Sistemasi: Jurnal Sistem Informasi
Volume 15, Nomor 4, 2026: 1269-1276
ISSN:2302-8149
e-ISSN:2540-9719

Analisis Penyajian Konten Statis *NGINX* terhadap Kinerja *Backend* Galeri Foto berbasis *Docker*

Analysis of NGINX Static Content Delivery on the Performance of a Docker-based Photo Gallery Backend

[1]Itsna Nur Hamida, [2]Masy Ari Ulinuha, [3]Hery Mustofa*, [4]Khothibul Umam
[1,2,3,4]Program Studi Teknologi Informasi, Fakultas Sains dan Teknologi, Universitas Islam Negeri Walisongo
[1,2,3,4]Jl. Walisongo No.3-5, Tambakaji, Kec. Ngaliyan, Kota Semarang, Jawa Tengah 50185
e-mail: 23080960029@student.walisongo.ac.id, ulinuha@walisongo.ac.id, herymustofa@walisongo.ac.id, khothibul_umam@walisongo.ac.id

(***received*:** 6 February 2026, ***revised*:** 10 February 2026, ***accepted*:** 12 April 2026)

Abstrak

Website studio foto umumnya menyajikan konten multimedia dalam jumlah besar melalui modul galeri foto. Pada banyak implementasi, konten ini dilayani secara dinamis oleh *backend* aplikasi, yang dapat meningkatkan beban pemrosesan dan menurunkan kinerja sistem. Penelitian ini menganalisis dampak penyajian konten statis menggunakan NGINX terhadap kinerja *backend* modul galeri foto berbasis Docker. Metode eksperimen kuantitatif diterapkan dengan membandingkan dua skenario, yaitu penyajian galeri secara dinamis tanpa NGINX dan penyajian konten statis menggunakan NGINX. Pengujian dilakukan menggunakan Apache Benchmark dengan parameter permintaan dan konkurensi yang sama, disertai pemantauan penggunaan sumber daya *backend*. Hasil menunjukkan bahwa penyajian dinamis menghasilkan *throughput* lebih rendah (463,78 permintaan/detik) dengan waktu respons lebih tinggi (108,27 ms) serta penggunaan CPU *backend* yang signifikan. Sebaliknya, penggunaan NGINX meningkatkan *throughput* hingga 1105,25 rps, menurunkan waktu respons menjadi 45,33 ms, dan secara drastis mengurangi beban CPU dan memori *backend*. Temuan ini membuktikan bahwa pemisahan penyajian konten statis dan dinamis menggunakan NGINX efektif dalam meningkatkan kinerja *backend* pada aplikasi galeri foto berbasis Docker.

Kata kunci: kinerja *backend*, docker, NGINX, galeri foto, konten statis.

Abstract

Photo studio websites typically present large volumes of multimedia content through photo gallery modules. In many implementations, this content is served dynamically by the application backend, which can increase processing load and degrade system performance. This study analyzes the impact of static content delivery using NGINX on the performance of a Docker-based photo gallery backend. A quantitative experimental method was applied by comparing two scenarios: dynamic content delivery without NGINX and static content delivery using NGINX. The experiments were conducted using Apache Benchmark with identical request and concurrency parameters, accompanied by monitoring of backend resource utilization. The results show that dynamic delivery produces lower throughput (463.78 requests per second) and higher response time (108.27 ms), along with significant backend CPU usage. In contrast, the use of NGINX increases throughput to 1105.25 requests per second, reduces response time to 45.33 ms, and significantly lowers backend CPU and memory consumption. These findings demonstrate that separating static and dynamic content delivery using NGINX is effective in improving backend performance in Docker-based photo gallery applications.

Keywords: *backend performance, docker, NGINX, photo gallery, static content.*

1 Pendahuluan

Perkembangan teknologi web telah meningkatkan permintaan akan sistem informasi yang dapat memproses permintaan pengguna dengan cepat, stabil, dan efisien. Situs web modern tidak hanya

 ISSN:2302-8149
e-ISSN:2540-9719

menawarkan konten dinamis, tetapi juga jumlah besar konten statis seperti gambar, video, dan berkas multimedia. Permintaan yang tinggi terhadap konten ini memerlukan pengelolaan server yang optimal untuk menjaga kualitas layanan. Studi sebelumnya menunjukkan bahwa pilihan arsitektur server web memiliki dampak signifikan terhadap kinerja situs web, terutama dalam kemampuan menangani permintaan klien secara bersamaan[1].

