Techno.COM, Vol. 24, No. 4, November 2025: 1092-1101

Implementasi Algoritma *Round Robin* dalam Sistem *Multi-agent* dan *Multi-client* untuk *Load balancing* Dinamis pada Jaringan Lokal

Implementation of the Round Robin Algorithm in a Multi-agent and Multi-client System for Dynamic Load Balancing on a Local Network

Wiwi Nopiana* [1], **Wawan Firgiwan**[2], **Muh. Fuad Mansyur**[3],
Informatika, Fakultas Teknik, Universitas Sulawesi Barat, Kab. Majene, Sulawesi Barat
E-mail : novianawiwi0@gmail.com[1], wawanfirgiawan@unsulbar.ac.id*[2], *muh.fuadm@unsulbar.ac.id*[3]
Received 14 May 2025; Revised 15 September 2025; Accepted 30 September 2025

Abstrak - *Load balancing* merupakan mekanisme penting dalam sistem layanan web untuk menjamin pemerataan beban kerja dan menjaga kestabilan performa layanan. Penelitian ini mengimplementasikan algoritma *Round Robin* dalam arsitektur sistem *multi-agent* dan *multi-client* yang dijalankan pada *local area networking* (LAN). Sistem dirancang menggunakan tiga komputer, di mana satu komputer berperan sebagai *agent controller* yang menjalankan logika *Round Robin*, dan dua komputer lainnya sebagai *server backend*. Beberapa *client* dalam jaringan mengirimkan permintaan secara simultan ke *controller*, yang kemudian secara bergiliran mendistribusikan permintaan tersebut ke *server* menggunakan konfigurasi *load balancing* berbasis NGINX. Pengujian dilakukan dalam tiga skenario beban, yaitu 50, 100, dan 200 permintaan. Hasil pengujian menunjukkan bahwa sistem mampu mendistribusikan permintaan secara merata antara dua *server backend*, serta menghasilkan waktu respons yang stabil pada skenario beban ringan hingga sedang. Kinerja sistem tetap berada dalam batas wajar meskipun jumlah permintaan meningkat. Sistem ini menunjukkan karakteristik modular, ringan, dan mudah diimplementasikan dalam lingkungan terbatas, serta memiliki potensi untuk dikembangkan lebih lanjut dengan integrasi algoritma adaptif dan sistem pemantauan otomatis.

Kata kunci - *Load balancing, Round Robin,* sistem *multi-agent, multi-client, jaringan local*

Abstract *— Load balancing is a crucial mechanism in web service systems to ensure equitable distribution of workloads and maintain service performance stability. This study implements the Round Robin algorithm within a multi-agent and multi-client system architecture operating over a local area network (LAN). The system is designed using three computers: one acts as the agent controller executing the Round Robin logic, while the other two function as backend servers. Multiple clients on the network simultaneously send requests to the controller, which then distributes them in a rotating manner to the servers using an NGINX-based load balancing configuration. Testing was conducted under three load scenarios: 50, 100, and 200 requests. The results demonstrate that the system effectively distributes requests evenly between the two backend servers and achieves stable response times under light to moderate load conditions. The system performance remains within acceptable limits even as the number of requests increases. This system exhibits a modular, lightweight, and easily deployable architecture suitable for constrained environments, with potential for further development through integration with adaptive algorithms and automated monitoring systems.*

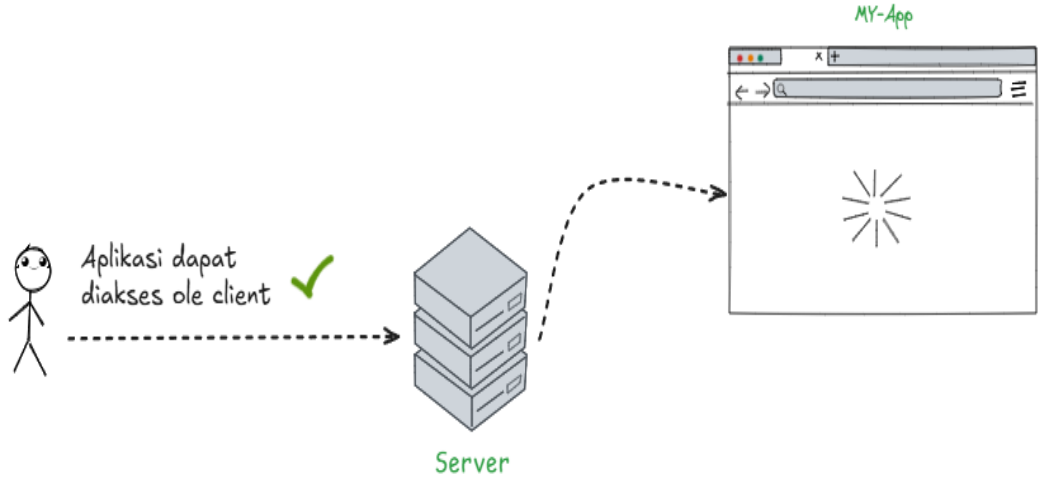
Keywords *— Load balancing, Round Robin, multi-agent system, multi-client, local network.*

