sec:     22.76KB
```

  - Concurrency 50
    wrk -c 1000 -t 50 http://localhost:44444/



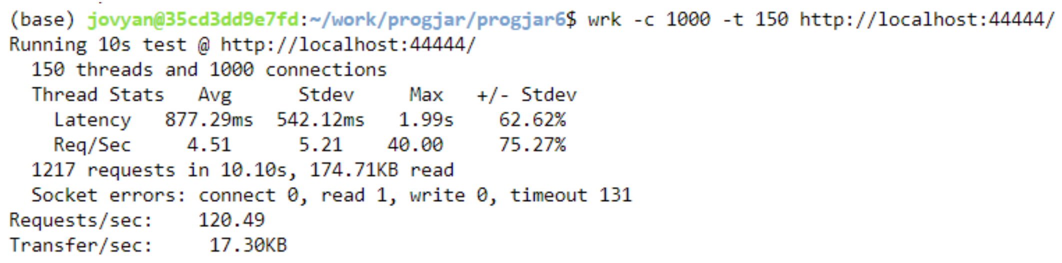
```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ wrk -c 1000 -t 50 http://localhost:44444/
Running 10s test @ http://localhost:44444/
  50 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency   938.59ms  477.08ms   1.96s    67.66%
    Req/Sec     8.31      7.76    70.00     80.26%
  1577 requests in 10.09s, 226.39KB read
  Socket errors: connect 0, read 0, write 0, timeout 74
Requests/sec:    156.35
Transfer/sec:     22.45KB
```

  - Concurrency 100
    wrk -c 1000 -t 100 http://localhost:44444/



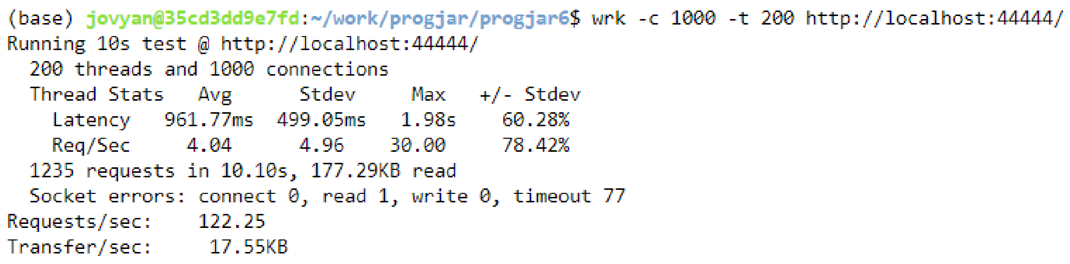
```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ wrk -c 1000 -t 100 http://localhost:44444/
Running 10s test @ http://localhost:44444/
  100 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency     1.04s   570.65ms   1.99s    64.14%
    Req/Sec     5.95      6.11    50.00     92.68%
  1356 requests in 10.10s, 194.69KB read
  Socket errors: connect 0, read 0, write 0, timeout 101
Requests/sec:    134.25
Transfer/sec:     19.28KB
```

- Concurrency 150
  wrk -c 1000 -t 150 http://localhost:44444/



```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ wrk -c 1000 -t 150 http://localhost:44444/
Running 10s test @ http://localhost:44444/
  150 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency   877.29ms  542.12ms   1.99s    62.62%
    Req/Sec     4.51      5.21    40.00     75.27%
  1217 requests in 10.10s, 174.71KB read
  Socket errors: connect 0, read 1, write 0, timeout 131
Requests/sec:    120.49
Transfer/sec:     17.30KB
```

- Concurrency 200
  wrk -c 1000 -t 200 http://localhost:44444/



```
(base) jovyan@35cd3dd9e7fd:~/work/progjar/progjar6$ wrk -c 1000 -t 200 http://localhost:44444/
Running 10s test @ http://localhost:44444/
  200 threads and 1000 connections
  Thread Stats   Avg      Stdev     Max   +/- Stdev
