PENGUJIAN PERFORMA LOAD BALANCING PADA WEB SERVER DENGAN ALGORITMA WEIGHTED LEAST CONNECTION DAN WEIGHTED ROUND ROBIN DI MICROSOFT AZURE

1st Hizkia Simanungkalit
D3 Teknologi Telekomunikasi
Telkom University
Bandung, Indonesia
hizkiasimanungkalit@student.telkomuniversity.ac.id

2nd Muhammad Iqbal, S.T.,M.T.
D3 Teknologi Telekomunikasi
Telkom University
Bandung, Indonesia
miqbal@telkomuniversity.ac.id

3rd Dr. Suci Aulia, S.T., M.T.
D3 Teknologi Telekomunikasi
Telkom University
Bandung, Indonesia
suciaulia@telkomuniversity.ac.id`

***Abstrak* — Peningkatan trafik pada layanan berbasis web kerap menimbulkan lonjakan permintaan yang berdampak pada waktu respons, stabilitas koneksi, dan efisiensi sistem. Untuk menjaga performa agar tetap optimal, penelitian ini menerapkan teknik *load balancing* menggunakan dua algoritma, yaitu *Weighted Least Connection* dan *Weighted Round Robin* pada *web server* di platform Microsoft Azure. Pengujian dilakukan dengan mengirimkan sejumlah permintaan dari sisi klien dan memantau tiga parameter kinerja utama yaitu waktu respons, kestabilan koneksi, serta throughput—dengan bantuan alat pemantauan Netdata. Hasil evaluasi menunjukkan bahwa Weighted Least Connection mampu menghasilkan throughput yang lebih tinggi serta waktu respons yang lebih singkat, namun kestabilan koneksinya menurun signifikan saat beban meningkat. Sebaliknya, Weighted Round Robin mempertahankan kestabilan koneksi dengan baik, walaupun throughput yang dihasilkan cenderung lebih rendah. Berdasarkan hasil ini, Weighted Round Robin dinilai lebih konsisten dalam menjaga kesehatan sistem secara keseluruhan, sementara Weighted Least Connection lebih rawan mengalami penurunan performa pada kondisi lalu lintas yang padat.**

***Kata kunci*— *Web Server, Load Balancing*, *Weighted Least Connection*, *Weighted Round Robin*, *Netdata*, Microsoft Azure.**

I. PENDAHULUAN

Perkembangan teknologi digital yang terus berlangsung telah mendorong peningkatan jumlah pengguna layanan berbasis web secara pesat. Berbagai pihak kini memanfaatkan jaringan untuk membangun aplikasi web yang dapat diakses secara luas, menjadikan *server web* sebagai komponen penting dalam menangani permintaan dalam jumlah besar [1]. Peningkatan pengguna dan kompleksitas layanan menuntut server menangani lebih banyak permintaan, yang tanpa pengelolaan beban memadai dapat menyebabkan penurunan performa dan gangguan akses. Sebagai solusi, load balancing digunakan untuk mendistribusikan traffic secara merata ke beberapa server, dengan tujuan meningkatkan throughput, mempercepat respons, dan mencegah server mengalami overload. Dengan distribusi yang seimbang, sistem tetap dapat bekerja secara optimal meskipun berada di bawah tekanan permintaan yang tinggi [2].

Banyak algoritma penyeimbangan beban telah dirancang untuk menyelaraskan distribusi beban dengan karakteristik lalu lintas dan kapasitas server, termasuk Round Robin, Koneksi Paling Sedikit, Round Robin Tertimbang, dan Koneksi Paling Sedikit Tertimbang. Algoritma berbobot, yaitu Weighted Round Robin dan Weighted Least Connection, mengalokasikan beban kerja berdasarkan kapasitas server. Weighted Least Connection meningkatkan Least Connection dengan menyertakan bobot kinerja, memungkinkan server dengan bobot lebih tinggi untuk mengelola lebih banyak koneksi aktif [3]. Sementara itu, algoritma *Weighted Round Robin* merupakan penyempurnaan dari *Round Robin*, yang memungkinkan distribusi beban lebih besar pada server atau kluster dengan sumber daya lebih tinggi, sekaligus mempertimbangkan perbedaan kapasitas pemrosesan setiap server [3].