Dalam sistem *website* berbasis galeri foto, sebagian besar permintaan pengguna bertujuan untuk mengakses file multimedia berukuran besar. Jika semua permintaan ini diproses secara dinamis di bagian belakang aplikasi (*backend*), hal ini dapat menyebabkan peningkatan beban pemrosesan dan penurunan kinerja sistem. Penelitian pada *website* berbasis WordPress menunjukkan bahwa pemrosesan konten dinamis memakan waktu lebih lama karena melibatkan kueri database dan eksekusi skrip *server-side*, yang mengakibatkan waktu muat halaman menjadi lebih lama[2].

Berbagai penelitian telah membandingkan kinerja server web, khususnya Apache dan NGINX, dalam memproses permintaan pengguna. Hasil pengujian menunjukkan bahwa NGINX, berkat arsitekturnya yang ramping dan berbasis *event-driven*, menawarkan keunggulan dibandingkan model berbasis proses Apache dalam memproses koneksi bersamaan dan menyajikan konten statis[3]. Selain itu, optimasi konfigurasi dan manajemen sumber daya NGINX telah terbukti secara signifikan meningkatkan *throughput* dan mengurangi latensi layanan web[4].

Penggunaan teknologi kontainer (misalnya, Docker) semakin banyak digunakan dalam pengembangan dan *deployment* aplikasi web, karena dapat meningkatkan mobilitas dan konsistensi sistem. Namun, beberapa penelitian menunjukkan bahwa lingkungan Docker dapat menimbulkan beban kinerja tertentu akibat konfigurasi sistem operasi dan infrastruktur yang digunakan[5]. Penelitian lain telah membandingkan kinerja server web berbasis Docker dengan server web non-Docker dan menunjukkan bahwa terdapat perbedaan kinerja tergantung pada skenario pengujian dan sumber daya yang tersedia[6].

Namun, pendekatan pemisahan konten statis dan dinamis telah terbukti sangat efektif dalam mengurangi beban *backend*. Implementasi NGINX sebagai *proxy* terbalik dan server untuk konten statis meningkatkan kecepatan akses, mengurangi beban CPU, dan mengurangi konsumsi memori server[7]. Penelitian tentang situs web statis dan dinamis juga menunjukkan bahwa penyampaian konten statis memberikan keunggulan dalam hal efisiensi sumber daya dan stabilitas sistem[8].

Selain kinerja, sistem pengiriman konten juga harus mempertimbangkan aspek keamanan dan distribusi data. Jaringan Pengiriman Konten (CDN) sering digunakan untuk mempercepat akses ke konten statis melalui mekanisme *caching* terdistribusi, tetapi mereka juga menimbulkan tantangan keamanan tersendiri[9]. Oleh karena itu, untuk sistem berukuran menengah, menggunakan NGINX untuk mengelola konten statis secara lokal menjadi alternatif yang lebih terkendali.

Berbagai penelitian telah membahas perbandingan server web, optimasi NGINX, dan kinerja Docker, namun sebagian besar penelitian masih berfokus pada pengujian umum atau perbandingan antar server web. Studi-studi seperti Chandra [1], [3] membandingkan kinerja server web secara umum, sementara penelitian lain [6] mengevaluasi Docker dalam lingkungan virtual. Namun, tidak satu pun dari penelitian tersebut yang mengukur secara langsung bagaimana penyajian konten statis melalui NGINX memengaruhi beban CPU dan RAM backend ketika dijalankan dalam kontainer Docker untuk kasus penggunaan modul galeri foto. Oleh karena itu, diperlukan penelitian yang secara khusus mengevaluasi efektivitas penyajian konten statis dan dinamis dalam konteks tersebut.

Karena masalah-masalah tersebut, penelitian ini bertujuan untuk menganalisis dampak pengiriman konten statis menggunakan NGINX pada modul galeri foto berbasis Docker terhadap kinerja *backend*. Kontribusi penelitian ini mencakup pengembangan rencana arsitektur sistem yang efisien guna meningkatkan kinerja, mengoptimalkan penggunaan sumber daya, dan menjaga stabilitas situs web multimedia.