    Latency   961.77ms  499.05ms   1.98s    60.28%
    Req/Sec     4.04      4.96    30.00     78.42%
  1235 requests in 10.10s, 177.29KB read
  Socket errors: connect 0, read 1, write 0, timeout 77
Requests/sec:    122.25
Transfer/sec:     17.55KB
```

- Laporkan kinerja dalam hal
  - Failed requests, request per second, waiting
    - **<u>Asynchronous load balancer</u>**
      - Concurrency 10
        Failed requests: 90
        Requests/sec: 880.36/sec
        Waiting: 81.64 ms
      - Concurrency 50
        Failed requests: 79
        Requests/sec: 854.80/sec
        Waiting: 84.08 ms
      - Concurrency 100
        Failed requests: 77
        Requests/sec: 816.56/sec
        Waiting: 88.58 ms
      - Concurrency 150
        Failed requests: 97
        Requests/sec: 933.49/sec
        Waiting: 90.10 ms

        - Concurrency 200
          Failed requests: 88
          Requests/sec: 871.28/sec
          Waiting: 91.79 ms

- **<u>Server pool load balancer</u>**

    - Concurrency 10
      Failed requests: 79
      Requests/sec: 158.58/sec
      Waiting: 864.71 ms

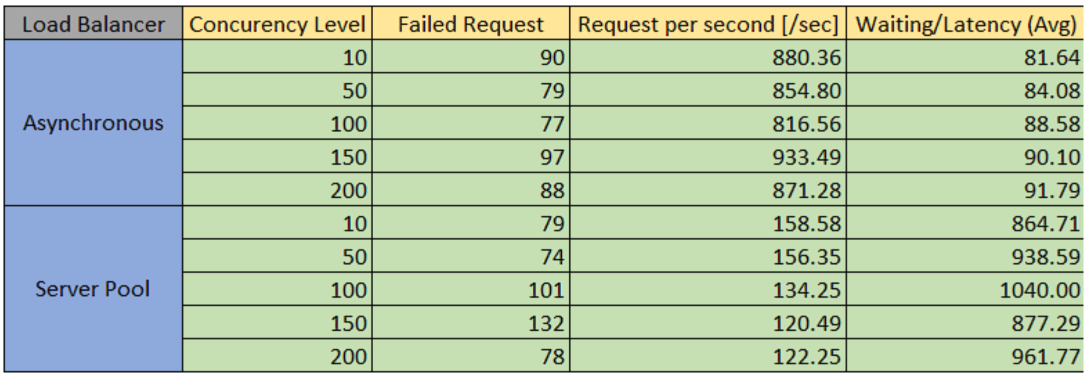
    - Concurrency 50
      Failed requests: 74
      Requests/sec: 156.35/sec
      Waiting: 938.59 ms

    - Concurrency 100
      Failed requests: 101
      Requests/sec: 134.25/sec
      Waiting: 1040 ms

    - Concurrency 150
      Failed requests: 132
      Requests/sec: 120.49/sec
      Waiting: 877.29 ms

    - Concurrency 200
      Failed requests: 78
      Requests/sec: 122.25/sec
      Waiting: 961.77 ms

- **<u>Tabel perbandingan hasil benchmarking</u>**



Load Balancer	Concurency Level	Failed Request	Request per second [/sec]	Waiting/Latency (Avg)
Asynchronous	10	90	880.36	81.64
	50	79	854.80	84.08
	100	77	816.56	88.58
	150	97	933.49	90.10
	200	88	871.28	91.79
Server Pool	10	79	158.58	864.71
	50	74	156.35	938.59
	100	101	134.25	1040.00
	150	132	120.49	877.29
	200	78	122.25	961.77

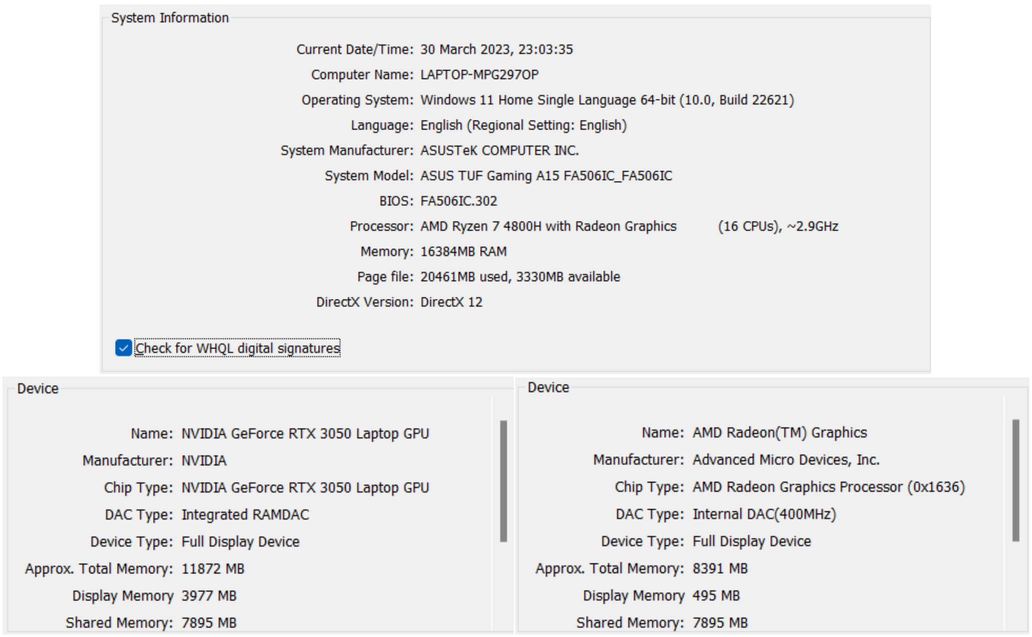
