Perbandingan Performa Aplikasi E-Commerce Dalam Lingkungan Kontainer dan Lingkungan Non-Kontainer

Muhammad Zikrillah Ihsan
Jurusan Teknik Informatika
Universitas Sriwijaya
Palembang, Indonesia
mzikrillahihsan@gmail.com

Mastura Diana Marieska*
Jurusan Teknik Informatika
Universitas Sriwijaya
Palembang, Indonesia
mastura.diana@ilkom.unsri.ac.id

***Abstrak*—Pemilihan antara lingkungan kontainer (container) dan non-kontainer (non-container) merupakan keputusan penting yang harus diambil oleh pengembang aplikasi saat mengimplementasikan sistemnya. Penelitian ini membandingkan performa aplikasi e-commerce pada lingkungan kontainer (Docker) dan non-kontainer (XAMPP) menggunakan alat load testing Grafana K6. Pengujian dilakukan untuk mengukur tingkat kesalahan (error rate) dan waktu pemrosesan (processing time) dengan variasi pengguna virtual (virtual user atau VU) dari 500 hingga 5000. Hasilnya menunjukkan lingkungan non-kontainer mengalami error rate yang tinggi (mencapai 73,14%) dan performa tidak stabil. Sebaliknya, lingkungan kontainer menunjukkan kinerja yang jauh lebih unggul dengan error rate yang rendah (maksimal 13,67%) dan processing time yang stabil. Disimpulkan bahwa kontainer lebih efektif dan andal untuk mendukung aplikasi e-commerce yang membutuhkan skalabilitas tinggi.**

Kata Kunci—Kontainer, Non-kontainer, *E-commerce, Error Rate, Processing Time*

Abstract—The choice between containerized and non-containerized environments is a critical decision for application developers during system implementation. This study compares the performance of an e-commerce application in container (Docker) and non-container (XAMPP) environments using the Grafana K6 load testing tool. The tests measured error rate and processing time with virtual user (VU) variations from 500 to 5000. The results show that the non-container environment experienced high error rates (reaching 73.14%) and unstable performance, while the container environment demonstrated significantly superior performance with low error rates (maximum 13.67%) and stable processing times. It is concluded that containers are more effective and reliable for supporting e-commerce applications that require high scalability.

Keywords—Container, Non-container, E-commerce, Error Rate, Processing Time

I. PENDAHULUAN

Pesatnya perkembangan teknologi informasi telah mendorong pertumbuhan industri *e-commerce* yang signifikan di Indonesia. Pada tahun 2023, jumlah pengguna *e-commerce* di Indonesia telah mencapai 58,63 juta orang [1]. Pertumbuhan ini menuntut penyedia layanan untuk memastikan performa aplikasi yang stabil dan andal, terutama dalam menghadapi lonjakan trafik yang tidak terduga, di mana kegagalan sistem dapat menyebabkan kerugian finansial dan hilangnya kepercayaan pengguna [2].

Dalam konteks pengembangan dan *deployment* aplikasi, terdapat dua pendekatan utama: lingkungan non-kontainer dan lingkungan kontainer. Lingkungan non-kontainer, seperti penggunaan XAMPP, menempatkan semua komponen aplikasi dalam satu mesin, sedangkan lingkungan kontainer, seperti Docker, mengemas aplikasi dan seluruh dependensinya menjadi unit terisolasi, menawarkan potensi skalabilitas dan konsistensi yang lebih baik [3].

Penelitian ini bertujuan untuk membandingkan performa aplikasi *e-commerce* pada kedua lingkungan tersebut, dengan fokus pada metrik tingkat kesalahan (*error rate*) dan waktu pemrosesan (*processing time*). Hasil dari penelitian ini diharapkan dapat memberikan wawasan teknis yang berguna bagi pengembang dan arsitek sistem dalam memilih lingkungan yang optimal untuk aplikasi *e-commerce* berskala besar.

II. METODOLOGI

Bagian ini merinci pendekatan metodis yang digunakan untuk mengevaluasi kinerja aplikasi e-commerce. Desain eksperimental disusun untuk memastikan perbandingan yang adil dan langsung antara kedua lingkungan *deployment* dengan menjaga logika aplikasi, perangkat keras, dan protokol pengujian tetap konstan.

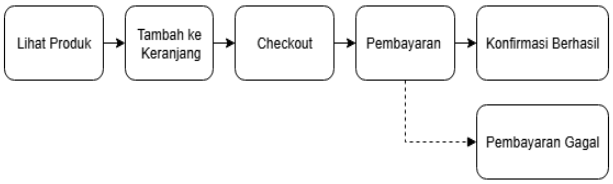
A. *E-Commerce*

E-commerce, atau perdagangan elektronik, merupakan platform digital yang memungkinkan proses jual beli barang dan jasa secara daring. Sistem ini banyak digunakan karena kemudahan akses, efisiensi, serta fleksibilitasnya [4], [5]. Melalui aplikasi berbasis web maupun mobile, e-commerce menghilangkan batasan geografis, mempercepat proses transaksi, serta meningkatkan pengalaman pengguna dengan fitur seperti pencarian produk, keranjang belanja, dan pembayaran daring. Selain itu, e-commerce juga mampu mengintegrasikan notifikasi real-time serta sistem pelacakan pesanan untuk mendukung kenyamanan pengguna.