e-ISSN : 2356-2579 | p-ISSN : 1412-2693, DOI : 10.62411/tc.v24i4.12777

1. PENDAHULUAN

Jaringan komputer telah menjadi komponen penting dalam kehidupan modern, dengan salah satu model yang paling banyak digunakan adalah model jaringan *client-server*. Model ini menjadi dasar dari berbagai aplikasi, mulai dari sistem email, layanan berbasis data, hingga platform komputasi awan *(cloud computing).* Seiring dengan pesatnya perkembangan teknologi informasi, terjadi lonjakan signifikan dalam jumlah layanan digital berbasis web seperti *e-commerce,* sistem manajemen informasi, platform media sosial, dan layanan cloud. Peningkatan jumlah pengguna dan akses yang terjadi secara simultan menyebabkan beban kerja pada *server* menjadi tidak merata, yang pada akhirnya dapat menurunkan kinerja sistem secara keseluruhan.

Era digital saat ini, layanan berbasis web memainkan peran yang sangat penting dalam mendukung berbagai aktivitas manusia, baik dalam bidang pendidikan, bisnis, pemerintahan, maupun kehidupan sosial. Permintaan terhadap sistem yang cepat, stabil, dan selalu tersedia meningkat seiring dengan bertambahnya jumlah pengguna yang mengakses layanan secara bersamaan. Masalah utama yang sering dihadapi oleh layanan web berskala kecil hingga menengah adalah tidak seimbangnya beban kerja antar *server*, yang dapat menyebabkan lambatnya waktu respons, gangguan layanan, bahkan kegagalan sistem secara keseluruhan.



Gambar 1. Ilustrasi sederhana konsep akses satu *server* dan beberapa client

Salah satu solusi teknis yang umum digunakan untuk mengatasi hal tersebut adalah *load balancing*, yaitu proses pendistribusian permintaan dari klien ke beberapa *server* secara merata untuk mengoptimalkan kinerja sistem dan menjaga stabilitas layanan. *Load balancing* juga memungkinkan sistem menjadi lebih skalabel dan memiliki toleransi terhadap kesalahan. Menurut Arora dan Sood [1], *load balancing* berperan krusial dalam arsitektur sistem terdistribusi karena dapat menghindari bottleneck dan meningkatkan efisiensi. Buku Buyya et al. [2] menyebutkan bahwa *load balancing* adalah fondasi dalam arsitektur cloud, tetapi prinsip-prinsip dasarnya juga relevan dalam sistem lokal dan edge computing.

Salah satu algoritma *load balancing* paling populer adalah *Round Robin*, karena sifatnya yang sederhana dan deterministik. *Round Robin* bekerja dengan cara mengirimkan setiap permintaan pengguna secara bergiliran ke *server* yang tersedia, tanpa mempertimbangkan beban atau status *server* tersebut. Meskipun tidak seadaptif algoritma dinamis seperti *Least Connection*, namun *Round Robin* tetap efektif dan efisien pada sistem dengan jumlah *server* terbatas dan beban yang relatif seimbang [3]. Penelitian oleh Wijaya et al. [4] menunjukkan bahwa implementasi *Round Robin* dalam sistem lokal memberikan distribusi beban yang cukup seimbang dengan waktu respons yang stabil. Hasil serupa ditemukan oleh Utami dan Haryanto [5] yang menyatakan bahwa *Round Robin* lebih unggul dari strategi acak dalam konteks *server* berbasis HAProxy.

e-ISSN : 2356-2579 | p-ISSN : 1412-2693, DOI : 10.62411/tc.v24i4.12777

Seiring berkembangnya arsitektur sistem, muncul pendekatan multi-agen sebagai metode baru dalam pengelolaan layanan secara terdistribusi. Dalam sistem multi-agen, setiap komponen (disebut agen) bekerja secara otonom namun tetap berkoordinasi satu sama lain. Menurut Arifin dan Nugroho [6], pendekatan multi-agen memberikan fleksibilitas dan modularitas tinggi dalam pengelolaan jaringan. Liu et al. [7] memperkenalkan model multi-agen kooperatif untuk distribusi beban dinamis pada sistem web, sementara Alfian et al. [8] membuktikan bahwa arsitektur ini dapat meningkatkan efisiensi dalam sistem IoT melalui komunikasi antar agen.

