Laman web jurnal: https://ejournal.unama.ac.id/index.php/processor

Jurnal Processor

P-ISSN: 1907-6738 | E-ISSN: 2528-0082

Analisis Performa, Overload, dan Kerentanan pada Website Basarnas Bengkulu untuk Optimalisasi Kinerja

Julyane Kevin Cheka[1], Rozali Toyib[2*], Ardi Wijaya[3], Muntahanah[4]

[1,2,3,4]Program Studi Teknik Informatika, Fakultas Teknik, Universitas Muhammadiyah Bengkulu, Jl. Bali, Kampung Bali, Kec. Teluk Segara, Kota Bengkulu, Bengkulu 38119, Indonesia.
*Penulis Korespondensi, Email: rozalitoyib@umb.ac.id

Abstrak− Website Basarnas Bengkulu mempunyai peran krusial dalam memberikan informasi berhubungan dengan kegiatan pencarian dan pertolongan kepada masyarakat. Namun seiring bertambahnya jumlah pengunjung serta kebutuhan akan informasi yang cepat dan akurat, situs ini menghadapi beberapa tantangan terkait performa, overload, dan kerentanan seperti waktu muat halaman yang lambat, website yang cendrung tidak responsif, serta indikasi defacement yang tentu akan merugikan. Maka dilakukan penelitian dengan tujuan untuk menganalisis serta memberikan rekomendasi untuk optimalisasi kinerja website. Mengimplementasikan metode Software Testing Life Cycle (STLC) yaitu metode yang digunakan untuk pengujian perangkat lunak serta dibantu dengan tools seperti PageSpeed Insights, Apache Jmeter, dan Acunetix memungkinkan penelitian dilakukan secara komprehensif dan efisien. Sehingga didapatkan hasil penelitian pada performa dengan nilai rata-rata 81,87 yang termasuk dalam range baik namun tetap perlu perbaikan. Pada overload didapatkan hasil penelitian yaitu skenario 1 dengan 5 user (105 sampel) terdapat 0.00% error, skenario 2 dengan 15 user (315 sampel) terdapat 0.32% error, serta skenario 3 dengan 45 user (945 sampel) terdapat 1.90% yang secara keseluruhan menunjukkan kinerja website masih cukup baik pada beban rendah namun menurun pada beban tinggi. Dan pada kerentanan didapatkan hasil penelitian pada kategori Low severity vulnerabilities yang artinya terdapat kerentanan minor dan masih relatif aman namun tetap perlu ditindaklanjuti untuk upaya pencegahan. Berdasarkan temuan ini direkomendasikan untuk memperhatikan infrastruktur server, mengoptimasi konten atau gambar, serta meningkatkan dan memperhatikan keamanan sistem agar dapat meminimalisir kerentanan untuk mencegah ancaman dan serangan yang dapat mengganggu kinerja website.

Kata Kunci: Kerentanan; Overload; Performa; Tools; Website.

Abstract−The Bengkulu Basarnas website plays a crucial role in providing information related to search and rescue activities to the public. However, with the increasing number of visitors and the need for fast and accurate information, the site faces several challenges related to performance, overload, and vulnerabilities such as slow page load times, a website that tends to be unresponsive, and indications of defacement, which are certainly detrimental. Therefore, research was conducted with the aim of analyzing and providing recommendations for optimizing website performance. Implementing the Software Testing Life Cycle (STLC) method, a method used for software testing, and assisted by tools such as PageSpeed Insights, Apache Jmeter, and Acunetix, enabled the research to be conducted comprehensively and efficiently. The results obtained an average performance score of 81.87, which is included in the good range but still needs improvement. On overload, the research results were obtained, namely scenario 1 with 5 users (105 samples) there was 0.00% error, scenario 2 with 15 users (315 samples) there was 0.32% error, and scenario 3 with 45 users (945 samples) there was 1.90% which overall shows that website performance is still quite good at low loads but decreases at high loads. And on vulnerabilities, the research results were in the Low severity vulnerabilities category, which means there are minor vulnerabilities and are still relatively safe but still need to be followed up for prevention efforts. Based on these findings, it is recommended to pay attention to server infrastructure, optimize content or images, and improve and pay attention to system security in order to minimize vulnerabilities to prevent threats and attacks that can disrupt website performance.

Keywords: Vurnerability; Overload; Performance; Tools; Website.

1. PENDAHULUAN

Badan Nasional Pencarian dan Pertolongan (BASARNAS) yang merupakan lembaga pemerintah *non*-kementerian dibawah presiden juga memanfaatkan *website* sebagai sarana dalam menyampaikan layanan informasi kepada masyarakat. Hal ini sejalan dengan Peraturan Presiden No. 95 Tahun 2018 tentang Sistem Pemerintahan Berbasis Elektronik (SPBE) yang bertujuan mewujudkan proses kerja yang lebih efisien, efektif, transparan, akuntabel serta dapat meningkatkan kualitas pelayanan publik, dalam penyebaran informasi menggunakan sarana website hal-hal yang berhubungan dengan bencana alam dan info-info yang lainnya agar masyarakat bisa mengakses berita-berita khususnya yang terjadi di Provinsi Bengkulu dan apabila kita menilai kondisi hingga saat ini kegunaan *website* sangat bermanfaat, terlebih tidak semua orang dapat memperoleh informasi dengan cara berinteraksi secara langsung baik dikarenakan jarak, maupun keterbatasan yang dimiliki

https://doi.org/10.33998/processor.2025.20.2.2507

Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025

oleh seseorang seperti contoh seorang penyandang disabilitas Hal ini tentu akan merugikan mereka dalam hak memperoleh informasi dalam menunjang keberlangsungan hidupnya [1]. *Website* adalah sekumpulan halaman yang terletak dalam satu *domain* yang menawarkan beragam informasi yang dapat dibaca dan dilihat oleh pengguna internet. Informasi ini mencakup dokumen multimedia seperti teks, gambar, suara, animasi, dan video yang diakses melalui *Hypertext Transfer Protocol* (HTTP) menggunakan *browser* [2].

Permasalahan dalam pengelolaan *website* Basarnas Bengkulu ditemukan beberapa permasalahan seperti waktu muat *website* yang lambat, *website* yang kadang tidak responsif, maupun adanya indikasi *defacement* yang dapat mengakibatkan kerugian baik materil ataupun *non*-materil bagi instansi. Maka penulis berniat melakukan analisis yaitu merujuk pada proses penguraian suatu system informasi menjadi bagian-bagain komponennya yang mendalam mengenai performa yaitu kecepatan dan responsivitas sebuah situs web dalam menyajikan konten kepada penggunannya, kemudian *overload* yaitu situasi dimana sebuah situs web menerima lalu lintas atau permintaan yang melebihi kapasitas yang dapat ditangani oleh infrastruktur servernya, serta kerentanan yaitu potensi resiko bagi suatu sistem pada website Basarnas Bengkulu untuk optimalisasi kinerja [3]-[4]-[5]-[6].

Dalam penelitian ini penulis merujuk beberapa referensi sebagai pedoman dalam pelaksanaan penelitianya. Adapun penelitian terdahulu yang berjudul "Pengembangan *Test Script* untuk *Load Testing* Web dengan metode *Software Testing Life Cycle*" menjelaskan bahwa dengan mengikuti alur STLC pengembang memperoleh informasi menyeluruh mengenai situs web yang diuji sehingga mereka dapat merancang skrip pengujian yang mencakup pengujian beban pada fungsi krusial seperti operasi CRUD yang digunakan oleh pengguna. Dan hasil dari penelitian ini menghasilkan sebuah skrip uji yang digunakan untuk mengevaluasi kinerja *website* menggunakan Apache JMeter [7].