2 Tinjauan Literatur

Kinerja server web merupakan salah satu faktor terpenting yang menentukan kualitas layanan situs web. Chandra [1] melakukan pengujian dengan Apache dan NGINX menggunakan Apache Benchmark dan menunjukkan khususnya bahwa NGINX dapat memproses permintaan klien dengan waktu respons yang lebih rendah daripada Apache pada beban permintaan yang tinggi. Hasil ini

 ISSN:2302-8149
e-ISSN:2540-9719

menunjukkan bahwa arsitektur berbasis *event-driven* NGINX lebih efisien dalam mengelola koneksi bersamaan daripada model berbasis proses Apache.

Penelitian lain oleh Herman et al. [3] membandingkan kinerja Apache, NGINX, dan Lighttpd dalam lingkungan VPS menggunakan alat WRK. Hasilnya menunjukkan bahwa meskipun NGINX memerlukan konfigurasi yang lebih kompleks daripada Apache, NGINX menunjukkan kinerja yang lebih unggul dalam memproses koneksi bersamaan dan transfer data. Hasil ini mendukung pandangan bahwa pemilihan server web harus disesuaikan dengan persyaratan dan karakteristik beban sistem. Satwika dan Semadi [10] juga membuktikan bahwa NGINX memiliki performa yang lebih efisien dibandingkan Apache dalam berbagai skenario pengujian jaringan, termasuk pada lingkungan IPv6.

Selain perbandingan antara server web, beberapa penelitian menekankan perbedaan dalam penyajian konten statis versus dinamis. Tomiša et al. [2] menganalisis kinerja situs web WordPress dalam bentuk dinamis dan statis. Hasil eksperimen menunjukkan bahwa versi statis memiliki waktu muat yang lebih singkat dan beban server yang lebih rendah karena tidak memerlukan pemrosesan database dan skrip *server-side* untuk setiap permintaan. Hal ini menunjukkan bahwa memisahkan konten statis dan dinamis dapat meningkatkan efisiensi sistem.

Padilla dan Yuliadi [8] melakukan penelitian serupa yang membandingkan situs web statis dan dinamis dalam hal kinerja, keamanan, dan efisiensi. Penelitian ini menyimpulkan bahwa meskipun situs web statis lebih unggul dalam hal penyederhanaan sistem dan pemanfaatan sumber daya, situs web dinamis lebih fleksibel namun memerlukan infrastruktur yang lebih kompleks. Temuan ini berlaku untuk pengelolaan modul galeri foto di mana konten multimedia statis mendominasi.

Upaya untuk mengoptimalkan kinerja server web telah diteliti secara komprehensif dengan mengimplementasikan NGINX sebagai *reverse proxy* dan server *cache*. Data et al. [7] mengimplementasikan NGINX sebagai *reverse proxy* pada server LAMP dengan spesifikasi rendah. Hasil pengujian menunjukkan waktu muat berkurang hingga 96% dan pengurangan signifikan dalam penggunaan CPU dan memori. Hal ini menunjukkan bahwa NGINX dapat secara efektif mengurangi beban *backend* melalui mekanisme *caching* dan pengalihan permintaan.

Rafli et al. [11] telah menunjukkan bahwa penggunaan NGINX sebagai *reverse proxy* dan *load balancer* dapat meningkatkan stabilitas sistem dan mengurangi waktu respons di bawah beban tinggi. Temuan ini menyoroti peran NGINX dalam mengelola lalu lintas jaringan secara efisien.

Wang dan Kai [4] meneliti optimasi kinerja server web berbasis NGINX dengan menyesuaikan parameter konfigurasi seperti jumlah *thread*, *cache*, dan waktu koneksi. Hasil penelitian menunjukkan bahwa optimasi konfigurasi dapat secara signifikan meningkatkan *throughput* dan efisiensi penggunaan sumber daya. Penelitian ini membuktikan bahwa kinerja NGINX tidak hanya dipengaruhi oleh arsitekturnya, tetapi juga oleh konfigurasi sistem yang digunakan.

Mengenai lingkungan berbasis kontainer, Sobieraj dan Kotyński [5] menganalisis kinerja Docker di berbagai sistem operasi. Hasil penelitian menunjukkan bahwa Docker menunjukkan kinerja yang lebih unggul pada sistem Linux dibandingkan dengan Windows dan macOS, karena tidak memerlukan lapisan virtualisasi tambahan. Temuan ini menyarankan bahwa konfigurasi lingkungan eksekusi memengaruhi kinerja aplikasi berbasis kontainer. Mabotha et al. [12] berpendapat bahwa arsitektur layanan berbasis Docker, jika dikonfigurasi dengan benar, dapat mendukung kinerja optimal dalam sistem terdistribusi. Hal ini menunjukkan bahwa penggunaan Docker tidak hanya mendukung fleksibilitas tetapi juga menjaga kinerja aplikasi.