Dalam penelitian ini, sistem e-commerce dipilih sebagai studi kasus karena memiliki volume transaksi yang tinggi dan relevan dengan kebutuhan bisnis digital saat ini. Platform e-commerce harus mampu memproses ribuan permintaan pengguna secara paralel, mulai dari penjelajahan produk, pengelolaan keranjang belanja, hingga pembayaran yang

aman. Oleh karena itu, dibutuhkan infrastruktur yang efisien dan andal untuk memastikan waktu respons rendah, tingkat kegagalan minimal, serta pengalaman pengguna yang konsisten meskipun terjadi lonjakan trafik.

Aplikasi e-commerce yang dikembangkan dalam penelitian ini mencakup fitur utama, yaitu registrasi akun, penelusuran produk, manajemen keranjang belanja, pemesanan, dan pembayaran. Gambar 1 menunjukkan alur sistem yang dirancang, dimulai dari pengguna melihat produk, menambahkannya ke keranjang, melanjutkan ke proses checkout, dan menyelesaikan pembayaran. Selain itu, sistem juga dilengkapi dengan mekanisme penanganan transaksi gagal untuk memberikan notifikasi yang sesuai kepada pengguna.



Gambar 1. Alur Sistem E-Commerce

Penelitian ini menggunakan tumpukan teknologi berbasis PHP sebagai bahasa pemrograman utama dan MySQL sebagai basis data. PHP dipilih karena fleksibilitas dan popularitasnya dalam pengembangan aplikasi web [6]. MySQL digunakan untuk mengelola data produk, pengguna, dan transaksi secara terstruktur. Sistem kemudian dijalankan pada dua lingkungan berbeda: XAMPP sebagai representasi lingkungan non-kontainer dan Docker sebagai representasi lingkungan berbasis kontainer. Untuk menguji performa, penelitian ini menggunakan Grafana K6 yang mampu melakukan simulasi pengguna secara bersamaan serta mengukur metrik performa utama [7].

B. Penerapan di Lingkungan Non-Kontainer

Pada lingkungan non-kontainer, aplikasi e-commerce dijalankan secara langsung di atas sistem operasi host menggunakan XAMPP sebagai server lokal. XAMPP terdiri atas beberapa komponen inti, yaitu Apache sebagai web server, MySQL sebagai basis data, dan PHP sebagai bahasa pemrograman utama. Arsitektur ini bersifat monolitik karena seluruh layanan aplikasi berjalan dalam satu lingkungan yang sama tanpa isolasi.

Meskipun sederhana dan mudah diimplementasikan, pendekatan ini memiliki sejumlah keterbatasan. Ketika trafik meningkat, konsumsi sumber daya menjadi tidak terkontrol karena semua komponen saling berbagi memori dan CPU pada host. Hal ini dapat menyebabkan penurunan performa dan meningkatkan risiko kegagalan layanan.

Gambar 2. Desain Sistem E-Commerce pada Lingkungan Non-Kontainer.

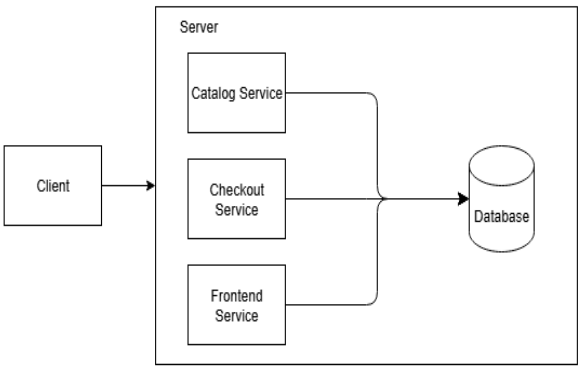
C. Penerapan Di lingkungan Kontainer

Lingkungan kontainer menggunakan Docker Compose untuk mengatur multi-service. Tidak seperti model monolitik pada XAMPP, setiap komponen aplikasi dijalankan dalam kontainer terpisah:

1. Database – menyimpan data produk, pengguna, dan transaksi.
2. Catalog Service – mengelola katalog produk.
3. Checkout Service – menangani proses pemesanan.
4. Frontend Service – menyajikan antarmuka aplikasi.

Komunikasi antar service difasilitasi oleh *Docker network*, sementara *volume* digunakan untuk menjaga data tetap persisten. Pengaturan ini didefinisikan melalui docker-compose.yml dan *Dockerfile* untuk membangun image PHP.

Pendekatan multi-service ini memberikan fleksibilitas, portabilitas, serta memudahkan replikasi. Selain itu, arsitektur kontainer lebih sesuai untuk pengujian performa karena memungkinkan kontrol penuh terhadap konfigurasi sistem dan mendukung skalabilitas [8].