Penelitian terdahulu lainnya berjudul "Optimasi *Website* Toko Kerja Menggunakan Uji Performa Google Pagespeed Insights" penelitian ini memanfaatkan data kuantitatif yang diambil dari hasil analisis yang dilakukan oleh Google PageSpeed Insights. Hasil analisis ini menampilkan skor kinerja situs dalam bentuk persentase serta waktu akses situs dalam satuan waktu yang jelas [8]. *Website* Invitees Menggunakan Metode *Load Testing* Dengan Apache Jmete menjelaskan bahwa dalam penelitian ini penulis melakukan *Load Testing* menggunakan Apache JMeter pada situs web Invitees. Hasil penelitian menunjukkan bahwa situs web Invitees telah beroperasi secara optimal saat digunakan, namun masih memerlukan perbaikan ketika layanan diakses oleh lebih dari 10 pengguna secara bersamaan [9]. Analisis Kerentanan Pada *Website* Servio Menggunakan Acunetix Web Vulnerability hasil dari penelitian ini mengidentifikasi beberapa kerentanan pada website Servio, seperti adanya HTML yang tidak dilindungi CSRF, rentan terhadap *clickjacking*, serta beberapa *alert* informasi yang perlu diperhatikan. Temuan dari Acunetix ini menunjukkan bahwa tingkat kerentanan tersebut berada pada level medium yang menyiratkan bahwa masalah ini disebabkan oleh kesalahan konfigurasi dan kelemahan dalam kode situs [10]. Analisis Keamanan Web *New* Kuta Golf Menggunakan Metode *Vulnerability Assessments* Dan Perhitungan *Security Metriks* menjelaskan bahwa dalam melakukan pengujian keamanan website, salah satu metode yang dapat digunakan adalah *Vulnerability Assessment* dengan bantuan aplikasi Acunetix. Berdasarkan hasil *vulnerability assessment* teridentifikasi bahwa tingkat kerentanannya bernilai tinggi [11].

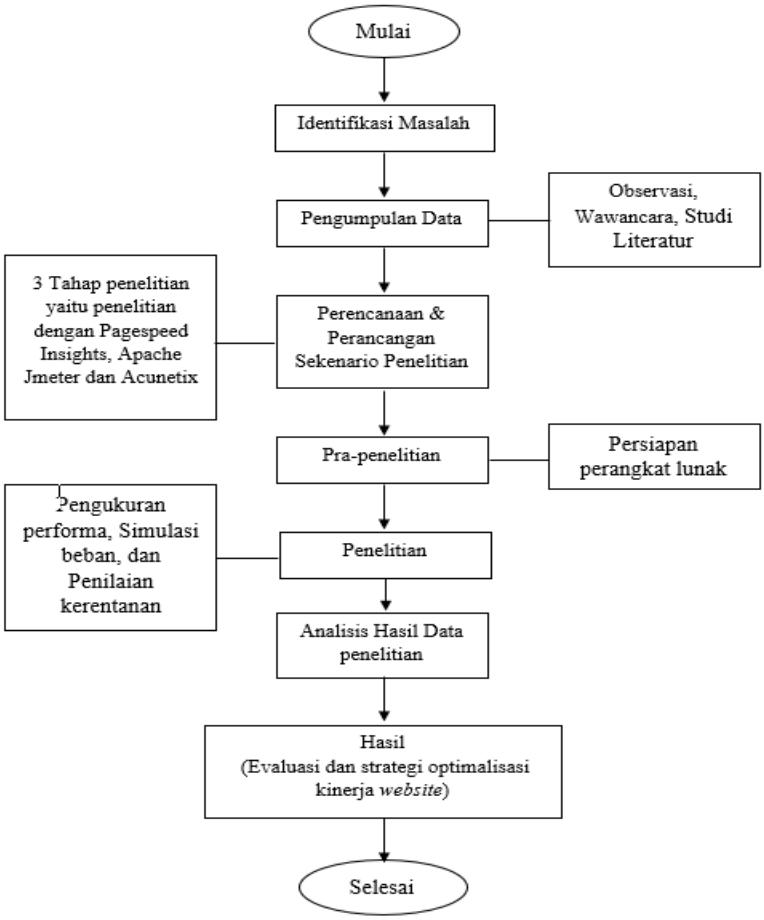
Sehingga dalam penelitian kali ini, peneliti menerapkan metode *Software Testing Life Cycle* (STLC) yang merupakan pendekatan umum dalam pengujian perangkat lunak [12]. Hal ini juga didukung dengan pemanfaatan *tools* yang relevan seperti PageSpeed Insights yaitu alat yang digunakan untuk mengukur kecepatan loading sebuah blog atau situs web [13]. Apache Jmeter yaitu sebuah proyek *open source* yang dikembangkan menggunakan bahasa Java dan berfungsi sebagai alat untuk pengujian beban [14]. Serta Acunetix yaitu perangkat lunak yang dirancang untuk melakukan pengujian keamanan secara otomatis [15]. Dengan penggunaan *tools* ini memungkinkan penelitian dapat dilakukan secara komprehensif dan efisien.

Perbedaan penelitian ini dengan penelitian terdahulu yaitu terletak pada keragaman skenario penelitian dalam alur metode STLC yang disesuaikan pada kebutuhan dari penelitian yaitu performa, overload, dan kerentanan yang dilakukan dengan memanfaatkan *tools* seperti PageSpeed Insights, Apache Jmeter, Acunetix. Selain itu manfaat penelitian ini yaitu merumuskan strategi optimalisasi untuk meningkatkan kinerja *website* dengan fokus yang mendalam untuk memberikan kontribusi signifikan terhadap pengembangan sistem informasi publik yang lebih baik, adaptif, dan responsif.

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025

2. METODOLOGI PENELITIAN

2.1 Kerangka Kerja Penelitian (Research Framework)



Gambar 1. Kerangka Kerja Penelitian (*Research Framework*)

2.2 Metode Pengembangan Sistem

Siklus Hidup Pengujian Perangkat Lunak (STLC) adalah serangkaian langkah yang teratur dalam pengujian perangkat lunak. Proses STLC merupakan komponen dari siklus hidup pengembangan perangkat lunak (SDLC) dan dilakukan pada fase pengujian. STLC mengacu pada proses pengujian yang mencakup langkah-langkah tertentu yang harus dilalui untuk menjamin bahwa standar kualitas tercapai sesuai dengan harapan. Setiap langkah memiliki kriteria dan hasil yang telah ditentukan sebelumnya.[16], yaitu :

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025

Source Picture: indiumsoftware.com/blog/software-testing-life-cycle/

Gambar 2. Alur Metode *Software Testing Life Cycle* (STLC)

a. *Requirement Analysis*
Analisis Kebutuhan adalah langkah pertama dalam STLC. *Di tahap ini, seorang Software Quality Assurance (SQA) akan menganalisis kebutuhan fungsional dan non-fungsional dari perspektif pengujian untuk menemukan kebutuhan yang dapat diuji. Tahap ini mendukung dalam menentukan ruang lingkup pengujian. Software Quality Assurance (SQA) juga akan berinteraksi dengan semua pihak terkait seperti System Analyst, Product Owner, Business Analyst, End Client, dan pihak lain untuk memastikan pengujian sesuai dengan spesifikasi klien.*

b. *Test Planning*
Test Planning merupakan tahapan kedua dari proses STLC. Informasi yang telah dikumpulkan selama tahap *Requirement Analysis* sebelumnya akan digunakan untuk menyiapkan perencanaan pengujian. SQA akan menyiapkan *test plan*, menentukan *tools* yang digunakan selama proses pengujian berlangsung, memperkirakan waktu dan sumber daya, dan menentukan siapa saja yang terlibat selama proses pengujian berlangsung.

c. *Test Case Development*
Tahapan ketiga adalah Pengembangan Kasus Uji, pada langkah ini, kasus uji akan disusun sesuai dengan rencana uji yang telah dirancang sebelumnya. Skrip pengujian otomatis juga akan disusun berdasarkan kasus uji serta membuat data uji.

d. *Environment Setup*
Sampai di titik ini, Jaminan Kualitas Perangkat Lunak (SQA) akan mempersiapkan tahap pengaturan lingkungan yang sesuai dengan jadwal, termasuk perangkat keras dan perangkat lunak yang akan digunakan, serta menyiapkan lingkungan pengujian dan data untuk pengujian. Tahap ini juga dapat dilaksanakan bersamaan dengan proses pembuatan kasus uji.*Test Execution.*

e. *Test Executio**n***
Tahap berikutnya adalah Pelaksanaan Uji, di mana SQA akan melaksanakan rencana pengujian dan kasus uji yang sudah disiapkan sebelumnya. Jika ada fitur yang tidak berfungsi sesuai dengan harapan yang diinginkan, maka hal itu akan dicatat dan dimasukkan ke dalam laporan bug. Kemudian, SQA akan menginformasikan permasalahan tersebut kepada tim pengembang untuk memperbaiki fitur itu dan akan diuji kembali oleh SQA.

f. Test Cycle Closure
Pada fase terakhir dari Siklus Hidup Pengujian Perangkat Lunak (STLC), jaminan kualitas perangkat lunak (SQA) akan menyelesaikan proses pengujian dengan menyusun laporan tentang hasil pengujian yang telah dilakukan, mengumpulkan informasi terkait hasil pengujian, serta melakukan penilaian dan analisis terhadap pengujian yang telah berlangsung. Tujuan dari fase ini adalah untuk meminimalkan kendala dalam proses pengujian yang akan datang dan meningkatkan mutu STLC di masa depan.