Namun, sebagian besar penelitian terdahulu masih fokus pada simulasi atau implementasi berbasis cloud. Studi oleh Rahmika et al. [9] mengembangkan mekanisme *load balancing* berbasis *Least Connection* dan multi-agen pada sistem web *server*, namun seluruh sistem diuji pada infrastruktur cloud. Di sisi lain, Singh et al. [10] melalui penelitian komprehensifnya menegaskan pentingnya kesesuaian strategi *load balancing* dengan lingkungan penerapannya. Dalam hal ini, belum banyak penelitian yang secara khusus mengkaji efektivitas algoritma *Round Robin* dalam skenario jaringan lokal dengan topologi multi-agen dan *multi-client*.

Penelitian ini menawarkan pendekatan yang berbeda dengan mengembangkan sistem *load balancing* berbasis algoritma *Round Robin*, yang diimplementasikan dalam arsitektur multi-agen dan *multi-client* pada jaringan lokal (LAN). Sistem terdiri dari tiga komputer lokal: satu sebagai agent controller yang menjalankan logika *Round Robin*, dan dua sebagai agent *server* yang menerima permintaan. Beberapa klien dalam jaringan mengakses layanan secara bersamaan, menciptakan simulasi kondisi nyata dalam lingkungan terbatas seperti laboratorium, sekolah, atau usaha kecil menengah.

Fokus dari penelitian ini terletak pada penerapan algoritma sederhana namun fungsional (*Round Robin*) dalam sistem terdistribusi nyata berbasis *multi-agen* dan *multi-client* pada LAN. Sementara studi sebelumnya banyak berfokus pada pendekatan berbasis cloud atau simulasi virtual, penelitian ini memfokuskan pada implementasi nyata, ringan, dan aplikatif yang mudah direplikasi di lingkungan dengan keterbatasan sumber daya. Selain itu, penelitian ini juga memberikan kontribusi dalam penyusunan dasar sistem *load balancing* edukatif atau prototipe sistem internal untuk layanan web sederhana tanpa ketergantungan terhadap infrastruktur *cloud.*

Dengan demikian, penelitian ini tidak hanya memberikan solusi teknis untuk distribusi beban, tetapi juga menawarkan model arsitektur praktis yang fleksibel dan relevan untuk kebutuhan riil, terutama pada sistem lokal berskala kecil hingga menengah yang membutuhkan efisiensi, kecepatan, dan kestabilan dalam melayani permintaan secara paralel.

2. METODE PENELITIAN

Penelitian ini menggunakan pendekatan eksperimen sistem untuk menguji performa algoritma *Round Robin* dalam skenario nyata berbasis jaringan lokal dengan konfigurasi sistem *multi-agent* dan *multi-client*. Dalam bab ini dijelaskan tahapan desain sistem, arsitektur perangkat lunak dan perangkat keras yang digunakan, alur kerja sistem, serta parameter dan skenario pengujian yang digunakan sebagai dasar evaluasi.

2.1. Desain Penelitian

Pada bagian ini dijelaskan rancangan umum sistem yang dibangun, yang terdiri dari beberapa agen terdistribusi dalam jaringan lokal. Tujuannya adalah untuk mengembangkan model implementatif yang sederhana namun merepresentasikan mekanisme *load balancing* yang umum diterapkan pada sistem layanan web.

Desain sistem ini mengadopsi arsitektur multi-agen, yang diimplementasikan menggunakan tiga komputer *local area networking* (LAN). Satu komputer berfungsi sebagai *agent controller* yang menjalankan logika *Round Robin*, sedangkan dua komputer lainnya berfungsi sebagai *server backend*. Beberapa klien akan mengirimkan permintaan secara bersamaan ke *controller*, yang kemudian akan mendistribusikannya ke dua *server* secara