Gambar 3. Desain Sistem E-Commerce pada Lingkungan Kontainer

D. Metrik performa

Pengukuran performa merupakan aspek penting dalam penelitian ini karena menjadi dasar untuk menilai perbedaan kualitas sistem pada dua lingkungan implementasi yang berbeda. Dua parameter utama yang digunakan adalah *error rate* dan *processing time*. *Error rate* didefinisikan sebagai rasio antara jumlah permintaan (*request*) yang gagal diproses dengan total permintaan yang dikirim selama pengujian. Dalam konteks aplikasi e-commerce, kegagalan dapat terjadi karena server tidak mampu merespons permintaan dalam batas waktu tertentu (*timeout*), atau karena sistem mengalami kelebihan beban yang menyebabkan sebagian *request* ditolak. Semakin tinggi nilai *error rate*, semakin menunjukkan bahwa sistem tidak sanggup menjaga kestabilan saat menerima jumlah pengguna yang besar. Sebaliknya, nilai *error rate* mendekati nol mengindikasikan ketahanan sistem yang baik terhadap lonjakan trafik [9].

Selain *error rate*, parameter lain yang digunakan adalah *processing time*. Parameter ini merujuk pada waktu yang dibutuhkan sistem untuk merespons permintaan dari

pengguna, dimulai sejak *request* dikirim hingga *response* diterima kembali. *Processing time* tidak hanya memberikan gambaran rata-rata, melainkan juga dianalisis dalam lima indikator, yaitu nilai minimum, maksimum, rata-rata (*average*), median, dan *percentile 90 (P90)*. Nilai minimum menunjukkan respon tercepat yang mampu dicapai sistem, sementara nilai maksimum mencerminkan respon terlama yang terjadi pada kondisi tertentu. Nilai rata-rata digunakan untuk menilai kinerja secara umum, median menggambarkan respon tipikal, sedangkan P90 menunjukkan distribusi waktu proses yang dialami oleh 90% pengguna. Kombinasi indikator ini sangat penting untuk memahami variasi performa sistem secara lebih mendalam.

Pemilihan kedua metrik ini didasarkan pada keterkaitannya dengan kualitas layanan (*Quality of Service*) dan pengalaman pengguna (*User Experience*). Dalam layanan e-commerce, kesuksesan transaksi bergantung pada kecepatan dan keandalan sistem. Sistem dengan *error rate* rendah dan *processing time* stabil akan memberikan pengalaman yang lebih baik kepada pengguna, terutama ketika terjadi lonjakan permintaan yang besar. Dengan demikian, metrik ini dapat menjadi indikator objektif untuk membandingkan efektivitas lingkungan kontainer dan non-kontainer.

E. Batasan Masalah

Penelitian ini menetapkan beberapa batasan yang bertujuan untuk menghindari perluasan ruang lingkup yang dapat mengurangi kedalaman analisis. Pertama, pengujian performa hanya dilakukan pada alur utama transaksi dalam sistem e-commerce. Alur ini meliputi pencarian produk, penambahan produk ke keranjang belanja, proses checkout, dan penyelesaian pembayaran. Dengan membatasi hanya pada proses inti, penelitian ini dapat lebih menekankan pada performa sistem dalam skenario yang paling krusial bagi keberhasilan layanan e-commerce.

Kedua, aplikasi e-commerce yang diuji dikembangkan menggunakan bahasa pemrograman PHP dengan basis data MySQL. Pemilihan teknologi ini didasarkan pada popularitas dan relevansinya dalam pengembangan aplikasi berbasis web, serta karena keduanya mewakili mayoritas platform yang digunakan pada e-commerce skala kecil hingga menengah. Namun, konsekuensinya adalah hasil penelitian ini hanya berlaku pada konteks aplikasi berbasis PHP dan MySQL, dan tidak serta-merta dapat digeneralisasikan ke aplikasi dengan arsitektur atau teknologi lain.

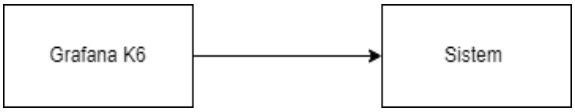
Ketiga, penelitian ini hanya membandingkan dua model implementasi, yaitu lingkungan non-kontainer menggunakan XAMPP dan lingkungan kontainer menggunakan Docker Compose. XAMPP dipilih karena mewakili pendekatan konvensional yang masih banyak digunakan, sementara Docker Compose dipilih karena mendukung pendekatan modern berbasis kontainer yang banyak diadopsi dalam pengembangan perangkat lunak saat ini. Dengan membatasi hanya pada dua platform ini, penelitian dapat memberikan perbandingan yang jelas mengenai perbedaan performa di antara keduanya.

Keempat, penelitian hanya menitikberatkan pada dua parameter performa, yaitu *error rate* dan *processing time*. Aspek lain seperti penggunaan memori, konsumsi CPU, throughput jaringan, maupun keamanan aplikasi tidak menjadi bagian dari pembahasan. Meskipun aspek tersebut juga penting dalam menilai performa aplikasi secara menyeluruh, fokus pada dua parameter utama memungkinkan penelitian ini lebih terarah dalam mengevaluasi stabilitas dan kecepatan respon sistem.