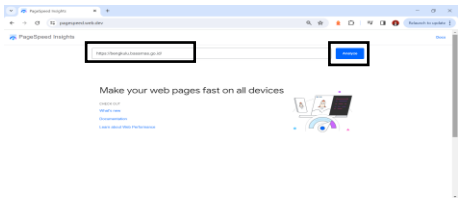
3. HASIL DAN PEMBAHASAN

3.1 Hasil

Penelitian ini dilakukan di Kantor Pencarian dan Pertolongan Bengkulu. Penelitian ini dilakukan menjadi beberapa tahap yaitu performa dengan *tools* Pagespeed Insights, *Overload* dengan *tools* Apache Jmeter, dan kerentanan dengan *tools* Acuneix.

a. Pagespeed Insights
Pada pengukuran performa peneliti menggunakan Pagespeed Insights dalam mengukur kinerja performa *website*. Adapun PageSpeed Insights merupakan *software testing* berbasis *website* yang diakses dengan browser dan internet. Langkah yang harus dilaksanakan dalam melakukan pengukuran performa yaitu:
1. Membuka halaman *website* PageSpeed Insights melalui *browser* dengan alamat https://pagespeed.web.dev/.
2. Salin alamat *website* Basarnas Bengkulu dan kemudian input kedalam *form input box* atau kotak "*Enter a web page URL*" pada halaman *page* PageSpeed Insight dan memilih kotak tindakan *Analyze* di gambar 3.

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025



Gambar 3. *Form Input Box*

3. Tunggu beberapa saat, PageSpeed Insights kemudian akan menganalisis *website* yang sudah diinputkan. Kemudian peneliti akan dihubungkan pada halaman yang berisikan hasil analisis dari pengukuran performa. Dan lakukan hal ini berulang sesuai dengan kebutuhan di gambar 4.

Gambar 4. Hasil Pengukuran Performa

Adapun rangkuman pengukuran performa yaitu pada *average* atau rata-rata nilai pada performa *website* Basarnas Bengkulu didapatkan skor 81,67 yang artinya sudah cukup baik untuk digunakan, kemudan untuk penilaian yang lebih mendalam dijelaskan pada FCP didapatkan skor 0,95 s (sangat baik), pada *speed index* didapat skor 3,48 s (perlu perbaikan *rendering*), pada TBT didapat skor 94,29 ms (baik), LCP didapatkan skor 1,79 s (baik) dan CLS didapatkan skor 0,003 (sangat baik). Untuk lebih jelasnya terdapat dalam tabel1 berikut.

Tabel 1. Hasil Pengukuran performa dengan Pagespeed Insights

Halaman	Performance	FCP	Speed Index	TBT	LCP	CLS
Beranda	72	1.0 s	4.2 s	190 ms	2.3 s	0.007
Sejarah Basarnas	80	0,9 s	4.3 s	30 ms	2.1 s	0
Arti Lambang	81	1.0 s	4.2 s	30 ms	1,9 s	0
Visi & Misi	80	0.9 s	3.2 s	60 ms	2.3 s	0
Tugas & Fungsi	83	0.9 s	3.0 s	40 ms	1.9 s	0
Struktur Organisasi	81	0.9 s	3.6 s	160 ms	1.6 s	0
Peraturan & Hukum	83	0.9 s	4.0 s	40 ms	1.7 s	0
Kerja Sama	77	1.0 s	4.1 s	180 ms	1.8 s	0
Kantor Pencarian & Pertolongabn	83	1.0 s	3.6 s	30 ms	1.8 s	0
Pelatihan Potensi Pencarian dan Pertolongan	85	0.9 s	3.3 s	40 ms	1.6 s	0.005
Sertifikasi Pencarian & Pertolongan	85	0.9 s	2.8 s	130 ms	1.6 s	0.008
Pemasyarakatan	83	1.0 s	2.7 s	150 ms	1.7 s	0.005
Sarana SAR Laut	86	1.0 s	3.0 s	50 ms	1.6 s	0.005
Sarana SAR Darat	83	0.9 s	3.1 s	90 ms	1.9 s	0.005
Sarana SAR Udara	82	1.0 s	4.1 s	100 ms	1.7 s	0.005
Siaga	85	1.0 s	2.9 s	30 ms	1.7 s	0.005
Latihan	71	1.0 s	4.1 s	270 ms	1.8 s	0.005
Standar Operasional Prosedur Pencarin & Pertolongan	82	1.0 s	4.0 s	100 ms	1.7 s	0.005
Sistem Komunikasi	87	0.9 s	2.5 s	30 ms	1.7 a	0.006
Peralatan Komunikasi	84	0.9 s	3.3 s	90 ms	1.7 s	0.005

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025

Hubungi Kami	82	1.0 s	3.1 s	140 ms	1.6 s	0.002
Rata-Rata	**81.67**	**0.95 s**	**3.48 s**	**94.28**	**1.79 s**	**0.003**

b. Apache Jmeter

Pada *overload* penelitian dilakukan secara bertahap menggunakan teknik *step load* yaitu meningkatkan jumlah *user* secara berkala. Skala yang akan digunakan peneliti adalah kelipatan 3 per-skenario yang dalam hal ini dimaksudkan agar dapat lebih efektif dalam melihat kinerja dari sistem dalam kondisi *user* yang bervariasi dan bertahap. Penentuan jumlah *thread* yang digunakan sebaiknya diambil dari data pengunjung pada *website*. Oleh karena itu disini peneliti menggunakan *tools* ubersugges untuk melihat perkiraan trafik data pengunjung pada *website* Basarnas Bengkulu. Ubersuggest sendiri adalah sebuah *tools* yang biasa digunakan untuk melihat daftar trafik pengunjung sebuah *website*. Dan berikut adalah data hasil pengunjung dari *website* Basarnas Bengkulu per 3 (tiga) bulan terakhirpad table 2:

Tabel 2. Data pengunjung *website* Basarnas Bengkulu

Bulan	Jumlah
Januari 2025	148
Februari 2025	149
Maret 2025	145
Total	442

Berdasarkan perkiraan tabel diatas yang mewakili jumlah pengunjung *website* Basarnas Bengkulu pada umumnya. Diperoleh perkiraan data user perharinya yaitu sebagai berikut:

$$\text{Jumlah User perhari} = \frac{\text{Jumlah user periode Januari – Maret}}{\text{Jumlah Bulan * Jumlah Hari}} \quad (1)$$

$$\text{Jumlah User Perhari} = \frac{442}{3*30} = 4.9111\ldots, \quad (2)$$

Dari persamaan diatas didapatkan bahwa perkiraan jumlah *user* perhari sebanyak 4.9111, atau apabila kita bulatkan menjadi 5 *user*, sehingga peneliti membuat skenario pertama yang mewakili penggunaan layanan *website* Basarnas Bengkulu pada saat kondisi normal didapat jumlah usernya 5 *user*.

Berdasarkan tabel data pengunjung diatas juga apabila kita simulasikan mewakili jumlah maksimal user pada saat layanan digunakan, diperoleh data user sebeagi berikut:

$$\text{Jumlah User Perhari} = \frac{\text{Jumlah user periode Januari – Maret}}{30 \text{ hari}} \quad (3)$$

$$\text{Jumlah User Perhari} = \frac{442}{30} = 14.7334\ldots, \quad (4)$$

Sehingga didapat persamaan bahwa jumlah *user* maksimalnya adalah 14.7334, yang apabila peneliti bulatkan menjadi 15 *user*. Oleh karena ini juga didapat skala yang digunakan yaitu 3 kali dari skenario pertama maka ini akan mewakili jumlah *user* pada saat kondisi layanan digunakan untuk skenario kedua.