Microsoft Azure adalah platform komputasi awan yang menyediakan layanan *Platform as a Service* (PaaS), yang memungkinkan pengguna menyewa dan memanfaatkan lingkungan pengembangan, termasuk sistem operasi, mesin basis data, dan jaringan, untuk menjalankan serta mengelola aplikasi tanpa harus membangun infrastruktur dari awal [4]. Layanan ini dapat dikonfigurasi untuk menggunakan algoritma penjadwalan trafik seperti *Weighted Least Connection* dan *Weighted Round Robin*, yang berperan dalam mendistribusikan beban secara lebih merata dan efisien.

Berdasarkan permasalahan tersebut, penelitian ini bertujuan untuk mengevaluasi performa algoritma *Weighted Least Connection* dan *Weighted Round Robin* dalam mendistribusikan *traffic* ke *web server* di Microsoft Azure. Evaluasi dilakukan berdasarkan parameter waktu respon, *throughput*, dan kestabilan koneksi. Penelitian ini juga mencakup perancangan sistem untuk menilai efektivitas kedua algoritma dalam menjaga kinerja saat beban meningkat.

II. KAJIAN TEORI

A. Load Balancing

Load balancing adalah metode dalam jaringan komputer yang mendistribusikan permintaan secara merata ke beberapa komputer atau kluster, bertujuan untuk memaksimalkan pemanfaatan sumber daya, meningkatkan throughput, mempercepat respons, dan menghindari overload pada sistem [5].

B. Cloud Computing

Cloud computing adalah model layanan teknologi informasi yang memungkinkan akses, pengelolaan, dan penyimpanan data secara terpusat melalui internet. Teknologi ini memanfaatkan sumber daya komputasi, termasuk server, penyimpanan, dan aplikasi, tanpa memerlukan infrastruktur fisik, melalui paradigma everything as a service, termasuk perangkat lunak sebagai layanan (SaaS), platform sebagai layanan (PaaS), dan infrastruktur sebagai layanan (IaaS).). Keuntungannya mencakup biaya yang efisien, fleksibilitas, dan kemampuan skala yang tinggi, sehingga menjadi solusi utama dalam pengembangan sistem TI modern [6].

C. Web Server

Web server merupakan sistem yang memberikan layanan sesuai permintaan pengguna melalui jaringan internet. Secara umum, *web server* terdiri atas tiga komponen utama, yaitu perangkat keras, sistem operasi, dan perangkat lunak aplikasi yang berfungsi mendukung komunikasi menggunakan protokol HTTP [7].

D. Docker

Docker adalah platform *open source* yang digunakan untuk mengotomatiskan penyebaran aplikasi ke dalam kontainer, yaitu lingkungan ringan yang memuat seluruh komponen yang dibutuhkan untuk menjalankan aplikasi. Berbeda dengan virtualisasi tradisional, docker tidak memerlukan sistem operasi terpisah pada setiap instansinya, sehingga lebih efisien dalam penggunaan sumber daya. Keunggulan ini menjadikan docker sebagai solusi yang efektif dalam pengembangan sistem terdistribusi [8].

E. Netdata

Netdata merupakan salah satu perangkat *monitoring* (pemantuan) sistem yang banyak digunakan pada platform *cloud* seperti *Amazon Web Services* (AWS) dan Microsoft Azure. Tool ini bekerja dengan mengumpulkan metrik sistem dan aplikasi melalui antarmuka aplikasi atau membaca langsung dari file sistem seperti */proc*, kemudian mengirimkan data tersebut ke kontroler untuk dianalisis lebih lanjut [9].

F. *Algoritma Weight Round Robin (WRR)*

Algoritma Round Robin Tertimbang adalah metode penjadwalan dalam sistem penyeimbangan beban yang mengalokasikan lalu lintas secara berurutan sesuai dengan bobot yang ditetapkan untuk setiap server. Server dengan bobot lebih besar akan mengelola lebih banyak permintaan, tetapi server dengan bobot serupa akan mendistribusikan beban secara merata, meningkatkan efisiensi distribusi lalu lintas [2].

G. Algoritma Weight Least Connection (WLC)

Algoritma Weighted Least Connection adalah teknik penjadwalan yang membagi beban lalu lintas dengan mempertimbangkan bobot tiap server dan jumlah koneksi aktif yang sedang ditangani. Bobot ini menunjukkan kapasitas atau kemampuan setiap server dalam menangani permintaan. Semakin tinggi bobot suatu server, maka semakin besar peluang server tersebut untuk menerima permintaan baru, terutama ketika jumlah koneksi aktif antar server seimbang. Pendekatan ini memungkinkan pengalokasian beban kerja yang lebih proporsional, serta memberikan fleksibilitas bagi *administrator* untuk menyesuaikan distribusi koneksi sesuai kebutuhan sistem [10].