Penetapan batasan ini dilakukan secara sadar agar penelitian tetap realistis, dapat diselesaikan sesuai waktu yang tersedia, serta menghasilkan analisis yang lebih mendalam mengenai variabel yang benar-benar relevan dengan tujuan penelitian [10].

F. Konfigurasi Pengujian

Konfigurasi pengujian dalam penelitian ini dirancang untuk merepresentasikan skenario nyata yang mungkin terjadi dalam penggunaan aplikasi e-commerce. Pengujian dilakukan menggunakan Grafana K6, sebuah perangkat lunak *open-source* yang banyak digunakan dalam pengujian performa aplikasi berbasis web. K6 dipilih karena mampu mensimulasikan ribuan pengguna virtual secara bersamaan dan menyediakan hasil pengukuran dalam bentuk statistik maupun visualisasi grafik. Fleksibilitas K6 dalam mendefinisikan skenario uji berbasis skrip juga memudahkan peneliti dalam mereplikasi alur transaksi nyata pengguna+.



Gambar 4. Komponen Uji Coba

Skenario pengujian meniru alur transaksi e-commerce yang umum dilakukan, yaitu pencarian produk, penambahan produk ke keranjang belanja, proses checkout, dan penyelesaian pembayaran. Seluruh alur ini dimasukkan ke dalam skrip K6 sehingga setiap pengguna virtual (VU) yang disimulasikan akan mengeksekusi alur yang sama. Jumlah VU divariasikan secara bertahap mulai dari 500 hingga 5000, dengan peningkatan 500 VU pada setiap siklus pengujian. Variasi ini dipilih untuk melihat sejauh mana sistem dapat mempertahankan performa ketika jumlah pengguna meningkat secara signifikan.

Pengujian dijalankan pada perangkat keras dengan spesifikasi prosesor AMD Ryzen 5 4600H dengan kecepatan 3.00 GHz, RAM sebesar 16 GB, dan penyimpanan SSD berkapasitas 512 GB. Lingkungan perangkat lunak menggunakan Windows 11 64-bit dengan dukungan Docker, XAMPP, PHP, MySQL, serta Grafana K6. Konfigurasi ini dipilih untuk menyeimbangkan kebutuhan penelitian dengan ketersediaan sumber daya.

Hasil pengujian dicatat dalam bentuk log yang mencakup nilai *error rate* dan *processing time* untuk masing-masing skenario. Data yang diperoleh kemudian dianalisis dan divisualisasikan dalam bentuk tabel serta grafik perbandingan antara lingkungan non-kontainer dan

kontainer. Dengan pendekatan ini, penelitian dapat memberikan gambaran yang jelas mengenai perbedaan performa kedua lingkungan implementasi dalam menghadapi variasi jumlah pengguna.

III. HASIL DAN PEMBAHASAN

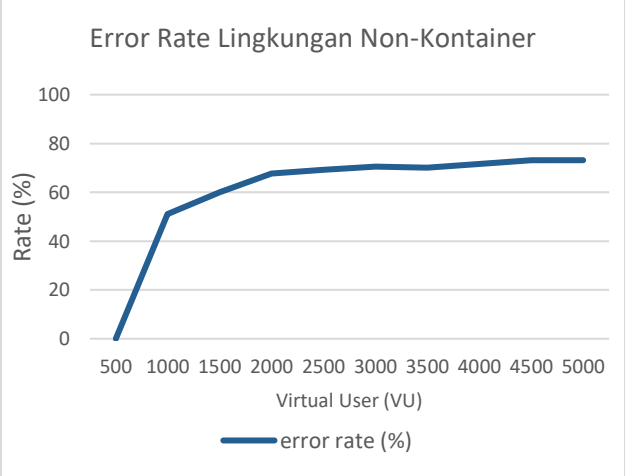
Bagian ini menyajikan data empiris yang dikumpulkan dari uji kinerja yang dilakukan pada lingkungan non-kontainer dan kontainer. Hasil disajikan secara objektif untuk memberikan dasar kuantitatif yang jelas untuk diskusi selanjutnya.