Sedangkan untuk skenario ketiga, peneliti akan menggunakan skala 3 kali sama seperti skenario sebelumnya, sehingga didapatkan jumlah user untuk skenario ketiga yakni 45 *user pada* tabel 3.

Tabel 3. Skenario Pengujian

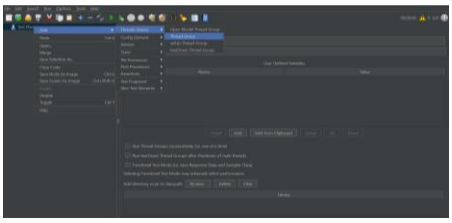
Skenario	Keterangan	Jumlah
1	Number of Threads	5
	Ramp-up period	1
	Loop Count	1
2	Number of Threads	15
	Ramp-up period	1
	Loop Count	1
3	Number of Threads	45
	Ramp-up period	1

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025

Loop Count	1

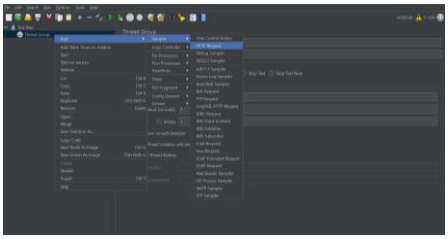
Penjelasannya yaitu pada skenario 1 ini mensimulasikan waktu muat *website* pada kondisi di mana layanan normal (*Baseline Testing*) berdasarkan data pengguna harian dari bulan Januari sampai Maret. Pada skenario 2 mensimulasikan waktu muat *website* saat layanan sedang digunakan dalam kondisi puncak (*Peak Testing*). Dan terakhir skenario 3 mensimulasikan waktu muat sistem dalam kondisi penggunaan maksimum (*Stress Testing*).
Sehingga pada simulasi beban dengan menggunakan *tools* Apache Jmeter. Langkah penelitian yang dilakukan sebagai berikut:

1. Pembuatan Thread Group penelitian. Dengan cara klik kanan *Test Plan* pada jendela Apache Jmeter, kemudian pilih *Add*, kemudian pilih *Threads (Users)*, kemudian klik *Thread Group* di gambar 5.



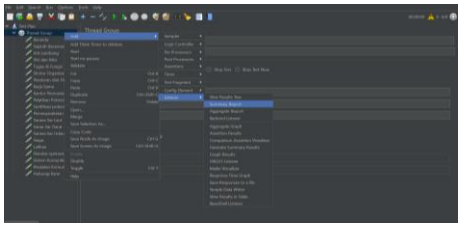
Gambar 5. Pembuatan Thread Group

2. Kemudian akan muncul halaman *Thread Group.* Selanjutnya konfigurasi sesuai skenario penelitian yang telah ditentukan sebelumnya pada jumlah user yang akan di simulasikan. Pada tahap ini peneliti membuat *Thread Group* dengan cara menyesuaikan *Number of Threads (User)* berdasarkan dari skenario yang telah dibuat yaitu 5, 15, dan 45 user.
3. Selanjutnya membuat sampler *HTTP Request* untuk fungsi *website* dengan cara klik kanan pada *Thread Group*, kemudian pilih *Add*, kemudian pilih *Sampler*, kemudian pilih *HTTP Request* di gambar 6.



Gambar 6. Pembuatan HTTP *Request*

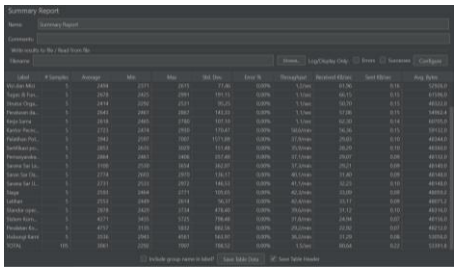
4. Kemudian lakukan konfigurasi dari sampler *HTTP Request* yang harus diisi seperti *Protocol* [http], server *name* atau IP, *GET Path*, *Port Number* agar proses simulasi beban dapat berjalan. Konfigurasi ini pun harus menyesuaikan dengan jumlah *user interface* yang dianalisis oleh peneliti.
5. Pembuatan *Event Listener* yaitu untuk menampilkan hasil simulasi penelitian dengan cara pilih *Thread Group*, kemudian pilih *Add*, kemudian pilih *Listener*, kemudian kita bisa memilih jenis *listener* yang dibutuhkan sebagai hasil di gambar 7.



Gambar 7. Pembuatan *Event Listener*

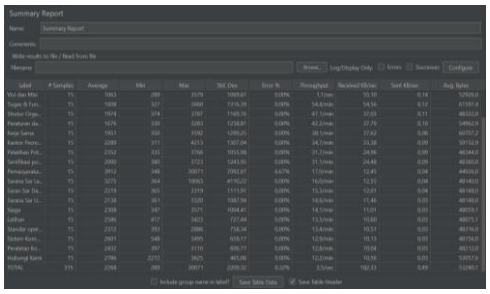
6. Adapun hasil simulasi beban terbaik yang didapat penelitidi gambar 8 adalah sebagai berikut.

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025



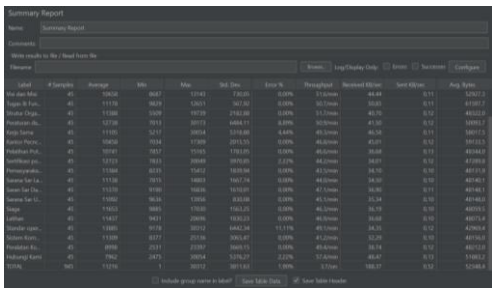
Gambar 8. Hasil *Summary Report* Pengujian 5 *user*.

Pada gambar diatas merupakan hasil *summary report* pengujian skenario 1 dengan menambahkan 5 *user*. Hasil dari pengujian menyatakan bahwa terdapat 105 sampel pengujian dengan hasil error 0,00% yang artinya pada simulasi skenario ini sudah baik di gambar 9.



Gambar 9. Hasil *Summary Report* Pengujian 15 *user*.

Pada gambar 9 merupakan hasil *summary report* pengujian skenario 2 dengan menambahkan 15 *user*. Hasil dari pengujian menyatakan bahwa terdapat 315 sampel pengujian dengan hasil error 0,32% pada 10.



Gambar 10. Hasil *Summary Report* Pengujian 45 *user*.

Dan pada gambar 10 merupakan hasil *summary report* pengujian skenario 3 dengan menambahkan 45 *user*. Hasil dari pengujian menyatakan bahwa terdapat 945 sampel pengujian dengan hasil error 1,90%.

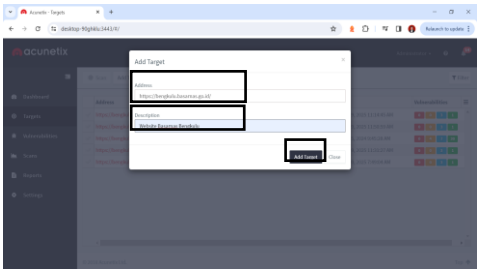
Secara keseluruhan sistem masih cukup baik di beban rendah dan menengah yaitu sampai 315 sampel namun dengan beban tinggi yaitu 945 sampel kinerja menurun drastis dengan indikasi waktu respon meningkat tajam, deviasi besar, dan tingkat error muncul dan sistem tampaknya memiliki batas optimal *throughput* sekitar 3,7/sec. maka diperlukan optimasi backend, skalabilitas, dan handling error agar sistem dapat menangani beban tinggi dengan stabil.

c. Acunetix

Pada penilaian kerentanan (*Vulnerability Assessment*) peneliti melaksanakan penelitian dibantu *tools* Acunetix. Dan langkah-langkah penilaian kerentanan yang dilakukan peneliti sebagai berikut:

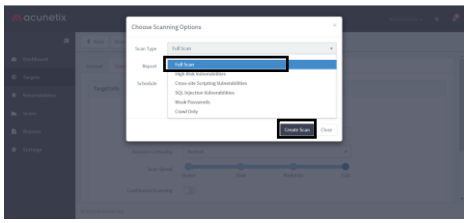
1. Setelah proses instalasi selesai kemudian kita bisa membuka *tools* acunetix.
2. Kemudian *Add* target yaitu input *url address* dan *description* pada kotak dialog dengan menggunakan url *address* yang ingin di analisa seperti pada gambar11.