- **<u>Kesimpulan</u>**

  Berdasarkan hasil benchmarking di atas, dapat dilhat bahwa asynchronous load balancer menjalankan request per second jauh lebih banyak dan mean latency yang jauh lebih rendah dibanding menggunakan server pool load balancer. Walaupun server pool dalam beberapa concurrency tertentu mampu menghasilkan failed request yang cukup minim, jika direratakan dan dipertimbangkan dengan request per second dan latency, asynchronous load balancer masih tetap lebih unggul dibandingkan dengan server pool.

- Laporan disubmit dalam bentuk
  - 1 dokumen PDF, maks 10 halaman
  - Isikan alamat repository yang berkaitan dengan tugas diatas

    Link Repository: https://github.com/Ichlas02/Tugas5_Progjar_A/tree/main

  - Spek komputer yang digunakan untuk menjalankan server dan melakukan testing, berserta gambar arsitektur percobaan



System Information

Current Date/Time:	30 March 2023, 23:03:35
Computer Name:	LAPTOP-MPG297OP
Operating System:	Windows 11 Home Single Language 64-bit (10.0, Build 22621)
Language:	English (Regional Setting: English)
System Manufacturer:	ASUSTeK COMPUTER INC.
System Model:	ASUS TUF Gaming A15 FA506IC_FA506IC
BIOS:	FA506IC.302
Processor:	AMD Ryzen 7 4800H with Radeon Graphics (16 CPUs), ~2.9GHz
Memory:	16384MB RAM
Page file:	20461MB used, 3330MB available
DirectX Version:	DirectX 12

[✓] Check for WHQL digital signatures

Device

Name:	NVIDIA GeForce RTX 3050 Laptop GPU
Manufacturer:	NVIDIA
Chip Type:	NVIDIA GeForce RTX 3050 Laptop GPU
DAC Type:	Integrated RAMDAC
Device Type:	Full Display Device
Approx. Total Memory:	11872 MB
Display Memory	3977 MB
Shared Memory:	7895 MB

Device

Name:	AMD Radeon(TM) Graphics
Manufacturer:	Advanced Micro Devices, Inc.
Chip Type:	AMD Radeon Graphics Processor (0x1636)
DAC Type:	Internal DAC(400MHz)
Device Type:	Full Display Device
Approx. Total Memory:	8391 MB
Display Memory	495 MB
Shared Memory:	7895 MB

*Arsitektur percobaan sudah dicantumkan di atas.