H. Microsoft Azure

Microsoft Azure merupakan platform komputasi awan yang dikembangkan oleh Microsoft, yang menyediakan berbagai layanan berbasis *cloud* untuk mendukung pengembangan dan pengelolaan aplikasi. Melalui platform ini, pengembang dapat membangun aplikasi web, menyimpan data secara terpusat, serta mengintegrasikan berbagai layanan lintas platform dengan lebih efisien dan terstruktur [11].

III. METODE

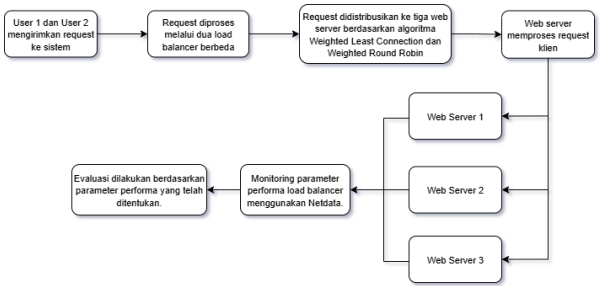
A. Alur Kerja Sistem

Dalam sistem ini, dua klien, User 1 dan User 2, mengirimkan permintaan ke layanan web. Permintaan tersebut kemudian diteruskan ke dua server load balancer, yang masing-masing dikonfigurasi dengan algoritma Weighted Least Connection dan Weighted Round Robin untuk mendistribusikan beban ke server web secara optimal. Algoritma Weighted Least Connection menyalurkan permintaan ke server dengan mempertimbangkan jumlah koneksi aktif serta bobot setiap server. Bobot ini merepresentasikan kapasitas server dalam menangani trafik. Semakin besar nilai bobot, maka semakin besar pula kemampuan server untuk menerima beban.

Dalam sistem ini, server 1 memiliki bobot 50, server 2 berbobot 30, dan server 3 berbobot 20. *Load balancer* akan menghitung rasio koneksi aktif terhadap bobot, contohnya server 1 dengan 25 koneksi memiliki rasio 25/50 = 0,5; server 2 dengan 15 koneksi = 15/30 = 0,5; dan server 3 dengan 18 koneksi = 18/20 = 0,9. Server dengan rasio terkecil dipilih untuk menerima permintaan. Perhitungan ini dilakukan secara dinamis setiap kali permintaan masuk, jika ada dua server dengan rasio sama, sistem akan membagi permintaan secara bergiliran di antara keduanya. Di sisi lain, algoritma Weighted Round Robin menyalurkan permintaan ke server secara bergantian sesuai bobot yang telah ditetapkan. Server dengan bobot yang lebih besar akan mendapatkan peningkatan jumlah permintaan. Misalnya, jika bobotnya adalah sebagai berikut: server 1 = 50, server 2 = 30, dan server 3 = 20, maka dari 10 permintaan, kira-kira 5 akan ditetapkan ke server 1, 3 ke server 2, dan 2 ke server 3. Distribusi dilakukan berurutan dan terus diulang agar tetap seimbang dan proporsional sesuai kapasitas masing-masing server.

Setelah permintaan diproses oleh *web server*, sistem dimonitor menggunakan Netdata yang terpasang di masing-masing *load balancer* untuk mencatat parameter performa seperti waktu respon, *throughput*, dan kestabilan koneksi.

Hasil *monitoring* digunakan untuk mengevaluasi efektivitas algoritma dalam mendistribusikan beban secara optimal.



Gambar 3. 1 Diagram Alur Kerja Load Balancing

B. Perancangan Komponen Infrastruktur

Berikut ini merupakan komponen infrastruktur yang digunakan pada pengujian performa *load balancing* dengan menggunakan kedua algoritma:

1. Dua virtual machine akan bekerja sebagai *load balancer*.
2. Tiga virtual machine akan bekerja sebagai *web server.*
3. Satu virtual machine akan bekerja sebagai *client* yang akan mengirimkan *requests* menuju kedua Algoritma.
4. Netdata akan digunakan sebagai alat *monitoring* untuk melakukan pemantauan terhadapat parameter yang akan diukur.