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025



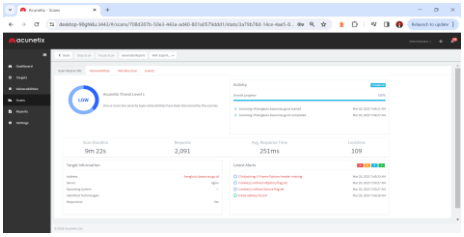
Gambar 11. *Add* Target *Website.*

3. Lakukan beberapa penyesuaian penelitian yang dibutuhkan sebelum melakuakan scanning. Dan apabila penyesuaian telah dilakukan kemudian dilakukan proses *Vurnerability Scanning*. Dan disini peneliti menggunakan tipe *full scan* agar mendapatkan hasil kerentanan secara keseluruhan. Setelah itu klik *create scan* seperti pada gambar12.



Gambar 12. *Create Scan.*

4. Kemudian akan tampil hasil scanning penilaian kerentanan yang akan menilai seberapa baik produk atau sistem melindungi informasi dan data pada *website* pada gambar 13 sebagai berikut.



Gambar 13. Hasil penilaian kerentanan

Dari gambar diatas dipaparkan kerentanan pada *website* Basarnas Bengkulu dikategorikan *Low-severity vulnerabilities* (ditemukan satu atau lebih kerentanan tingkat rendah) yang artinya masih relatif aman hanya terdapat kerentanan minor atau tidak terlalu mengancam sistem secara langsung namun tetap perlu ditindaklanjuti karena bisa dimanfaatkan sebagai bagian dari eksploitasi lanjutan.

3.2 Pembahasan

Pada bagian ini akan dibahas hasil penelitian performa, *overload*, dan kerentanan. Hasil analisis ini akan membahas apakah *website* masih dalam kondisi kinerja yang baik untuk dioperasikan dan apa saja upaya rekomendasi optimalisasi yang dapat dilakukan untuk meingkatkan kinerja *website*.

a. Pengukuran Performa

Pada pengukuran peforma hasil dari *tools* Pagespeed Insights dilakukan pada lebih dari satu halaman yang nantinya diambil rata-rata pengukuran di table 4.

Tabel 4. Rangkuman Statistik Kinerja Performa *Website.*

Metrik	Rata-rata	Ideal	Keterangan
Performance	81,67	≥ 90	Cukup Baik

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025

FCP *(First Contentful Paint)*	0,95 s	< 1,8 s	Sangat Baik
Speed Index	3,48 s	< 3 s	Perlu Perbaikan *Visual Randering*
TBT *(Total Bloking Time)*	94,28 ms	< 200 ms	Baik
LCP *(Largest Contentful Paint)*	1,79 s	< 2,5 s	Baik
CLS *(Cumulative Layout Shift)*	0,003	< 0,1	Sangat Baik

Pada hasil pengukuran performa diatas didapatkan hasil nilai rata-rata performa sebesar 81,67 yang berarti cukup baik karena 80 keatas umumnya diangap baik dan termasuk indikasi hijau dalam pagespeed insights. Performa sendiri adalah gabungan dari berbagai metrix baik FCP, *Speed Index*, TBT, LCP dan CLS. Sehingga hal ini menunjukkan bahwa secara keseluruhan *website* Basarnas Bengkulu termasuk dalam *website* yang optimasi cukup baik namun tetap masih ada ruang untuk perbaikan untuk mendapatkan hasil yang maksimal karena masih terdapat beberapa *user interface* yang berada dibawah ambang batas nilai ideal. Hal ini bisa berasal dari waktu muat yang lambat atau interaktivitas rendah.

First Contentful Paint (FCP) adalah menandakan seberapa cepat konten pertama muncul di layar pengguna. Pada hasil pengukuran diatas didapat nilai rata-rata FCP sebesar 0.95 s. Nilai ini tergolong sangat baik karena FCP idealnya adalah dibawah 1,8 s. Hampir semua *user interface* berada pada atau dibawah 1 s hal ini menunjukkan bahwa konten pertama muncul dengan cepat di layar pengguna. Hal ini juga menandakan bahwa HTML awal dan sumber daya utama CSS (*Cascading Style Sheet*) yaitu bahasa yang digunakan untuk mengatur tampilan atau layout dari elemen HTML dan JS kritikal (bagian dari skrip javascript yang benar-benar diperlukan agar halaman web dapat berfungsi atau tampil dengan benar) sudah cepat dimuat.

Speed index menunjukkan seberapa cepat konten terlihat seluruhnya di layar. Dari hasil pengukuran diatas didapat nilai rata-rata *speed index* sebesar 3,48 s yang artinya perlu perbaikan visual *rendering* karena nilai ideal harusnya kurang dari 3 s. Banyak *user interface*nya melebihi batas ini artinya pengalaman pemuatan visual terasa lambat. Hal ini bisa dipengaruhi oleh gambar yang ukurannya besar dan tidak di kompresi, javascript yang berat dan kadang komponen visual di bawah layar (*below-the fold*) dimuat terlalu awal. Hal ini bermaksud menampilkan bagian halaman yang hanya terlihat setelah pengguna scroll ke bawah. Namun bisa memperlambat waktu muat halaman. Optimasi gambar dan pengguna *resource renderbloking* dapat membantu dalam permasalahan ini.

Total Blocking Time (TBT) adalah total waktu dimana halaman tidak bisa merespons input pengguna (*klik, scroll*, dll) karena tugas JS berat. Dimana pada hasil pengukuran kali ini TBT yang didapat sebesar 94,28 ms termasuk kategori baik karena target dari google adalah kurang dari 200 ms yang artinya eksekusi JS tidak memblokir *rendering* secara berlebihan. *Rendering* sendiri adalah proses mengubah kode (HTML, SCC, JS) menjadi tampilan visual di layar pengguna. Namun ada beberapa *user interface* melebihi batas ideal yang berarti pengguna mengalami *delay* saat mencoba berinteraksi dan hal ini perlu optimasi Javascript.

Largest Contentful Paint (LCP) adalah waktu yang dibutuhkan untuk menampilkan elemen utama (biasanya gambar atau judul utama). Pada hasil pengukuran LCP kali ini mendapatkan skor 1.79 s masuk dalam kategori baik karena jauh dibawah batas ideal 2.5 s yang berarti server dan struktur HTML mendukung render cepat.

Cumulative layout shift (CLS) adalah mengukur kestabilan visual saat halaman dimuat (layout tidak bergeser tiba-tiba). Didapat skor nilai rata-rata hasil pengukuran sebesar 0.003 atau sangat baik karena masih dibawah waktu ideal yaitu 0.1 dan ini berarti tata letak halaman stabil, tidak ada pergeseran elemen akibat gambar tanpa dimensi, iklan, frame, dan sebagainya.

Berdasarkan hasil pengukuran performa diatas juga terdapat beberapa isu penting yang menjadi perhatian di atbel 5:

Tabel 5. Rangkuman Isu Perbaikan

Halaman	*Avoid an excessive DOM size*	*Eliminate render-blocking resources*	*Largest Contentful Paint element*	Minify CSS	Minify JavaScript	Minimize main-thread work	Properly size images	Reduce initial server response time	Reduce unused CSS	Reduce unused JavaScript	Serve images in next-gen formats
Beranda	v	v	v		v	v	v		v	v	v
Sejarah Basarnas		v	v		v		v	v	v	v	v
Arti Lambang		v	v	v	v		v	v	v	v	v
Visi & Misi		v	v	v	v		v	v	v	v	v
Tugas & Fungsi		v	v	v	v		v	v	v	v	v

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025

Struktur Organisasi	v	v	v	v	v	v	v	v	v
Peraturan & Hukum	v	v	v	v	v	v	v	v	v
Kerja Sama	v	v	v	v	v	v	v	v	v
Kantor Pencarian & Pertolongabn	v	v	v	v	v	v	v	v	v
Pelatihan Potensi Pencarian dan Pertolongan	v	v	v	v	v	v	v	v	v
Sertifikasi Pencarian & Pertolongan	v	v	v	v	v	v	v	v	v
Pemasyarakatan	v	v	v	v	v	v	v	v	v
Sarana SAR Laut	v	v	v	v	v	v	v	v	v
Sarana SAR Darat	v	v	v	v	v	v	v	v	v
Sarana SAR Udara	v	v	v	v	v	v	v	v	v
Siaga	v	v	v	v	v	v	v	v	v
Latihan	v	v	v	v	v	v	v	v	v
Standar Operasional Prosedur Pencarin & Pertolongan	v	v	v	v	v	v	v	v	v
Sistem Komunikasi	v	v	v	v	v	v	v	v	v
Peralatan Komunikasi	v	v	v	v	v	v	v	v	v
Hubungi Kami	v	v	v	v	v		v	v	v

Adapun penjelasannya dijabarkan serta rekomendasi perbaikannya diajabrkan sebagai berikut.