A. *Kinerja Lingkungan Non-Kontainer*

Tabel 1. Data Pengujian Lingkungan Non-Kontainer

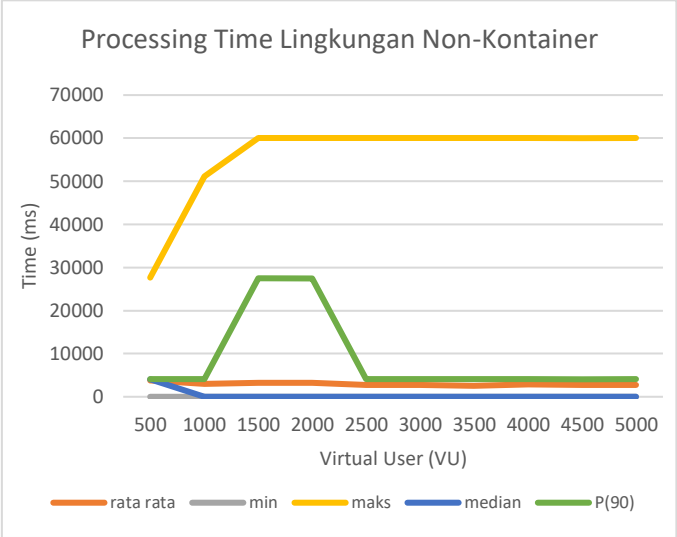
Virtual User (VU)	*Processing time* (ms)					*Error rate* (%)
	Rata-Rata	Minimal	Maksimal	Median	P(90)	
500	3787.95	0.529	27650.63	4052.22	4106.84	0
1000	2965.03	0	51118.70	0	4088.70	51.10
1500	3223.67	0	60001.65	0	27444.15	59.94
2000	3237.42	0	60001.05	0	27453.26	67.60
2500	2793.47	0	60000.79	0	4063.64	69.30
3000	2768.21	0	60000.60	0	4067.95	70.42
3500	2543.73	0	60006.93	0	4063.34	70.18
4000	2828.71	0	60006.45	0	4062.88	71.72
4500	2793.36	0	60007.08	0	4058.40	73.14
5000	2793.91	0	60000.84	0	4059.18	73.09

Aplikasi yang di-deploy menggunakan XAMPP menunjukkan degradasi kinerja dan ketidakstabilan yang signifikan seiring dengan meningkatnya beban pengguna. Pada beban dasar 500 VU, sistem berfungsi tanpa kesalahan. Namun, saat beban digandakan menjadi 1.000 VU, sistem mengalami kegagalan katastropik, dengan *error rate* melonjak tajam menjadi 51,10%. Nilai *error rate* terus meningkat seiring dengan bertambahnya beban, mencapai 73,09% pada 5.000 VU, yang menunjukkan bahwa sebagian besar permintaan pengguna gagal.



Gambar 5. Grafik Error Rate Lingkungan Non-Kontainer

Metrik *processing time* semakin menunjukkan ketidakstabilan sistem. Seperti yang ditunjukkan pada Tabel 1, *processing time* rata-rata sangat tidak menentu, berfluktuasi tanpa korelasi yang jelas dengan peningkatan beban. Indikator kritis dari keruntuhan sistem adalah median, yang turun menjadi 0 ms untuk semua pengujian mulai dari 1.000 VU ke atas. Ini menandakan bahwa lebih dari setengah permintaan gagal begitu cepat sehingga tidak mencatatkan nilai *processing time*. Hal ini merupakan indikasi bahwa server kewalahan dan menolak koneksi. Selanjutnya, *processing time* maksimum secara konsisten melebihi 60.000 ms dari 1.500 VU, menunjukkan bahwa beberapa permintaan yang tidak langsung ditolak mengalami waktu habis (*timeout*).



Gambar 6. Grafik *Processing Time* Lingkungan Non-Kontainer

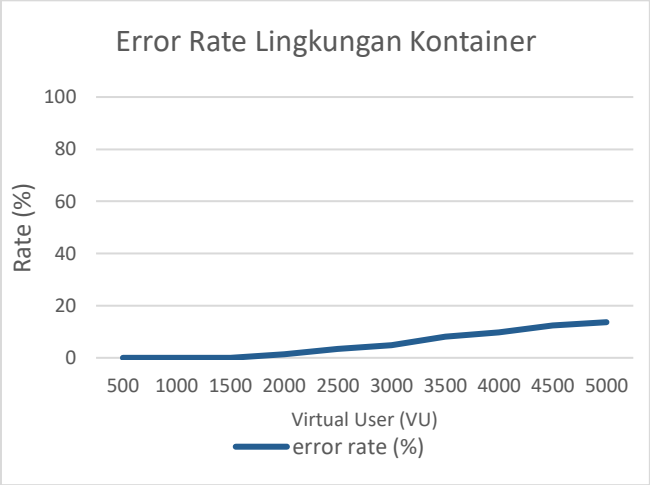
B. *Kinerja Lingkungan Kontainer*

Tabel 2. Data Pengujian Lingkungan Non-Kontainer

Virtual User (VU)	*Processing time* (ms)					*Error rate* (%)
	Rata-Rata	Minimal	Maksimal	Median	P(90)	
500	775.34	0.507	20014.61	5.44	221.06	0
1000	1454.75	0.50	39387.68	5.94	241.17	0
1500	1685.08	0	54921.84	6.43	267.50	0
2000	2717.99	0	60000.85	6.50	4753.15	1.42
2500	4317.32	0	60003.90	6.85	17425.94	3.33
3000	5457.53	0	60016.83	6.92	20356.27	4.87
3500	8591.92	0	60010.92	7.62	27724.27	8.04
4000	9690.85	0	60019.34	7.95	51646.56	9.82
4500	11525.72	0	60016.07	8.34	59999.22	12.33
5000	12374.08	0	60024.22	10.23	59999.20	13.67