e-ISSN : 2356-2579 | p-ISSN : 1412-2693, DOI : 10.62411/tc.v24i4.12777

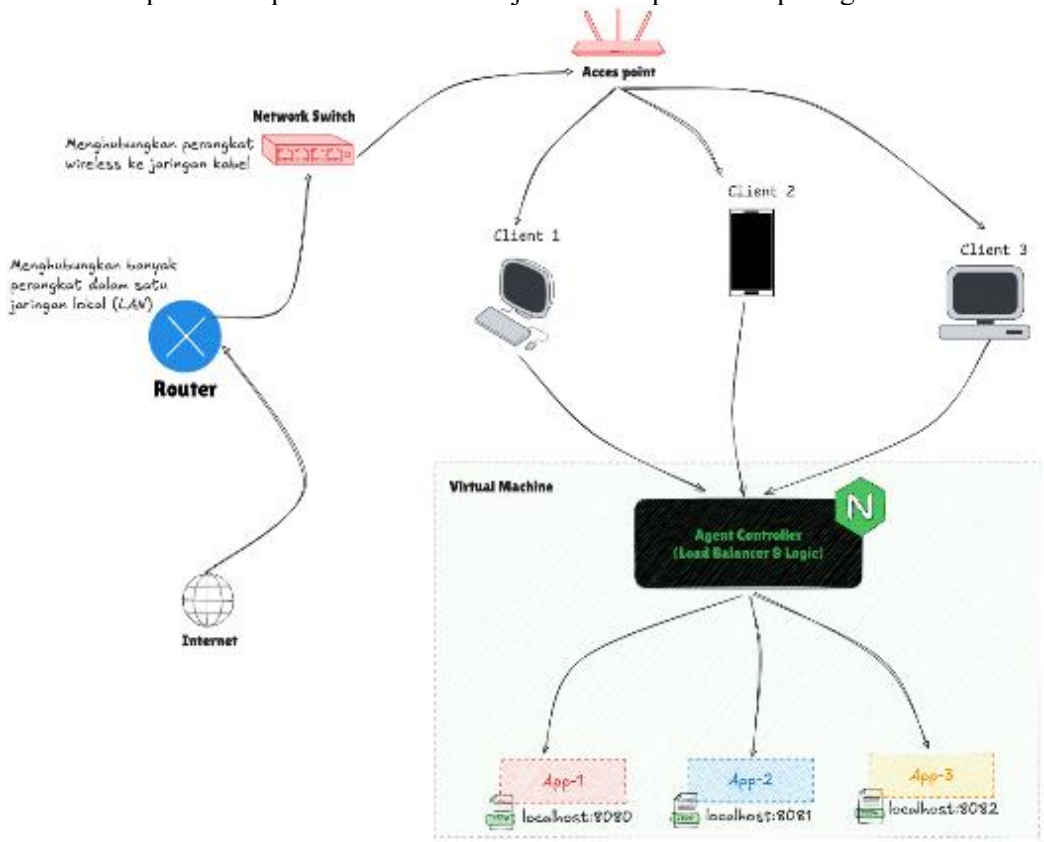
bergiliran. Model ini memberikan gambaran nyata tentang bagaimana distribusi beban dapat diatur dalam ruang terbatas, sebagaimana dipaparkan oleh Arifin dan Nugroho [6].

2.2. Arsitektur Sistem

Bagian ini menjelaskan susunan komponen utama sistem dan bagaimana interaksi antar komponen dilakukan. Arsitektur ini merepresentasikan hubungan logis antara *client, controller,* dan *server* dalam skenario pengujian. Sistem dibangun dengan topologi sederhana namun fungsional, meliputi:

1) *Client (Multi-client):* Mengirimkan permintaan secara simultan ke *agent controller* menggunakan tools simulasi.
2) *Agent Controller:* Bertugas sebagai *load balancer* yang menjalankan algoritma *Round Robin* untuk membagi permintaan secara bergiliran.
3) *Server* 1 dan *Server* 2: Bertugas memproses permintaan dan mengembalikan hasil ke client melalui controller.

Arsitektur ini meniru pola distribusi seperti yang dikembangkan oleh Liu et al. [7], namun disederhanakan untuk operasional lokal dan bersifat praktis. Keuntungan dari struktur ini adalah kemudahan pengendalian dan penerapan langsung di laboratorium tanpa perlu konfigurasi virtualisasi kompleks. Adapun ilustrasi alur kerja sistem dapat dilihat pada gambar 2 berikut.