1. *Avoid an excessive DOM size* yaitu masalahnya DOM (*Document Object Model*) yang terlalu besar atau kompleks. Kelebihan dari DOM yang besar dapat mendukung fitur yang kompleks dan interaktif namun kekurangannya dapat memperlambat persing (proses di mana *browser* membaca dan memecah file HTML menjadi struktur data yang bisa dimengerti), menghambat *style calculation* (browser menghitung *style* apa yang berlaku untuk setiap elemen), serta menghambat rendering (proses menggambar halaman ke layar setelah DOM dan CSSOM terbentuk dengan browser membuat *render tree*, mengatur layout, dan menampilkan elemen). Sehingga hai ini dapat mempengaruhi TBT dan LCP. Adapun rekomendasi perbaikannya yaitu hindari nested HTML berlebih (jangan membuat struktur HTML yang terlalu dalam atau bersarang terlalu banyak tingkat), kita bisa juga memecah konten besar menjadi bagian dinamis (lazy load atau pagination), serta kita dapat juga menghapus elemen HTML yang tak digunakan.
2. *Eliminate render-blocking resources* yaitu masalahnya CSS dan JS di <head> menghambat rendering halaman awal. Kelebihannya yaitu dapat memastikan elemen penting dimuat dahulu namun kekurangannya dapat memperlambat FCP dan LCP. Adapun rekomendasi perbaikannya yaitu gunakan *async* atau *defer* pada skrip, inline CSS kecil untuk konten *above-the-fold*, serta dapat menggunakan *critical CSS*.
3. *Largest Contentful Paint element* yaitu elemen terbesar (biasanya gambar/benner) lambat dibuat. Kelebihannya yaitu elemen besar biasanya paling mencolok (*hero*, *image*, *banner*, dll) namun kekurangannya yaitu LCP yang tinggi mengakibatkan pengalaman pengguna yang buruk. Adapun rekomendasi perbaikannya yaitu optimasi ukuran gambar, gunakan *preload* untuk gambar LCP, serta pastikan elemen LCP muncul di awal HTML dan tidak terlambat karena JS.
4. *Minify CSS* yaitu masalahnya CSS terlalu besar karena tidak di-*minify* (menghapus karakter yang tidak perlu termasuk spasi dan komentar). Kelebihannya yaitu CSS lebih mudah dibaca saat dev namun kekurangannya membuat file lebih besar dan lambat diunduh. Adapun rekomendasi perbaikannya yaitu gunakan *tools* seperti cssnano, PurgeCSS, atau *minifier online* serta dapat aktifkan minify CSS di CMS seperti wordPress (melalui plugin: Autoptimize, WP Rocket).
5. *Minify javascript* yaitu masalahnya file JS tidak di-*minify*. Kelebihannya yaitu sama seperti CSS akan mudah dibaca dan *debug* namun kekurangannya ukuran file lebih besar dan memperlambat *load*. Adapun upaya rekomendasi perbaikannya yaitu gunakan *uglify*-js atau *build tools* (Webpack Rollup), serta terapkan *minify* otomatis saat *deployment*.

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025

6. *Minimize main-thread work* yaitu masalahnya proses di *main thread* terlalu berat (JavaScript parsing, layout, paint, dll). Kelebihannya yaitu fitur dinamis bisa diproses secara local namun kekeurangannya UI jadi lambat atau tidak responsif (meningkatkan TBT dan INP). Adapun rekomendasi perbaikannya yaitu optimalkan JS, gunakan web worker untuk proses berat, serta hindari penggunaan *framework* besar untuk fungsi kecil.
7. *Properly size images* yaitu masalahnya Gambar lebih besar dari ukuran tampilan yang dibutuhkan. Kelebihannya yaitu Gambar berkualitas tinggi ditampilkan namun *bandwidth* boros, waktu *load* meningkat. Adapun rekomendasi perbaikannya yaitu gunakan gambar dengan ukuran sesuai konteks (*srcset*, *sizes*) dan gunakan *lazy-loading* untuk gambar di bawah *fold*.
8. *Reduce initial server response time* yaitu masalahnya server butuh waktu lama menjawab permintaan awal (TTFB tinggi). Kelebihannya yaitu server kompleks bisa memproses banyak permintaan namun kekurangannya *delay* ini memperlambat seluruh proses *rendering*. Adapun rekomendasi perbaikannya yaitu gunakan *server cache* (Redis, Varnish), *upgrade server* (CPU, RAM), serta optimalkan *backend* dan *query database*.
9. *Reduce unused CSS* yaitu masalahnya banyak CSS tidak digunakan pada halaman yang dirender. Kelebihannya yaitu Style global bisa digunakan lintas halaman namun kekurangannya Membebani browser untuk parsing dan menunda rendering. Adapun rekomendasi perbaikannya yaitu gunakan PurgeCSS untuk menghapus CSS tidak digunakan, dan dapat juga menggunakan framework CSS modular (Tailwind, SASS).
10. *Reduce unused Javascript* yaitu masalahnya file JS memuat script yang tidak dipakai di halaman aktif. Kelebihannya yaitu menyediakan fitur siap pakai di semua halaman namun File besar memperlambat load dan eksekusi. Adapun rekomendasi perbaikannya yaitu split kode (code splitting) dan memuat hanya JS yang dibutuhkan dengan lazy loading.
11. *Serve Images in next-gen format* yaitu masalahnya Gambar masih pakai format lama (JPEG/PNG) bukan WebP/AVIF. Kelebihannya yaitu JPEG/PNG lebih kompatibel secara luas namun kekurangannya yaitu Ukuran file lebih besar. Adapun rekomendasi perbaikkanya yaitu konversi ke WebP atau AVIF dan gunakan fallback ke format lama untuk browser lama (jika perlu).

b. Simulasi Beban.

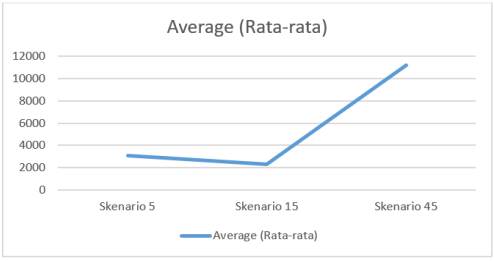
Berikut merupakan tabel rangkuman hasil simulasi beban dari *tools* Apache Jmeter dengan hasil pengujian skenario 5, 15, dan 45. Hasil pengujian 945 sampel. *Throughput* dihitung sebagai permintaan / satuan waktu. Waktu dihitung dari awal sampel pertama hingga akhir sampel terakhir. Ini termasuk interval antar sampel karena seharusnya mewakili beban di server. *Deviation* menunjukkan penyimpangan dari rata-rata ditabel 6.