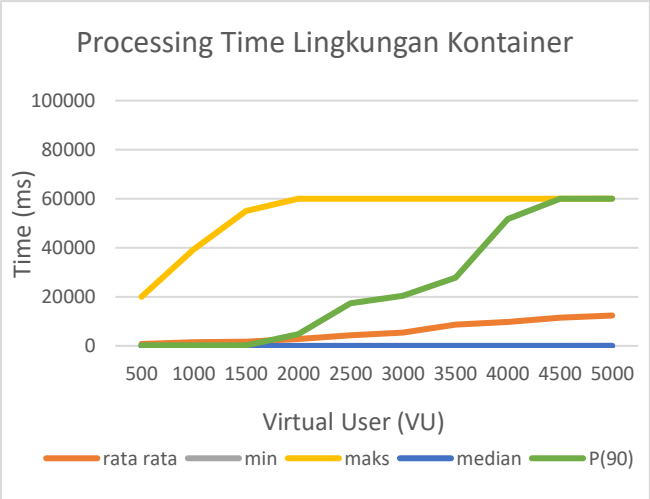
Sangat kontras, deployment berbasis Docker menunjukkan ketahanan yang superior dan pola degradasi kinerja yang terkendali di bawah beban. Sistem mempertahankan *error rate* 0% yang sempurna hingga beban

1.500 VU. Kesalahan baru mulai muncul pada 2.000 VU, dengan tingkat yang dapat diabaikan sebesar 1,42%. Bahkan pada beban maksimum yang diuji sebesar 5.000 VU, *error rate* tetap relatif rendah di 13,67%, menunjukkan bahwa sistem masih berhasil memproses sebagian besar permintaan.



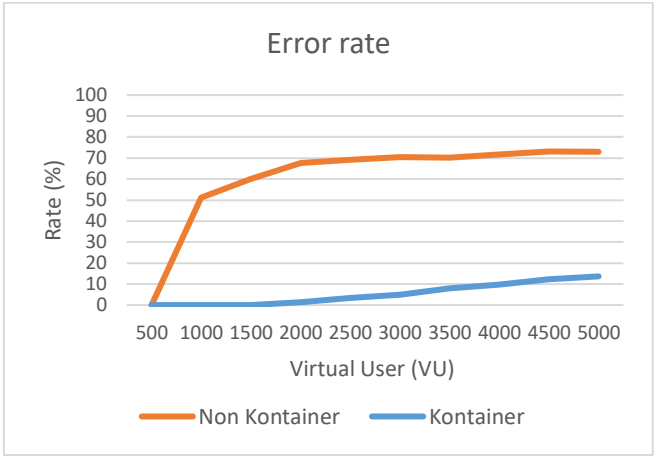
Gambar 7. Grafik Error Rate Lingkungan Non-Kontainer

Metrik *processing time* untuk lingkungan kontainer, yang dirinci dalam Tabel 2, menunjukkan profil kinerja yang dapat diprediksi dan stabil. *Average processing time* meningkat secara stabil dari 775,34 ms pada 500 VU menjadi 12.374,08 ms pada 5.000 VU. Peningkatan latensi yang konsisten dan linear ini merupakan karakteristik dari sistem yang berperilaku baik dalam mengelola sumber dayanya secara efektif di bawah tekanan yang meningkat. Nilai median dan P(90) juga menunjukkan pola peningkatan bertahap yang serupa, mengonfirmasi bahwa meskipun sistem melambat, sistem tidak runtuh. Sistem terus melayani permintaan dengan sukses, meskipun dengan latensi yang lebih tinggi, yang merupakan perilaku yang jauh lebih diinginkan untuk sistem produksi.



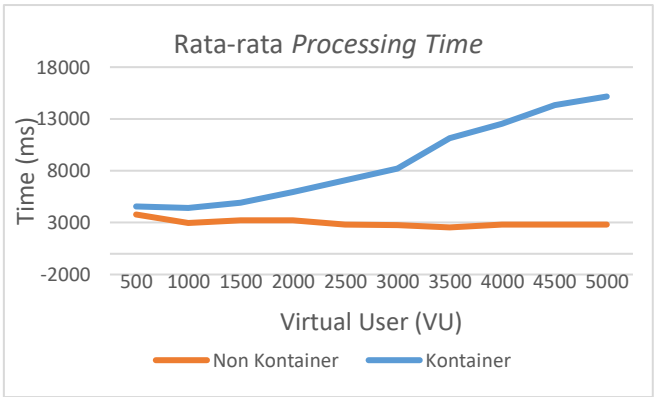
Gambar 8. Grafik *Processing Time* Lingkungan Kontainer

Perbedaan kinerja paling jelas diilustrasikan dengan membandingkan *error rate* dari kedua lingkungan. Gambar 9 memplot tingkat kesalahan terhadap jumlah pengguna virtual. Garis untuk lingkungan non-kontainer menunjukkan kenaikan yang hampir vertikal pada 1.000 VU, menandakan kegagalan sistem yang tiba-tiba. Sebaliknya, garis untuk lingkungan Docker tetap datar di 0% hingga 1.500 VU dan kemudian menunjukkan kenaikan yang landai dan terkendali. Visualisasi ini memberikan bukti kuat tentang keandalan superior dari arsitektur kontainer.