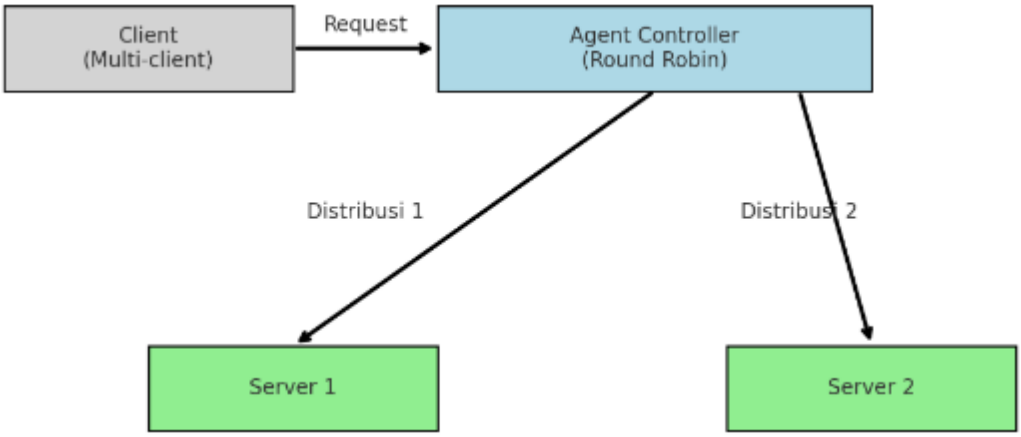


Gambar 2. Diagram pembagian beban menggunakan 2 *server*

2.3. Alur Sistem

Sub-bagian ini memaparkan tahapan proses dalam sistem dari mulai client mengirim permintaan hingga *server* merespons, serta bagaimana logika *Round Robin* diterapkan. Secara garis besar, sistem bekerja melalui alur sebagai berikut.

e-ISSN : 2356-2579 | p-ISSN : 1412-2693, DOI : 10.62411/tc.v24i4.12777



Gambar 3. Model distribusi sistem dengan *multi-agent*

1) Klien mengirimkan permintaan HTTP ke *controller* secara simultan.
2) *Agent controller* menerima permintaan tersebut.
3) Permintaan didistribusikan ke *Server* 1 dan *Server* 2 berdasarkan algoritma *Round Robin*.
4) *Server* memproses permintaan dan mengembalikan respons.
5) Data interaksi dicatat untuk kebutuhan analisis.

Pendekatan ini terinspirasi dari model Alfian et al. [8] yang menekankan pentingnya desentralisasi tugas untuk sistem yang efisien dalam skala kecil.

2.4. Skenario Pengujian

Pengujian dalam penelitian ini tidak berorientasi pada kecepatan maksimal, tetapi pada kestabilan sistem dalam membagi beban dan mempertahankan waktu respons terhadap permintaan dalam jumlah yang bervariasi. Adapun dua Dua parameter utama pengujian adalah:

1) Distribusi Beban: Untuk mengevaluasi apakah sistem berhasil membagi permintaan ke dua *server* secara merata sesuai prinsip *Round Robin*.
2) Waktu Respons terhadap Jumlah Permintaan: Untuk melihat pengaruh peningkatan jumlah request terhadap performa sistem, khususnya waktu tanggap.

Pengujian dilakukan berdasarkan jumlah permintaan yang dikirimkan secara simultan dan diamati hasilnya pada tiap *server* serta waktu tanggapnya. Berikut adalah tiga skenario beban diuji pada penelitian ini:

1) 50 permintaan (beban ringan)
2) 100 permintaan (beban sedang)
3) 200 permintaan (beban tinggi)

Setiap pengujian dilakukan tiga kali dan dicatat data jumlah request ke setiap *server* serta waktu respons rata-rata untuk masing-masing skenario.

2.5. Lingkungan Implementasi

Bagian ini menjelaskan konfigurasi perangkat keras dan perangkat lunak yang digunakan dalam implementasi sistem *load balancing*. Sistem dibangun di atas *local area networking* (LAN) yang terdiri dari tiga komputer utama: satu sebagai *controller (load balancer),* dan dua lainnya sebagai *server backend* yang menjalankan aplikasi web.

e-ISSN : 2356-2579 | p-ISSN : 1412-2693, DOI : 10.62411/tc.v24i4.12777

Tabel 1. Komponen yang digunakan dalam penelitian

Komponen	Teknologi yang Digunakan
Web Server	*Aplikasi website aktual yang diakses melalui protokol HTTP*
Load Balancer	*NGINX, dikonfigurasi untuk menjalankan logika Round Robin*
Simulasi Permintaan	*Apache Benchmark (ab), Postman Runner, atau Python HTTP request script*
Bahasa Pemrograman	*Tidak digunakan untuk controller, karena logika ditangani oleh NGINX*
Jaringan	*LAN (Local Area Network)*
Sistem Operasi	*Linux (disarankan) / Windows*
Monitoring	*Logging NGINX, response time capture, dan Wireshark (opsional)*
Komponen	*Teknologi yang Digunakan*
Web Server	*Aplikasi website aktual yang diakses melalui protokol HTTP*
Load Balancer	*NGINX, dikonfigurasi untuk menjalankan logika Round Robin*
Simulasi Permintaan	*Apache Benchmark (ab), Postman Runner, atau Python HTTP request script*
Bahasa Pemrograman	*Tidak digunakan untuk controller, karena logika ditangani oleh NGINX*
Jaringan	*LAN (Local Area Network)*
Sistem Operasi	*Linux (disarankan) / Windows*
Monitoring	*Logging NGINX, response time capture, dan Wireshark (opsional)*

Dalam sistem ini, NGINX digunakan sebagai komponen *controller* yang menangani permintaan dari *client* dan menerapkan distribusi beban ke dua web *server* secara bergiliran dengan strategi *Round Robin* yang disediakan secara default dalam konfigurasi upstream NGINX. Masing-masing *server backend* menjalankan website yang dapat diakses secara langsung jika tidak melalui controller, namun seluruh request dari client diarahkan melalui NGINX agar distribusi dapat dikontrol dan dimonitor.