Ramadan et al. [6]membandingkan kinerja server web yang berjalan dalam lingkungan Docker dan lingkungan non-Docker pada mesin virtual. Hasil penelitian menunjukkan bahwa server tanpa Docker memiliki penggunaan CPU yang lebih rendah, sementara penggunaan RAM relatif stabil di kedua lingkungan. Hal ini menunjukkan bahwa penggunaan Docker dapat mempengaruhi kinerja sistem tergantung pada beban kerja dan konfigurasi infrastruktur.

Banyak peneliti juga telah melakukan penelitian mengenai kinerja server web dalam lingkungan kontainer. Penelitian yang mengevaluasi kinerja Apache dan NGINX berbasis Docker menunjukkan bahwa dalam lingkungan kontainer, NGINX unggul dalam menangani koneksi simultan dan efisiensi penggunaan sumber daya [13]. Hasil ini menunjukkan bahwa dalam lingkungan Docker, pilihan server web memiliki dampak signifikan terhadap kinerja sistem.

Selain kinerja, Putra dan Affandi [14] juga meneliti penggunaan Docker dan NGINX *Proxy* Manager untuk meningkatkan keamanan web. Hasil penelitian menunjukkan bahwa kombinasi

 ISSN:2302-8149
e-ISSN:2540-9719

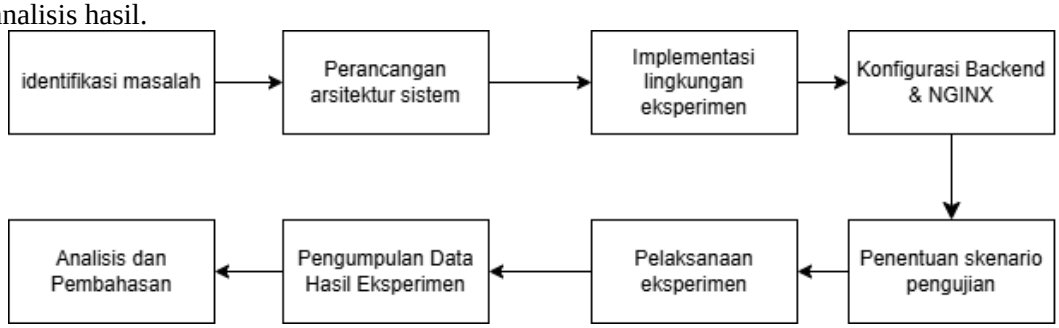
teknologi ini dapat meningkatkan kemampuan pertahanan terhadap berbagai serangan siber tanpa mengorbankan stabilitas layanan. Meskipun penelitian ini berfokus pada keamanan, ditemukan bahwa integrasi NGINX dan Docker juga berpotensi mendukung keandalan sistem secara keseluruhan. Mengenai keamanan aplikasi web, berbagai penelitian menunjukkan bahwa kerentanan seperti *cross-site scripting* (XSS) dan serangan *man-in-the-middle* tetap menjadi ancaman serius bagi sistem web modern[15]. Situasi ini menyoroti pentingnya menyeimbangkan optimasi kinerja *backend* dengan implementasi arsitektur yang aman dan terkelola.

Berdasarkan tinjauan terhadap penelitian yang ada, dapat disimpulkan bahwa NGINX menawarkan keunggulan dalam pemrosesan konten statis dan penanganan koneksi bersamaan, serta penggunaan Docker memungkinkan fleksibilitas dalam administrasi sistem. Namun, sebagian besar penelitian masih berfokus pada perbandingan umum antara server web, optimasi konfigurasi, atau evaluasi terpisah terhadap lingkungan kontainer. Masih relatif sedikit penelitian yang secara khusus membahas dampak penyajian konten statis dengan NGINX dalam lingkungan berbasis Docker terhadap kinerja modul galeri foto di sisi *backend*.

Oleh karena itu, penelitian ini berfokus pada pengkajian integrasi antara NGINX sebagai server konten statis dan Docker sebagai platform kontainerisasi dalam lingkungan aplikasi galeri foto. Fokus ini bertujuan untuk mengisi celah dalam penelitian sebelumnya dengan menyediakan bukti empiris mengenai dampak pemisahan konten statis dan dinamis terhadap kinerja *backend*.