Tabel 6. Rangkuman Hasil Penelitian dengan Apache Jmeter

Parameter	Skenario 1 5 User (105 Sampel)	Skenario 2 15 User (315 Sampel)	Skenario 3 45 User (945 Sampel)	Analisis
Average	3061 ms	2268 ms	11216 ms	Waktu respon memburuk drastis di skenario 45
Deviation	788,52	2209,32	3811,63	Variasi respon makin besar tanda ketidakstabilan
Error	0,00%	0,32%	1,90%	Munculnya error saat beban naik menunjukkan limitasi sistem
Throughput	1,5/sec	3,5/sec	3,7/sec	Throughput meningkat tapi stagnan di skenario 45
Byte send	0,22 kb	0,49 kb	0,52 kb	Permintaan makin kompleks / tambah data
Byte Receive	80, 64 kb	182,33 kb	188,37 kb	Respon makin besar, tapi pertumbuhan melambat

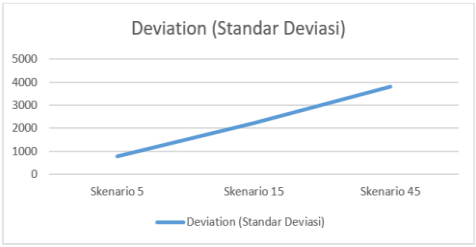
https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025

Pengujian yang telah dilakukan terhadap skenario 5, skenario 15, dan skenario 45 pada Apache Jmeter menghasilkan beberapa point penting. Point penting yang ditunjukkan pada beberapa parameter. Berikut ini merupakan penjelasan mengani hasil pengujian tersebut pada gambar 14



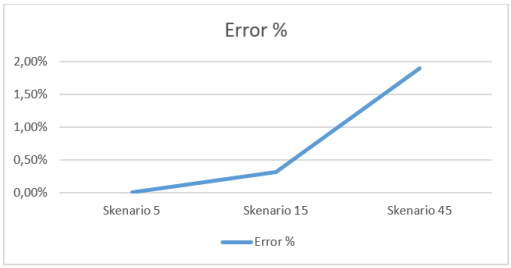
Gambar 14. Hasil *Average*.

Pada gambar diatas ditampilkan hasil parameter *average* yaitu waktu respon rata-rata *website*. Dari hasil diatas kelebihan waktu respon cukup baik di beban menengah hal ini terjadi mungkin karena caching atau efisiensi sistem, namun kekurangannya waktu respon meningkat tajam di beban tinggi (skenario 45) menunjukkan bahwa sistem mulai *overload* saat jumlah sampel mencapai 945. Adapun rekomendasi perbaikan yaitu lakukan optimasi kode backend dan database, gunakan caching dan gunakan load balancing untuk distribusi beban di gambar 15.



Gambar 15. Hasil *Deviation*.

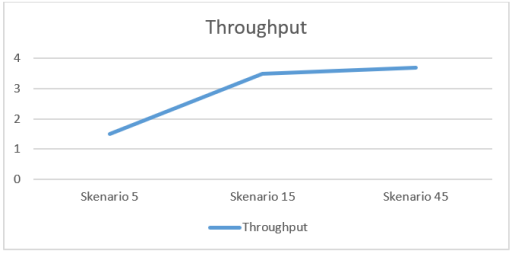
Pada gambar diatas ditampilkan hasil parameter *deviation* yaitu sebagai alat ukur untuk mengetahui seberapa besar suatu nilai atau kondisi menyimpang dari nilai atau kondisi yang seharusnya. Dari gambar yang diatas kelebihannya deviasi rendah di skenario ringan menandakan kestabilan namun kekurangannya deviasi tinggi di beban besar yang menandakan ketidakkonsistenan respon. Adapun rekomendasi perbaikan yaitu gunakan *thread pooling* dan *optimasi concurrency* pada *website* serta perbaiki *response time variability* dengan *profiling* performa pada gambar 16.



Gambar 16. Hasil *Error*.

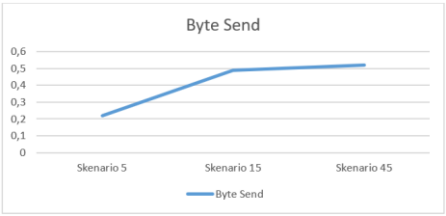
Pada gambar diatas ditampikan hasil *error* yaitu hasil nilai respon atau *output* dari pengujian yang gagal atau tidak sesuai dengan yang diharapkan. Pada gambar diatas kelebihannya tidak ada *error* di beban ringan 0,00%, namun kekurangannya error naik ke 1,90% di beban tinggi. Kesalahan mulai muncul saat jumlah beban meningkat ini adalah indikator bahwa sistem tidak mampu memproses semua permintaan dengan benar atau beban besar. Adapun rekomendasi perbaikan yaitu tingkatkan *timeout*, koneksi maksimum dan perbaiki *handling error* atau cek *log error*, serta gunakan *retry mechanism* di sisi client/server di gambar 17.

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025



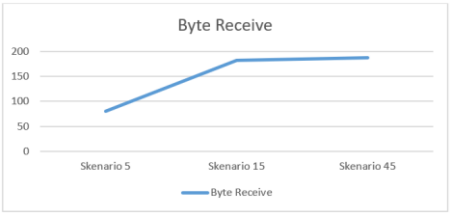
Gambar 17. Hasil *Throughput* (Jumlah Permintaan per Detik)

Pada gambar diatas ditampikan hasil *throughput* penelitian yaitu ukuran berapa banyak pekerjaan atau data yang berhasil diproses dalam satuan waktu yang menunjukkan kinerja kapasitas sistem dalam menangani beban. Pada gambar diatas kelebihannya yaitu *throughput* meningkat signifikan dari skenario 5 ke 15, namun kekurangannya *throughput* stagnan di skenario 45 hal ini mengindikasi *bottleneck* atau batas kinerja sistem. Rekomendasi perbaikan yaitu dapat menerapkan *horizontal scaling* (penambahan *instance server*) dan gunakan *asynchronous processing* untuk permintaan berat yaitu cara kerja sistem dimana sebuah tugas tidak harus selesai sebelum tugas berikutnya dimulai, sehingga sistem akan menjalankan tugas-tugas lain sambil menunggu hasil dari tugas yang lambat tanpa memblokir jalannya program Digambar 18.



Gambar 18. Hasil *Byte Send*

Pada gambar diatas ditampilkan hasil *byte send* penelitian yaitu jumlah data (dalam satuan byte) yang dikirim dari klien ke server. Pada gambar diatas kelebihannya pertumbuhan masih terkendali, namun kekurangannya peningkatan bisa membebani jaringan jika tidak dikompresi. Rekomendasi perbaikan yaitu gunakan kompresi HTTP (GZIP) dan pastikan payload yang dikirim hanya berisi data penting Digambar 19.



Gambar 19. Hasil *Byte Receive*

Pada gambar diatas ditampikan hasil *Byte Receive* yaitu jumlah data (dalam satuan byte) yang diterima oleh klien dari server dalam suatu komunikasi jaringan. Pada gambar diatas kelebihannya yaitu pertumbuhan terkendali walau beban naik, namun kekurangannya yaitu ukuran respon besar dapat menambah waktu loading. Ini menunjukkan bahwa hasil respon sistem mulai stagnan meskipun beban meningkat. Adapun rekomendasi perbaikan yaitu optimalkan struktur respon yang dikirim ke server, gunakan paginasi atau *lazy loading* untuk data besar yaitu optimasi dimana konten atau data hanya dimuat saat dibutuhkan bukan semua sekaligus di awal.

c. Penilaian Kerentanan

Hasil penilaian kerentanan dari *tools* Acunetix yang mengukur sejauh mana sistem dapat melindungi informasi dan data pada *website*, sehingga orang atau sistem lain memiliki tingkat akses sesuai dengan otoritasnya. Untuk itu hasil penilaian kerenatanan dengan menggunakan Acunetix dirangkum hasil ditabel 7 sebagai berikut.

Tabel 7. Hasil Penilaian Kerentanan dengan Acuneix

No	Kerentanan	*CVSS Score*	Tingkat Kerentanan
1	*Clickjacking: X-Frame-Options header missing*	6.8	*Low*

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025

2	*Cookie(s) without HttpOnly flag set*	0.0	*Low*
3	*Cookie(s) without Secure flag set*	0.0	*Low*
4	*Email address found*	0.0	*Informational*

Pada daftar kerentanan yang telah ditemukan tentunya memiliki dampak kerentanan yang berbeda di setiap kerentanannya. Adapun penjelasan dari kerentanan tersebut sebagai berikut.