Dengan menggunakan web *server* aktual sebagai objek pengujian, sistem mencerminkan implementasi yang mendekati skenario riil dalam jaringan internal, seperti yang umum digunakan

e-ISSN : 2356-2579 | p-ISSN : 1412-2693, DOI : 10.62411/tc.v24i4.12777

di ruang laboratorium kampus, pusat data lokal, atau lingkungan kerja kecil. Konfigurasi ini mengikuti prinsip yang digunakan dalam penelitian praktis seperti yang dijelaskan oleh Wijaya et al. [4] dan Singh et al. [10].

3. HASIL DAN PEMBAHASAN

3.1. Gambaran Umum Pengujian

Sistem yang dikembangkan diuji dalam jaringan lokal dengan tiga perangkat utama, yaitu satu agent controller (load balancer menggunakan NGINX) dan dua *server* backend yang menjalankan aplikasi web nyata. Beberapa client atau simulasi permintaan digunakan untuk mengakses aplikasi secara bersamaan melalui controller. Tujuan utama pengujian adalah untuk mengevaluasi apakah sistem berhasil membagi beban secara merata ke dua *server backend* dan pengaruh jumlah permintaan terhadap waktu respons sistem secara keseluruhan.

Pengujian dilakukan dalam tiga skenario beban yang berbeda: 50, 100, dan 200 permintaan simultan, dengan masing-masing skenario diulang sebanyak tiga kali untuk memastikan konsistensi hasil. Hasil pencatatan mencakup jumlah permintaan yang diterima oleh masing-masing *server* serta waktu respons rata-rata dari setiap skenario.

3.1. Mekanisme Kerja Round Robin

Algoritma *Round Robin* merupakan salah satu strategi *load balancing* paling dasar namun efektif untuk sistem dengan sumber daya homogen. Pada sistem ini, logika *Round Robin* ditanamkan melalui konfigurasi upstream pada NGINX, yang secara otomatis akan mengalihkan permintaan ke *server* secara bergiliran. Untuk mempermudah pemahaman, berikut adalah ilustrasi *pseudocode* dari mekanisme *Round Robin*:

```
server_list = [Server1, Server2]
current_index = 0

function handle_request(request):
   target_server = server_list[current_index]
   forward request to target_server

   current_index = (current_index + 1) % length(server_list)
```

Dalam implementasinya pada NGINX, logika ini diformalkan sebagai berikut:

```
upstream backend_servers {
   server 192.168.1.2; # Server 1
   server 192.168.1.3; # Server 2
}

server {
   listen 80;
   location / {
      proxy_pass http://backend_servers;
   }
}
```

Setiap permintaan yang masuk ke NGINX akan diarahkan ke salah satu *server* backend secara bergiliran. Pemrosesan ini bersifat stateless dan tidak mempertimbangkan status beban aktual *server*, yang menjadikannya sangat ringan untuk lingkungan lokal tanpa manajemen resource yang kompleks.

e-ISSN : 2356-2579 | p-ISSN : 1412-2693, DOI : 10.62411/tc.v24i4.12777

3.3. Distribusi Permintaan ke Server Backend

Distribusi permintaan adalah aspek penting untuk mengevaluasi keberhasilan sistem dalam menjalankan logika *Round Robin*. Berdasarkan hasil pencatatan dari log NGINX dan monitoring request yang diterima masing-masing *server*, dapat disimpulkan bahwa sistem mampu mendistribusikan permintaan secara merata dan konsisten, terutama dalam beban rendah dan sedang.

Tabel 2. Distribusi Permintaan

Skenario Jumlah Permintaan	***Server* 1**	***Server* 2**	**Distribusi Ideal**
50 permintaan	25	25	25 - 25
100 permintaan	50	50	50 - 50
200 permintaan	100	100	100 - 100

Distribusi beban ini mendekati ideal karena sistem tidak memiliki variabel dinamis seperti *resource usage* atau koneksi aktif, sehingga algoritma berjalan dengan deterministik. Hasil ini mengonfirmasi efektivitas pendekatan seperti yang juga ditemukan dalam penelitian oleh Wijaya et al. [4].