3 Metode Penelitian

Penelitian ini menggunakan metodologi eksperimental dengan pendekatan kuantitatif untuk menganalisis dampak penyajian konten statis melalui NGINX terhadap kinerja *backend* berbasis Docker untuk modul galeri foto. Lingkungan pengujian dijalankan pada sistem dengan spesifikasi prosesor Intel Core i5-1155G7 (2 core) dan RAM sebesar 2 GB. Arsitektur sistem terdiri dari backend berbasis Apache dan PHP yang berjalan dalam container Docker, serta NGINX yang berfungsi sebagai server untuk penyajian konten statis. Pendekatan ini dipilih karena memungkinkan pengujian kinerja sistem secara terkontrol dengan membandingkan dua skenario pengujian dalam lingkungan yang sama[2], [8]. Alur penelitian pada Gambar 1 menunjukkan tahapan penelitian yang meliputi identifikasi masalah, perancangan arsitektur sistem, implementasi lingkungan eksperimen, konfigurasi backend dan NGINX, penentuan skenario pengujian, pelaksanaan eksperimen, pengumpulan data, hingga analisis hasil.

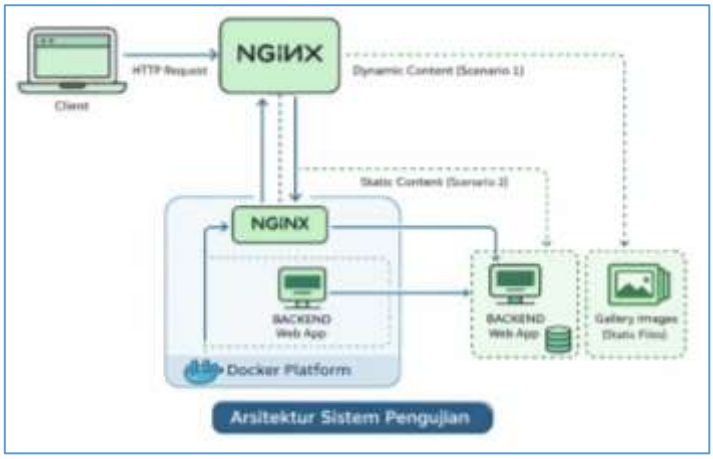


Gambar 1 Alur tahapan penelitian

Lingkungan penelitian disiapkan menggunakan platform Docker yang berjalan pada sistem operasi Ubuntu Server. Sistem tersebut mencakup kontainer-kontainer utama, termasuk *backend* aplikasi berbasis Apache dan PHP, basis data MySQL, dan NGINX yang bertindak sebagai server untuk penyajian konten statis. NGINX dikonfigurasi untuk melayani konten statis melalui direktif alias pada path /images/, sehingga file gambar disajikan langsung tanpa melalui backend. Sementara itu, permintaan lainnya diteruskan ke backend Apache menggunakan proxy_pass. Tidak digunakan mekanisme caching server-side seperti proxy_cache atau fastcgi_cache memastikan hasil pengujian merefleksikan perbedaan arsitektur sistem. NGINX dijalankan menggunakan konfigurasi default tanpa tuning tambahan. Semua layanan dikonfigurasi dalam lingkungan terisolasi tunggal untuk memastikan konsistensi pengujian dan meminimalkan pengaruh faktor eksternal. Gambar 2 menunjukkan arsitektur sistem pengujian yang terdiri dari klien, NGINX sebagai gateway, backend

 ISSN:2302-8149
e-ISSN:2540-9719

aplikasi berbasis Apache dan PHP, serta platform Docker pada dua skenario pengujian, yaitu penyajian konten dinamis dan statis.



Gambar 2 Arsitektur sistem pengujian konten dinamis dan statis

Pengujian dilakukan dalam dua skenario utama. Pada skenario pertama, konten galeri foto disajikan secara dinamis (semua permintaan file gambar diproses melalui *backend* aplikasi), sedangkan pada skenario kedua, konten disajikan secara statis (file gambar disajikan langsung dari NGINX tanpa melalui *backend*). Dataset dan konfigurasi sistem yang identik digunakan dalam kedua skenario untuk memastikan perbandingan yang adil.