C. Rancangan Uji Performa Load Balancing

Pada rancangan ini akan dilakukan pengujian performa *load balancing* pada *web server* menggunakan algoritma *Weighted Least Connection* dan *Weighted Round Robin* di Microsoft Azure. Parameter yang diukur meliputi waktu respons, *throughput*, dan kestabilan koneksi. Pengiriman permintaan (*request*) dibagi menjadi empat jenis, di mana masing-masing jenis diuji sebanyak sepuluh kali. Adapun keempat jenis *request* tersebut adalah sebagai berikut:

1. 10 *request*, diuji sebanyak 10 kali
2. 50 *request*, diuji sebanyak 10 kali
3. 100 *request*, diuji sebanyak 10 kali
4. 150 *request*, diuji sebanyak 10 kali

IV. HASIL DAN PEMBAHASAN

Hasil dari pengujian performa *load balancing* pada web server dengan algoritma *Weighted Least Connection* dan *Weighted Round Robin* di microsoft azure dapat disimpulkan pada table-tabel parameter berikut:

A. Respons Time

Tabel 1 Respons Time

Algoritma	*Weighted Least Connection*	*Weighted Round Robin*
Request	Avg (ms)	Avg (ms)
10	230	232
50	225	225
100	218	224
150	223	224

Berdasarkan data pada Tabel 1, hasil dari *respons time* pada algoritma *Weighted Least Connection* menunjukkan performa yang lebih stabil dan efisien pada seluruh skenario pengujian. Bahkan pada beban rendah seperti 10 *request*, waktu responsnya sudah lebih baik dibandingkan *Weighted Round Robin*, dan keunggulan ini semakin terlihat pada skenario beban tinggi seperti 100 dan 150 *request*. Mekanisme dinamis yang mempertimbangkan jumlah koneksi aktif memungkinkan distribusi beban lebih seimbang, sehingga *respons time* dapat ditekan. Sebaliknya, *Weighted Round Robin* hanya menunjukkan performa sebanding pada beban sedang di 50 *request*, namun mulai tertinggal saat trafik meningkat karena tidak memperhitungkan kondisi server secara *real time*. Secara keseluruhan, *Weighted Least Connection* terbukti lebih andal dalam menjaga performa sistem saat beban meningkat secara bertahap.

B. Throughput

Tabel 2 Throughput

Algoritma	*Weighted Least Connection*	*Weighted Round Robin*
Request	Avg (pkt/s)	Avg (pkt/s)
10	20.29	12.74
50	27.72	16.82
100	27.08	21.43
150	35.81	24.85

Berdasarkan data pada Tabel 2, menunjukkan efektivitas yang lebih tinggi dari algoritma *Weighted Least Connection* dalam mengelola *throughput* dibandingkan dengan *Weighted Round Robin* pada setiap skenario beban. Keunggulan *Weighted Least Connection* ini berakar alokasi permintaan didasarkan pada jumlah koneksi aktif server, sehingga mampu menjaga utilisasi sumber daya secara optimal dan menangani peningkatan trafik dengan sangat baik. Sebaliknya, performa *Weighted Round Robin* yang lebih rendah yang dapat dilihat pada pola distribusinya yang kaku. Meskipun kapabilitasnya turut meningkat seiring bertambahnya beban, pendekatan tersebut tidak mempertimbangkan kondisi server secara nyata menjadi hambatan utama dalam mencapai efisiensi yang setara dengan *Weighted Least Connection*, terutama pada skenario beban tinggi.

C. Stabilitas Koneksi

Tabel 3 Stabilitas Koneksi

Algoritma	*Weighted Least Connection*	*Weighted Round Robin*
Request	Status	Status
10	Stabil	Stabil
50	Stabil	Stabil
100	Tidak Stabil	Tidak Stabil
150	Tidak Stabil	Tidak Stabil

Berdasarkan hasil data pada Tabel 3, menurut [12] hasil pengujian menunjukkan keunggulan algoritma *Weighted Round Robin* dibandingkan *Weighted Least Connection* dalam hal stabilitas saat menghadapi peningkatan beban. Meskipun keduanya berkinerja setara pada beban kerja ringan, perbedaan krusial tampak pada 100 *request*, di mana *Weighted Least Connection* mulai mengalami kegagalan sistem dengan tingkat *error* yang tinggi. Sebaliknya, *Weighted Round Robin* terbukti jauh lebih tangguh dengan hanya menunjukkan *error* dalam jumlah minimal, yang menegaskan kemampuannya untuk bertahan di bawah tekanan. Ketahanan *Weighted Round Robin* dalam menoleransi beban yang lebih tinggi menjadikannya pilihan yang lebih dapat diandalkan untuk menjaga kestabilan sistem dari lonjakan lalu lintas.