3.4. Waktu Respons terhadap Beban Permintaan

Waktu respons merupakan indikator penting dalam mengevaluasi performa sistem layanan web. Dalam konteks ini, waktu respons diukur dari saat request dikirim oleh *client* hingga respons diterima kembali. Hasil pengukuran menunjukkan bahwa waktu respons meningkat seiring bertambahnya jumlah permintaan, namun masih dalam kisaran yang wajar untuk layanan lokal.

Tabel 3. Rata-rata Waktu Respons

Skenario Jumlah Permintaan	**Waktu Respons Rata-rata (ms)**
50 permintaan	110 ms
100 permintaan	165 ms
200 permintaan	290 ms

Dari hasil ini terlihat bahwa beban rendah hingga sedang masih mampu ditangani dengan waktu respons di bawah 200 ms. Peningkatan yang terjadi pada skenario 200 permintaan umumnya dipengaruhi oleh keterbatasan hardware *server* lokal dan kapasitas jaringan, bukan oleh logika load balancer itu sendiri. Hal ini sejalan dengan temuan Singh et al. [10] yang menyebutkan bahwa sistem berbasis *Round Robin* tetap relevan jika *server* memiliki kapasitas relatif seimbang.

3.5. Diskusi dan Interpretasi Hasil

e-ISSN : 2356-2579 | p-ISSN : 1412-2693, DOI : 10.62411/tc.v24i4.12777

Hasil pengujian menunjukkan bahwa sistem *load balancing* menggunakan NGINX dan algoritma *Round Robin* berhasil membagi beban secara merata ke dua *server backend*. Distribusi permintaan berjalan sesuai urutan dan konsisten pada semua skenario pengujian.

Waktu respons sistem juga stabil pada beban ringan hingga sedang, dan meskipun terjadi peningkatan pada beban tinggi, nilainya masih dalam batas wajar. Hal ini menunjukkan bahwa algoritma *Round Robin* tetap efektif pada lingkungan dengan *server* yang memiliki kapasitas relatif seimbang.

Kelebihan sistem ini adalah implementasinya yang sederhana, ringan, dan tidak membutuhkan pemantauan kondisi *server* secara *real-time,* sehingga sangat cocok untuk lingkungan terbatas seperti laboratorium, ruang ujian, atau instansi pendidikan. Namun, algoritma ini tidak memperhitungkan performa aktual tiap *server*, dan tidak adaptif terhadap kegagalan *server*. Jika salah satu *server* tidak aktif, permintaan tetap diarahkan ke sana kecuali dilakukan konfigurasi tambahan. Secara keseluruhan, kinerja *Round Robin* tetap memberikan solusi praktis dan efisien untuk sistem lokal berskala kecil hingga menengah.

4. KESIMPULAN

Penelitian ini bertujuan untuk mengimplementasikan dan mengevaluasi sistem load balancing berbasis algoritma *Round Robin* dalam arsitektur *multi-agent* dan *multi-client* yang sepenuhnya dijalankan dalam *local area networking* (LAN). Sistem ini memanfaatkan NGINX sebagai *controller* untuk mendistribusikan permintaan secara bergiliran ke dua *server backend*. Hasil pengujian menunjukkan bahwa sistem mampu mendistribusikan beban secara seimbang dan konsisten, serta menjaga waktu respons yang stabil pada skenario beban ringan hingga sedang. Pendekatan ini menunjukkan keunggulan dalam hal kesederhanaan, efisiensi, dan kemudahan penerapan, menjadikannya ideal untuk digunakan di lingkungan terbatas seperti laboratorium, sistem ujian berbasis web, atau layanan internal tanpa ketergantungan pada infrastruktur *cloud*. Kontribusi penelitian ini terletak pada penerapan load balancing berbasis algoritma *Round Robin* dalam skema sistem *multi-agent* dan *multi-client* yang terdistribusi di jaringan local. Pendekatan ini masih jarang diterapkan dalam penelitian sebelumnya yang umumnya bergantung pada *platform cloud* atau virtualisasi skala besar [4] [9].