Pengujian dilakukan menggunakan Apache Benchmark dengan parameter 1000 *request* (-n 1000) dan concurrency level 50 (-c 50) pada setiap skenario. Dua skenario pengujian digunakan, yaitu: (1) penyajian konten melalui *backend* tanpa NGINX, dan (2) penyajian konten statis langsung melalui NGINX. Dataset yang digunakan berupa *file* gambar dengan ukuran berkisar antara 30 KB hingga 1 MB, yang merepresentasikan konten multimedia pada aplikasi galeri foto.

Parameter kinerja yang diukur dalam penelitian ini meliputi kueri per detik, waktu respons rata-rata (waktu per kueri), pemanfaatan CPU *backend*, dan pemanfaatan memori *backend* yang dipantau menggunakan docker stats. Data uji dianalisis secara kuantitatif dengan membandingkan nilai rata-rata setiap parameter dalam dua skenario uji.

Analisis ini berfokus pada evaluasi efektivitas penggunaan NGINX untuk melayani konten statis dalam mengurangi beban pada *backend* dan meningkatkan kinerja sistem secara keseluruhan. Hasil analisis kemudian digunakan untuk menarik kesimpulan tentang dampak arsitektur pengiriman konten terhadap kinerja modul galeri foto berbasis Docker.

4 Hasil dan Pembahasan

Uji kinerja dilakukan sesuai dengan metode yang dijelaskan pada bagian sebelumnya dan membandingkan skenario pengiriman konten galeri foto dinamis tanpa NGINX dengan penyajian konten statis menggunakan NGINX. Parameter yang dianalisis meliputi jumlah permintaan per detik, waktu per permintaan, beban CPU *backend*, dan penggunaan memori *backend*. Hasil pengujian pada skenario penyajian konten dinamis tanpa NGINX disajikan pada Tabel 1.

Tabel 1 Hasil pengujian performa konten dinamis tanpa NGINX

Test	*Request/sec*	*Time per Request (ms)*	*CPU Backend (Peak)*	RAM *Backend*
1.	462,19	108,180	124,07 %	25,63 MiB
2.	427,31	117,011	118,75 %	25,46 MiB
3.	501,85	99,632	129,10 %	25,46 MiB
Rata-rata	463,78	108,27	123,97 %	25,52 MiB

 ISSN:2302-8149
e-ISSN:2540-9719

Menurut Tabel 1, rata-rata jumlah permintaan yang dapat ditangani oleh *backend* adalah 463,78 permintaan per detik, dengan waktu respons rata-rata 108,27 ms. Penggunaan CPU *backend* mencapai puncak rata-rata 123,97%, menunjukkan bahwa *backend* mengalami beban pemrosesan yang tinggi karena harus menangani semua permintaan, termasuk pengambilan dan pemrosesan file gambar dinamis. Kondisi ini mengindikasikan adanya tekanan beban tinggi akibat banyaknya permintaan simultan (*high concurrency),* yang dapat menyebabkan peningkatan latensi dan penurunan efisiensi sistem apabila tidak diimbangi dengan mekanisme distribusi beban yang efektif[16].

Tabel 2 Hasil pengujian performa konten statis menggunakan NGINX

Test	*Request/sec*	*Time per Request (ms)*	CPU *Backend (Peak)*	RAM *Backend*
1.	1129,78	44,256	0,01 %	12,01 MiB
2.	1147,68	43,566	0,01 %	12,01 MiB
3.	1038,30	48,156	0,01 %	12,01 MiB
Rata-rata	1105,25	45,33	0,01 %	12,01 MiB

Tabel 2 menunjukkan bahwa penggunaan NGINX sebagai server konten statis secara signifikan meningkatkan *throughput* menjadi rata-rata 1.105,25 permintaan per detik dan mengurangi waktu respons menjadi 45,33 ms. Peningkatan kinerja ini sejalan dengan penelitian Wang dan Kai yang menyatakan bahwa performa web server sangat dipengaruhi oleh konfigurasi sistem, sumber daya, dan optimasi server, sehingga penggunaan NGINX dapat meningkatkan efisiensi dan *throughput* sistem secara signifikan[4]. Penggunaan CPU yang sangat rendah di *backend* menunjukkan bahwa sebagian besar beban pemrosesan berhasil dipindahkan dari *backend* ke NGINX, memungkinkan *backend* untuk fokus pada penanganan konten dinamis.

Membandingkan kedua skenario menunjukkan bahwa menggunakan NGINX untuk menyajikan konten statis menghasilkan *throughput* lebih dari dua kali lipat dibandingkan dengan menyajikan konten dinamis tanpa NGINX. Selain itu, waktu respons sistem berkurang lebih dari 50%, yang secara langsung meningkatkan pengalaman pengguna.