Gambar 9. Perbandingan *Error Rate*

Demikian pula, Gambar 10 membandingkan *average processing time*. Kinerja lingkungan non-kontainer tidak menentu dan tidak dapat diprediksi. Sebaliknya, lingkungan Docker menampilkan peningkatan *processing time* yang tidak signifikan dan dapat diprediksi, memperkuat kesimpulan bahwa ia menangani beban dengan cara yang lebih stabil dan mudah dikelola.



Gambar 10. Perbandingan Rata-rata Processing Time

IV. PEMBAHASAN

Hasil yang disajikan di bagian sebelumnya menunjukkan kesenjangan kinerja yang tajam dan konklusif antara lingkungan deployment kontainer dan non-kontainer. Diskusi sekarang beralih ke interpretasi temuan ini, menganalisis penyebab mendasar dari perbedaan kinerja, dan mengeksplorasi implikasi praktisnya bagi arsitektur sistem dan operasi bisnis.

Temuan utama adalah bahwa arsitektur kontainer tidak hanya lebih cepat; namun juga secara fundamental lebih tangguh. Kedua sistem menunjukkan perilaku yang sama sekali berbeda di bawah tekanan. Lingkungan non-kontainer mengalami apa yang dapat disebut sebagai kegagalan. Ketidakmampuannya menangani lebih dari 500 pengguna

bersamaan, yang dibuktikan dengan ledakan *error rate* dan median *processing time* 0 ms, menunjukkan bahwa sistem menjadi begitu kewalahan sehingga berhenti berfungsi secara efektif, secara aktif menolak sebagian besar permintaan yang masuk. Jenis mode kegagalan yang rapuh ini tidak dapat diterima untuk aplikasi bisnis-kritis mana pun, karena ini secara langsung berarti kehilangan pendapatan dan pengalaman pengguna yang sangat menurun.

Sebaliknya, lingkungan kontainer menunjukkan degradasi kinerja yang terkendali (*graceful degradation*). Seiring meningkatnya beban, waktu respons sistem melambat, tetapi terus berhasil memproses persentase permintaan yang tinggi. Perilaku ini jauh lebih diinginkan dalam lingkungan produksi. Transaksi yang membutuhkan beberapa detik lebih lama untuk diselesaikan jauh lebih baik daripada transaksi yang gagal sama sekali. Ini menunjukkan bahwa sistem kontainer secara efektif mengelola sumber dayanya dan mengantrekan permintaan, memprioritaskan penyelesaian yang berhasil daripada respons instan saat berada di bawah tekanan. Karakteristik "*fail-safe*" ini adalah ciri khas dari sistem yang kuat dan andal.

Perbedaan signifikan dalam keandalan ini dapat dikaitkan dengan keunggulan arsitektur inti dari kontainerisasi. Docker, dengan memanfaatkan fitur-fitur dari kernel Linux yang mendasarinya, menyediakan manajemen sumber daya yang superior. Setiap kontainer berjalan di lingkungan yang terisolasi dengan alokasi CPU, memori, dan sumber daya jaringannya sendiri. Isolasi ini mencegah satu proses atau komponen yang kelebihan beban mengonsumsi semua sumber daya sistem yang tersedia dan menyebabkan kegagalan di seluruh sistem, sebuah kerentanan umum di lingkungan monolitik seperti XAMPP yang berjalan pada sistem operasi serba guna. Model proses dan jaringan dalam ekosistem kontainer secara inheren dioptimalkan untuk beban kerja multi-tenant dengan konkurensi tinggi yang khas dari aplikasi server, yang mengarah pada kinerja yang lebih efisien dan stabil.

Implikasi praktis dari temuan ini sangat mendalam. Untuk platform e-commerce atau aplikasi web apa pun yang mengantisipasi lalu lintas yang bervariasi, tinggi, atau tidak dapat diprediksi, melakukan *deployment* pada arsitektur tradisional non-kontainer menimbulkan risiko bisnis yang signifikan. Namun, arsitektur berbasis kontainer memberikan ketahanan dan skalabilitas mendasar yang diperlukan untuk memastikan ketersediaan tinggi. Di ranah e-commerce, dimana waktu aktif dan responsivitas sistem berkorelasi langsung dengan kepuasan pelanggan dan pendapatan, pilihan arsitektur ini menjadi pendorong penting bagi kesuksesan bisnis.

V. KESIMPULAN

Studi ini melakukan perbandingan empiris kinerja aplikasi e-commerce di lingkungan kontainer (Docker) dan non-kontainer (XAMPP). Bukti yang ada sangat mendukung kesimpulan bahwa arsitektur *deployment* berbasis kontainer menawarkan kinerja, stabilitas, dan ketahanan yang jauh lebih superior untuk aplikasi web di bawah beban berat.