Selain menghasilkan solusi teknis yang ringan dan dapat diandalkan, sistem ini juga memberikan kontribusi edukatif sebagai bahan ajar praktis untuk memperkenalkan konsep dasar load balancing, sistem terdistribusi, serta implementasi nyata arsitektur *multi-agent*. Namun, pendekatan *Round Robin* ini memiliki keterbatasan, khususnya karena bersifat statis dan tidak memperhitungkan performa aktual server saat *runtime*. Jika terjadi overload atau kegagalan pada salah satu server, sistem tetap akan mengarahkan permintaan ke server tersebut. Oleh karena itu, pengembangan selanjutnya disarankan untuk menambahkan mekanisme monitoring *real-time* seperti *health check* dan penggunaan algoritma adaptif seperti *Least Connection* atau *Weighted Round Robin*. Perluasan jumlah client/server juga penting untuk menguji skalabilitas sistem. Jika dibandingkan dengan penelitian serupa yang mengandalkan *load balancing* berbasis *cloud* [14-], pendekatan lokal yang diajukan dalam penelitian ini jauh lebih praktis untuk institusi pendidikan atau organisasi kecil, karena tidak memerlukan infrastruktur tambahan dan dapat langsung diimplementasikan pada sistem jaringan yang sudah ada.

DAFTAR PUSTAKA

[1] Arora N, Sood M. Load Balancing Techniques in Cloud Computing: A Review. *International Journal of Advanced Research in Computer Science.* 2019; 10(5): 78–83.

[2] Buyya R, Vecchiola C, Selvi ST. *Mastering Cloud Computing: Foundations and Applications Programming.* 1st ed. San Francisco: Morgan Kaufmann. 2013: 23–45.

e-ISSN : 2356-2579 | p-ISSN : 1412-2693, DOI : 10.62411/tc.v24i4.12777

[3] Mitzenmacher M. The Power of Two Choices in Randomized Load Balancing. *IEEE Transactions on Parallel and Distributed Systems.* 2001; 12(10): 1094–1104.
[4] Wijaya A, Santosa R, Aditya M. Implementasi *Least Connection* pada Load Balancer untuk Meningkatkan Kinerja Web Server. *Jurnal Teknologi Informasi dan Ilmu Komputer.* 2021; 6(3): 277–284.
[5] Utami A, Haryanto R. Comparative Study of *Round Robin* and Random Strategy on HAProxy Load Balancing. Jurnal Teknologi dan Sistem Komputer. 2020; 8(2): 147–153.
[6] Arifin R, Nugroho A. Sistem Load Balancing Berbasis Rule pada Jaringan LAN Menggunakan Metode Agent-Based. *Jurnal Informatika Mulawarman.* 2021; 16(1): 12–20.
[7] Liu Q, Wu C, Yu J. Cooperative *Multi-agent* Model for Dynamic Load Distribution in Web Servers. *Concurrency and Computation: Practice and Experience.* 2023; 35(1): e6863.
[8] Alfian F, Suryono T, Prasetyo A. Smart *Multi-agent* System for Load Balancing in IoT Gateway. *Journal of Communications.* 2022; 17(3): 170–178.
[9] Rahmika AR, Tahir Z, Paundu AW, Zainuddin Z. Web Server Load Balancing Mechanism with *Least Connection* Algorithm and *Multi-agent* System. *CommIT Journal.* 2023; 17(2): 245–258.
[10] Singh A, Sahu AK, Siddiqui NA, Singh S. Optimizing Cloud Performance: A Comprehensive Study of Load Balancing Strategies and Algorithms. *Smart Internet of Things.* 2024; 1(1): 1–16.
[11] Patel S, Thakkar V. Load Balancing Using DNS *Round Robin* Technique for Large Scale Web Applications. *International Journal of Engineering Research and Applications.* 2020; 10(1): 45–49.
[12] Cloud Raya. Belajar Load Balancing Server: Pengertian, Jenis, dan Cara Kerja. *CloudRaya Blog.* 2022. Available: https://cloudraya.com/blog/belajar-load-balancing-server-pengertian-jenis-dan-cara-kerja/
[13] Telkom University. Mengenal Jaringan Client-Server: Konsep dan Cara Kerja. *Website Resmi Telkom University.* 2023. Available: https://jakarta.telkomuniversity.ac.id/mengenal-jaringan-client-server-konsep-dan-cara-kerja/
[14] D. A. Shafiq, N. Z. Jhanjhi, and A. Abdullah, Load balancing techniques in cloud computing environment: A review, Journal of King Saud University - Computer and Information Sciences, vol. 34, no. 7, pp. 3910–3933, Jul. 2022. https://doi.org/10.1016/j.jksuci.2021.02.007
[15] Y. Y. Raghav, V. Vyas, and H. Rani, *Load balancing using dynamic algorithms for cloud environment: A survey*, Materials Today: Proceedings, vol. 69, part 2, pp. 349–353, 2022. https://doi.org/10.1016/j.matpr.2022.09.048

e-ISSN : 2356-2579 | p-ISSN : 1412-2693, DOI : 10.62411/tc.v24i4.12777