Penurunan signifikan dalam penggunaan CPU dan memori *backend* menunjukkan bahwa memisahkan konten statis dan dinamis meningkatkan efisiensi penggunaan sumber daya sistem. Dengan mengurangi beban pada *backend*, potensi kemacetan sistem diminimalkan, membantu menjaga stabilitas layanan *website*. Hasil ini sejalan dengan temuan sebelumnya yang menunjukkan bahwa NGINX unggul dalam menangani konten statis dan koneksi bersamaan secara efisien, terutama dalam lingkungan berbasis kontainer.

Nilai throughput pada skenario tanpa NGINX berkisar antara 427,31 hingga 501,85 *request* per detik, sedangkan pada skenario dengan NGINX berkisar antara 1038,30 hingga 1147,68 *request* per detik, menunjukkan hasil yang konsisten. Uji statistik menggunakan independent t-test menunjukkan nilai p sebesar 0,000089 ($p < 0,05$), yang mengindikasikan adanya perbedaan yang signifikan antara kedua skenario. Namun demikian, mengingat jumlah sampel yang terbatas, hasil ini digunakan sebagai indikasi yang mendukung temuan eksperimen.

Hasil penelitian ini sejalan dengan temuan Chandra [1]dan Herman et al. [3], yang menunjukkan bahwa NGINX memberikan kinerja yang lebih unggul daripada Apache dalam menangani permintaan bersamaan. Selain itu, hasil ini sejalan dengan studi oleh Satwika dan Semadi [10], yang menunjukkan efisiensi NGINX dalam berbagai skenario jaringan, serta Ramadhan et al. [6], yang menganalisis kinerja server web dalam lingkungan virtual dan kontainer. Hal ini menyoroti pentingnya keunggulan NGINX dalam menyajikan konten statis di berbagai platform dan arsitektur sistem.

Keunggulan penelitian ini adalah penyelidikan empiris integrasi NGINX sebagai server konten statis dalam modul galeri foto berbasis Docker, dengan fokus pada beban kerja multimedia. Sementara studi sebelumnya umumnya berfokus pada perbandingan server web atau optimasi sistem secara umum, penelitian ini secara khusus mengevaluasi pemisahan konten statis dan dinamis dalam konteks aplikasi galeri foto berbasis kontainer. Selain itu, dengan menggunakan skenario uji terkontrol dan parameter kinerja komprehensif, hasil yang diperoleh lebih mencerminkan kondisi penggunaan dunia nyata.

 ISSN:2302-8149
e-ISSN:2540-9719

5 Kesimpulan

Penelitian ini menunjukkan bahwa penggunaan NGINX sebagai server konten statis secara signifikan meningkatkan kinerja backend berbasis Docker pada modul galeri foto. Hasil pengujian memperlihatkan peningkatan *throughput* dan penurunan waktu respons yang signifikan dibandingkan penyajian konten secara dinamis, serta pengurangan beban penggunaan sumber daya backend. Pemisahan antara konten statis dan dinamis terbukti efektif dalam meningkatkan efisiensi dan stabilitas sistem, sehingga backend dapat lebih optimal dalam memproses logika aplikasi pada lingkungan web berbasis multimedia.

Referensi

[1] A. Y. Chandra, "Analisis Performansi Antara *Apache & Amp; Nginx Web Server dalam menangani Client Request*," *Jurnal Sistem dan Informatika (JSI)*, Vol. 14, No. 1, pp. 48–56, Nov. 2019, DOI: 10.30864/jsi.v14i1.248.

[2] M. Tomiša, M. Milković, and M. Čačić, "*Performance Evaluation of Dynamic and Static WordPress-based Websites*," in *2019 23rd International Computer Science and Engineering Conference (ICSEC)*, IEEE, Oct. 2019, pp. 321–324. DOI: 10.1109/ICSEC47112.2019.8974709.

[3] W. Wicoksono, H. A. Mustaqhim, P. P. Anwas, dan L. N. L. Badratul, "*Performance Comparison of NGINX, Apache, and Lighttpd using WRK on a Debian*," *bit-Tech*, Vol. 8, No. 1, pp. 670–680, Aug. 2025, DOI: 10.32877/bt.v8i1.2661.