V. KESIMPULAN

Berdasarkan hasil pemodelan, perancangan, pengujian, dan evaluasi performa *load balancing* pada *web server* menggunakan algoritma *Weighted Least Connection* dan *Weighted Round Robin* di Microsoft Azure, diperoleh kesimpulan sebagai berikut:

1. Hasil pengujian menunjukan bahwa algoritma *Weighted Least Connection* memiliki kinerja yang lebih unggul dibandingkan *Weighted Round Robin* dalam menangani waktu respon. Namun pada *request* 50 kedua algoritma ini memiliki performa yang seimbang yaitu dengan rata-rata 225 ms.
2. Analisis pada skenario 100 *request* menunjukkan adanya pertukaran antara *throughput* dan stabilitas. Algoritma *Weighted Least Connection* memang mencatat *throughput* yang lebih tinggi sebesar 27,08 pkt/s, namun capaian ini terjadi dengan mengorbankan stabilitas, karena pada saat yang sama sistem masuk ke dalam kondisi gagal akibat munculnya *error* koneksi. Sebaliknya, *Weighted Round Robin* menghasilkan *throughput* yang lebih rendah (21,43 pkt/s) namun dengan tingkat kegagalan yang jauh lebih terkendali. Hal ini membuktikan bahwa angka *throughput weighted Least Connection* yang lebih tinggi bukanlah kinerja yang unggul, melainkan tanda bahwa sistem dipaksa bekerja melewati batas amannya.
3. Algoritma *Weighted Least Connection* dan *Weighted Round Robin* terbukti mampu menjaga kestabilan koneksi dan beroperasi dalam status stabil pada skenario beban kerja ringan hingga sedang. Namun, saat beban kerja meningkat pada skenario 100 *request*, keduanya memasuki kondisi gagal karena mulai menghasilkan *error* fungsional. Meskipun sama-sama gagal, terdapat perbedaan krusial pada tingkat keparahannya. *Weighted Round Robin* menunjukkan performa yang lebih baik dengan jumlah *error* yang minimal, sementara *Weighted Least Connection* mengalami kegagalan performa yang jauh lebih signifikan. Kondisi gagal ini berlanjut untuk kedua algoritma hingga beban puncak pada skenario 150 *request*, di mana keduanya sudah tidak mampu menangani beban kerja yang diberikan secara efektif.
4. Keunggulan utama dari algoritma *Weighted Least Connection* terletak pada mekanisme kerjanya yang dinamis, karena mampu mempertimbangkan kondisi aktual masing-masing server untuk menjaga distribusi beban tetap seimbang. Namun, algoritma ini memiliki keterbatasan pada kestabilan koneksi saat beban yang diterima semakin besar. Berbeda dengan *Weighted Round Robin* yang bekerja secara bergiliran dan tidak memperhitungkan kondisi server, sehingga kurang efektif dalam mencegah terjadinya kelebihan beban saat trafik tinggi. Meski demikian, pada pengujian dengan skenario 100 *request*, *Weighted Round Robin* justru menunjukkan kestabilan koneksi yang lebih baik dibandingkan *Weighted Least Connection*.

Berdasarkan hasil pengujian, dapat disimpulkan bahwa algoritma *Weighted Least Connection* menunjukkan kinerja yang lebih unggul dalam hal waktu respon dan *throughput*, terutama saat menghadapi trafik yang terus meningkat. Namun, algoritma ini memiliki tantangan pada kestabilan koneksi ketika beban permintaan menjadi tinggi. Di sisi lain, *Weighted Round Robin* meskipun memiliki kinerja yang lebih rendah dalam dua parameter utama, justru mampu mempertahankan kestabilan koneksi yang lebih baik pada skenario tertentu.