Temuan kuantitatif utama menggarisbawahi kesimpulan ini. Sistem kontainer berhasil menangani beban hingga 2.000 pengguna virtual dengan *error rate* mendekati nol dan mempertahankan *error rate* yang dapat dikelola sebesar 13,67% pada 5.000 VU. Sebaliknya, sistem non-kontainer mengalami keruntuhan katastropik hanya pada 1.000 VU, dengan *error rate* melebihi 73% pada beban puncak. Meskipun lingkungan non-kontainer sedikit lebih cepat pada beban yang sangat rendah, kinerjanya menjadi tidak menentu dan tidak dapat diandalkan seiring meningkatnya trafik, sedangkan sistem kontainer menunjukkan degradasi yang dapat diprediksi dan terkendali.

Penelitian ini memposisikan kontainerisasi bukan hanya sebagai opsi deployment alternatif, tetapi sebagai teknologi penting untuk membangun sistem e-commerce modern yang skalabel dan andal yang mampu memenuhi tuntutan ekonomi digital. Bagi pengembang atau organisasi yang berencana untuk bermigrasi ke arsitektur kontainer, disarankan untuk memprioritaskan modularisasi aplikasi. Sebaiknya aplikasi dipecah menjadi layanan-layanan mikro (*microservices*) yang lebih kecil dan terisolasi. Hal ini penting untuk memaksimalkan manfaat isolasi sumber daya dan skalabilitas horizontal yang ditawarkan oleh kontainer, sehingga proses migrasi dapat berjalan lancar dan menghasilkan peningkatan performa yang optimal.

Untuk penelitian selanjutnya, disarankan dua jalur utama untuk membangun temuan ini. Pertama, akan bermanfaat untuk melakukan perbandingan kinerja serupa menggunakan tumpukan non-kontainer lain, seperti Nginx atau Apache mandiri, untuk menentukan apakah kegagalan yang diamati spesifik untuk XAMPP atau merupakan karakteristik arsitektur non-kontainer secara lebih luas. Kedua, penelitian di masa depan harus menggunakan aplikasi e-commerce dengan kompleksitas yang lebih besar, menggabungkan fitur-fitur seperti manajemen inventaris waktu nyata, integrasi gateway pembayaran pihak ketiga, dan kueri basis data yang lebih kompleks. Menguji sistem semacam itu akan memberikan tolok ukur kinerja yang lebih representatif dalam kondisi produksi dunia nyata.

REFERENCES

[1] A. Solehudin, U. S. Karawang, and T. Timur, “Analisis Perbandingan Performa Server Websocket Dengan Menggunakan Payload JSON , Binary Serialization , dan Protobuf dengan Menggunakan Metode Load Testing,” vol. 9, no. 1, pp. 706–712, 2025.

[2] S. Suhari, A. Faqih, and F. M. Basysyar, “Human Resources Information System Using Agile Development Method at CV. Angkasa Raya,” *J. Teknol. dan Inf.*, vol. 12, no. 1, pp. 30–45, 2022, doi: 10.34010/jati.v12i1.

[3] H. T. Tasik, S. Paembonan, and M. Muhallim, “Penjualan Kursi Dan Meja Jepara Pada Toko Arlan Mebel Berbasis Mobile Di Luwu Utara,” *J. Inform. dan Tek. Elektro Terap.*, vol. 12, no. 3S1, 2024, doi: 10.23960/jitet.v12i3s1.5163.

[4] W. M. Sari *et al.*, “Penerapan E-Commerce

Menggunakan Metode Extreme," vol. 05, no. 02, pp. 136–144, 2020.

[5] H. H. Halimah *et al.*, "Perancangan Sistem E-Commerce Berbasis Web," vol. 8, no. 3, pp. 3380–3386, 2024.

[6] Rina Noviana, "Pembuatan Aplikasi Penjualan Berbasis Web Monja Store Menggunakan Php Dan Mysql," *J. Tek. dan Sci.*, vol. 1, no. 2, pp. 112–124, 2022, doi: 10.56127/jts.v1i2.128.

[7] D. Y. Kristiyanto and A. Schmidt, "Performance Evaluation of Monolithic Architectural Design in Online Ornamental Plant Sales Platform," *2024 3rd Int. Conf. Creat. Commun. Innov. Technol.*, pp. 1–8, 2024, doi: 10.1109/ICCIT62134.2024.10701232.

[8] H. Aqasizade, E. Ataie, and M. Bastam, "Experimental Assessment of Containers Running on Top of Virtual Machines," 2024, [Online]. Available: http://arxiv.org/abs/2401.07539

[9] S. E. Prasetyo and Benny, "Analisis perbandingan performa virtualisasi berbasis container dengan virtualisasi berbasis hypervisor," *Pharmacogn. Mag.*, vol. 75, no. 17, pp. 399–405, 2021.

[10] J. G. Mulyana, W. S. Raharjo, and G. Indriyanta, "Studi Komparasi Performa NGINX dan HAPROXY Sebagai Load Balancer di Cloud Menggunakan Teknologi Kontainer," *J. Terap. Teknol. Inf.*, vol. 5, no. 1, pp. 11–17, 2021, doi: 10.21460/jutei.2021.51.229.