[4] J. Wang and Z. Kai, "*Performance Analysis and Optimization of Nginx-based Web Server*," *J. Phys. Conf. Ser.*, Vol. 1955, No. 1, p. 012033, Jun. 2021, DOI: 10.1088/1742-6596/1955/1/012033.

[5] M. Sobieraj and D. Kotyński, "*Docker Performance Evaluation Across Operating Systems*," *Applied Sciences*, Vol. 14, No. 15, p. 6672, Jul. 2024, DOI: 10.3390/app14156672.

[6] F. K. Ramadhan, G. Garno, and A. Solehudin, "*Comparative Study of Web Server Performance Testing with and without Docker based on Virtual Machines*," *Journal of Applied Informatics and Computing*, Vol. 8, No. 1, pp. 155–166, Jul. 2024, DOI: 10.30871/jaic.v8i1.3884.

[7] M. Data, M. Luthfi, and W. Yahya, "*Optimizing Single Low-End LAMP Server using NGINX Reverse Proxy Caching*," in *2017 International Conference on Sustainable Information Engineering and Technology (SIET)*, IEEE, Nov. 2017, pp. 21–23. DOI: 10.1109/SIET.2017.8304102.

[8] F. Az-zahra and S. P. Yuliadi, "Analisis Perbandingan Kinerja *Website* Statis dan Dinamis dalam Optimalisasi Layanan Informasi Digital," *Jurnal Sadewa : Publikasi Ilmu Pendidikan, Pembelajaran dan Ilmu Sosial*, Vol. 3, No. 4, pp. 91–100, Oct. 2025, DOI: 10.61132/sadewa.v3i4.2430.

[9] M. Ghaznavi, E. Jalalpour, M. A. Salahuddin, R. Boutaba, D. Migault, and S. Preda, "*Content Delivery Network Security: A Survey*," *IEEE Communications Surveys & Tutorials*, Vol. 23, No. 4, pp. 2166–2190, Jun. 2021, DOI: 10.1109/COMST.2021.3093492.

[10] I. K. S. Satwika and K. N. Semadi, "Perbandingan Performansi *Web Server Apache* dan *Nginx* dengan menggunakan IPv6," *SCAN - Jurnal Teknologi Informasi dan Komunikasi*, Vol. 15, No. 1, Feb. 2020, DOI: 10.33005/scan.v15i1.1847.

[11] M. Rafli, "Jurnal Pengujian Kinerja *Load Balancing Web Server* menggunakan *Nginx Riverse Proxy berbasis OS Centos 7*," *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, Vol. 9, No. 3, pp. 1824–1840, Sep. 2022, DOI: 10.35957/jatisi.v9i3.2185.

[12] E. Mabotha, N. E. Mabunda, and A. Ali, "*A Dockerized Approach to Dynamic Endpoint Management for RESTful Application Programming Interfaces in Internet of Things Ecosystems*," *Sensors*, Vol. 25, No. 10, p. 2993, May 2025, DOI: 10.3390/s25102993.

[13] W. M. C. J. T. Kithulwatta, K. P. N. Jayasena, B. T. G. S. Kumara, and R. M. K. T. Rathnayaka, "*Performance Evaluation of Docker-based Apache and Nginx Web Server*," in

2022 3rd International Conference for Emerging Technology (INCET), IEEE, May 2022, pp. 1–6. DOI: 10.1109/INCET54531.2022.9824303.

[14] M. Y. S. Putra and A. S. Affandi, "Analisis Tingkat Efektivitas *Cloudflare, Docker* dan *Nginx Proxy Manager* sebagai Sarana untuk meningkatkan Keamanan *Web*," *Jurnal Ilmiah Teknologi Informasi Asia*, Vol. 19, No. 1, pp. 24–30, Mar. 2025, DOI: 10.32815/jitika.v19i1.1070.

[15] C. Catalano, A. Chezzi, V. S. Barletta, and F. Tommasi, "*Defeating FIDO2/CTAP2/WebAuthn using Browser in the Middle and Reflected Cross Site Scripting*," *Journal of Computer Virology and Hacking Techniques*, Vol. 21, No. 1, p. 11, May 2025, DOI: 10.1007/s11416-025-00556-2.

[16] C. Ma and Y. Chi, "*Evaluation Test and Improvement of Load Balancing Algorithms of Nginx*," *IEEE Access*, Vol. 10, pp. 14311–14324, 2022, DOI: 10.1109/ACCESS.2022.3146422.