SARAN

Mengacu pada hasil dan pembahasan dalam Tugas Akhir ini, berikut adalah beberapa poin saran yang dapat dijadikan bahan pertimbangan untuk riset selanjutnya:

1. Pengujian dalam laporan ini dilakukan dengan skenario beban hingga 150 *request*. Untuk menguji ketahanan dan titik jenuh (*breaking point*) dari kedua algoritma, penelitian selanjutnya dapat meningkatkan *volume request* secara signifikan.
2. *Web server* yang digunakan dalam penelitian ini menjalankan konten statis yang diambil dari repositori *GitHub*. Akan sangat bermanfaat untuk melakukan pengujian serupa pada aplikasi dengan konten dinamis yang terhubung ke basis data atau layanan eksternal untuk melihat bagaimana pemrosesan di sisi server memengaruhi performa *load balancing*.
3. Pengujian ini berfokus pada tiga parameter utama: waktu respon, *throughput*, dan kestabilan koneksi. Untuk analisis yang lebih mendalam, disarankan untuk menambahkan parameter server-side seperti utilisasi CPU dan penggunaan memori pada masing-masing *web server*. Hal ini akan memberikan gambaran yang lebih jelas mengenai efisiensi sumber daya dan bagaimana setiap algoritma membebani server secara individual.

Proses pemantauan saat ini menggunakan *dashboard* Netdata untuk observasi. Untuk meningkatkan akurasi dan efisiensi, disarankan untuk mengembangkan sistem monitoring yang lebih otomatis.

Referensi

[1] M. N. A. Rizqi and I. K. D. Nuryana, "Analisis Perbandingan Kinerja Algoritma Weighted Round Robin dan Weighted Least Connection Menggunakan Load Balancing Nginx Pada Virtual Private Server (VPS)," *Journal of Informatics and Computer Science,* vol. 4, no. 1, pp. 67-75, 2022.

[2] F. P. Perdana, B. Irawan and R. Latuconsina, "ANALISIS PERFORMANSI LOAD BALANCING DENGAN ALGORITMA WEIGHTED ROUND ROBIN PADA SOFTWARE DEFINED NETWORK (SDN)," *e-Proceeding of Engineering,* vol. 4, no. 3, pp. 4161-4168, Desember 2017.

[3] G. Singh and K. Kaur, "An Improved Weighted Least Connection Scheduling Algorithm for Load Balancing in Web Cluster Systems," *An Improved Weighted Least Connection Scheduling Algorithm for Load Balancing in Web Cluster Systems,* vol. 05, no. 03, Maret 2018.

[4] A. Amrullah, A. Nugroho and Z. Ramadhan, "PERBANDINGAN KINERJA WEBSERVER PADA PENYEDIA LAYANAN CLOUD MICROSOFT AZURE DAN AMAZON WEB SERVICES MENGGUNAKAN METODE BENCHMARKING," *JINTEKS (Jurnal Informatika Teknologi dan Sains),* vol. 5, no. 1, p. 92 – 97, Februari 2023.

[5] S. D. Riskiono and D. Darwis, "Peran Load Balancing Dalam Meningkatkan Kinerja Web Server di Lingkungan Cloud," *JURNAL TEKNIK INFORMATIKA,* vol. 8, no. 2, pp. 1 - 8, November 2020.

[6] M. Marliana, "KEAMANAN DAN PENCEGAHAN DATABASE CLOUD COMPUTING UNTUK PENGGUNA LAYANAN," *Jurnal PRODUKTIF,* vol. 3, no. 2, pp. 331-336, 2019.

[7] S. N. Rifiera and H. Nurwasito, "Implementasi Load Balancing dan Failover pada Proses Migrasi Container Docker," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* vol. 6, no. 5, pp. 2025-2033, Mei 2022.

[8] D. S. Afis, M. Data and W. Yahya, "Load Balancing Server Web Berdasarkan Jumlah Koneksi Klien Pada Docker Swarm," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* vol. 3, no. 1, pp. 925-930, Januari 2019.

[9] Z. Zhang, P. He, J. Chen, W. Bai, W. Wang, G. Lu, X. Lin and B. Li, "Zero Overhead Monitoring for Cloud-native Infrastructure using RDMA," in *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)*, Bologna.

[10] R. Baeturohman and B. Santoso, "Analisis Kinerja Algoritme WRR dengan WLC pada Load Balancing Nginx," *Indonesian Journal of Computer Science v,* vol. 13, no. 1, pp. 1063-1076, Februari 2024.

[11] A. Gumelar and A. M. Bachtiar, "PEMBANGUNAN BACKEND UNTUK APLIKASI PENGAWASAN PENGGUNAAN INTERNET ANAK “DODO KIDS BROWSER” DENGAN TEKNOLOGI MICROSOFT AZURE," *Jurnal Ilmiah Komputer dan Informatika (KOMPUTA),* pp. 45-51.

[12] B. Gregg, Systems Performance: Enterprise and the Cloud, Upper Saddle River, NJ: Prentice Hall, 2014.