1. *Clickjacking: X-Frame-Options header missing*. *Clickjacking* adalah serangan dimana penyerang menipu pengguna untuk mengklik sesuatu yang mereka tidak sadari sedang diklik. Biasanya dilakukan dengan cara menyisipkan website target ke dalam sebuah <iframe> di halaman berbahaya yang dibuat oleh penyerang. Tujuannya adalah untuk mencuri data atau informasi pribadi atau mengizinkan akses tanpa izin ke pengguna. Sedangkan *header X-Frame-Options* digunakan untuk mengontol apakah sebuah halaman boleh dimuat dalam <iframe> oleh domain lain. Oleh karena itu *Clickjacking: X-Frame-Options header missing* adalah kerentanan keamanan pada aplikasi *website* dimana header HTTP X-Frame Option tidak disetel sehingga halaman web bisa dimuat (*di-embed*) dalam elaman <iframe> oleh situs lain. Tingkat kerentanan pada penelitian ini adalah *low* dengan CVSS skor 6.8 tapi berpotensi dieksploitasi secara *social engineering.*
2. *Cookie(s) without HttpOnly flag set*. *HttpOnly* adalah atribut opsional pada cookie yang dirancang untuk mencegah akses cookie melalui JavaScript disisi klien. Cookie merupakan data kecil yang disimpan oleh *browser* yang digunakan untuk menyimpan sesi *login*, preferensi pengguna, atau informasi pelacakan. Oleh karena itu *Cookie(s) without HttpOnly flag set* merupakan kerentanan kemanan yang menunjukkan bahwa satu atau lebih cookie dikirim *browser* tanpa menyertakan *HttpOnly* yang berarti cookie tersebut bisa diakses melalui JavaScript (document.cookie) yang celah ini dapat dimanfaatkan dalam serangan *Cross-Sote Scripting* (XSS) untuk mencuri data seperti session ID. Tingkat kerentanan pada penelitian ini adalah *low* dengan CVSS skor 0.0 namun bisa menjadi *severe* jika dikombinasikan dengan XSS.
3. *Cookie(s) without Secure flag set. Secure* adalah atribut cookie yang memastikan bahwa cookie hanya akan dikirim ke server melalui koneksi HTTPS (terenkripsi). Jika *secure* tidak diatur, *browser* bisa mengirim cookie lewat HTTP yang memungkinkan pihak ketiga mencuri atau memodifikasi cookie tersebut. Oleh karena itu *Cookie(s) without Secure flag set* merupakan kerentanan keamanan yang berarti satu atau lebih cookie dikirim tanpa atribut *secure* , sehingga cookie tersebut dapat dikirim melalui koneksi HTTP biasa (tidak terenkripsi) dan rentan disadap oleh pihak ketiga. Tingkat kerentanan pada panelitian ini adalah *low* dengan CVSS skor 0.0 namun akan berdapkan jika tidak menggunakan HTTPS sepenuhnya.
4. *Email address found* merupakan kerentanan yang menunjukkan bahwa satu atau lebih alamat email terdeteksi dalam halaman web, file atau kode sumber. Ini bukan kerentanan langsung tetapi bisa menjadi indikator resiko kemanan atau privasi terutama jika data tersebut bersifat sensitif atau tidak semestinya diekspos karena dapat mengakibatkan masalah seperti serangan spam & phishing, kebocoran informasi pribadi, serta dapat dimanfaatkan untuk pemetaan sistem oleh penyerang apabila email tersebut disalahgunakan. Tingkat kerentanan pada penilaian ini adalah low dengan CVSS skor 0.0 *informational* tapi penting dari sisi privasi dan reputasi.

Setelah proses identifikasi ditemukan beberapa tingkat kerentanan pada *website* Basarnas Bengkulu yaitu *low* dan *informational* tentunya dari masing-masing kerentanan tersebut memiliki dampak kerentanan yang berbeda sehingga pada tahap ini akan dirangkum setiap kerentanan yang ditemukan dan memberikan rekomendasi terhadap kerentanan tersebut seperti tabel 8.

Tabel 8. Tabel Rekomendasi Perbaikan

Kerentanan	Kelebihan	Kekurangan	Rekomendasi Perbaikan
Clickjacking: X-Frame-Options header missing	Mudah diperbaiki hanya dengan menambahkan satu header	Tanpa perbaikan, situs bisa digunakan untuk phishing via iframe	Tambahkan header X-Frame-Option: "DENY" atau "SAMEORIGIN" di server side nya. Gunakan Content-Security-Policy (CSP) dengan Frame-ancestors
Cookie(s) without HttpOnly flag set	Deteksi cepat oleh scanner, perbaikan cepat	Sulit dimitigasi jika tidak mengontrol semua cookie (misalnya dari plugin)	Tambahkan flag HttpOnly saat mengatur cookie

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025

Cookie(s) without Secure flag set	Bisa diperbaiki dengan sedikit konfigurasi	Masih rawan jika situs campur antara HTTP dan HTTPS	Tambahkan *flag secure* saat mengatur cookie. Pastikan seluruh situs menggunakan HTTPS-only. Gunakan kombinasi atribut untuk perlindungan maksimal
Email address found	Meningkatkan transparansi dan kontak pengguna	Rentan scraping otomatis oleh bot	Gunakan formulir kontak dari pada menampilkan email langsung. Jika harus ditampilkan gunakan *obfuscation* (Metode Penyamaran). Atur *bot protection* CAPTCHA untuk mencegah *abuse form.* Validasi keamanan saat audit.

4. KESIMPULAN

Berdasarkan hasil penelitian analisis performa, overload dan kerentanan pada website Basarnas Bengkulu. Pada performa didapatkan rata-rata nilai skor sebesar 81,87 yang termasuk dalam range (baik) namun tetap perlu perbaikan berdasarkan pemanfaatan *tools* Pagespeed insights. Pada overload dengan memanfaatkan *tools* Apache Jmeter didapatkan hasil penelitian yaitu skenario 1 dengan 5 user (105 sampel) terdapat 0.00% error, skenario 2 dengan 15 user (315 sampel) terdapat 0.32% error, serta skenario 3 dengan 45 user (945 sampel) terdapat 1.90% yang secara keseluruhan menunjukkan kinerja website masih cukup baik pada beban rendah namun menurun pada beban tinggi. Dan pada kerentanan dengan memanfaatkan *tools* Acunetix didapatkan hasil penelitian pada kategori *low severity vulnerabilities* yang artinya terdapat kerentanan minor dan masih relatif aman namun tetap perlu ditindaklanjuti untuk upaya pencegahan.

Pada performa secara teknik sistem telah mampu mengoptimalkan berbagai aspek penting seperti kecepatan muat halaman, efisiensi penggunaan sumber daya, dan responsivitas terhadap pengguna. Hal ini mendukung teori bahwa performa sistem yang baik dapat meningkatkan pengalaman pengguna dan menunjang efektivitas penyampaian informasi pada layanan publik. Pada overload sistem mampu mempertahankan stabilitas ketika diuji pada kondisi beban rendah hingga menengah yang membuktikan bahwa infrastruktur server telah disesuaikan dengan kebutuhan operasional saat ini selaras dengan teori sistem terdistribusi dan manajemen beban yang menyatakan bahwa ketahanan sistem terhadap beban adalah indikator penting dalam pengembangan website. Serta dalam hal keamanan hasil penelitian menunjukkan bahwa tingkat kerentanan sistem berada pada level rendah yang menunjukkan bahwa pengelolaan sistem telah memperhatikan aspek perlindungan dasar terhadap potensi celah keamanan yang secara teori sistem dengan tingkat kerentanan rendah memberikan kepercayaan lebih tinggi terhadap integritas dan keamanan data yang disajikan.

UCAPAN TERIMAKASIH

Penulis mengucapkan terima kasih kepada Pimpinan, Staf dan Karyawan Badan Nasional Pencarian dan Pertolongan (BASARNAS) Provinsi Bengkulu.

REFERENCES

[1] R. M. I. R. Rusdy and S. Flambonita, "Penerapan Sistem Pemerintahan Berbasis Elektronik (Spbe) Di Pemerintah Daerah Untuk Mewujudkan Good Governance," *Lex LATA*, vol. 5, no. 2, pp. 218–239, 2023, doi: 10.28946/lexl.v5i2.2351.

[2] M. Septiani, R. Aulianita, V. Sofica, and N. Hasan, "Sistem Informasi Penjualan Kayu Kusen Berbasis Website," *Bianglala Inform.*, vol. 9, no. 2, pp. 103–107, 2021, doi: 10.31294/bi.v9i2.11603.

[3] N. Huda and M. Megawaty, "Analisis Kinerja Website Dinas Komunikasi dan Informatika Menggunakan Metode Pieces," *J. Sisfokom (Sistem Inf. dan Komputer)*, vol. 10, no. 2, pp. 155–161, 2021, doi: 10.32736/sisfokom.v10i2.1018.

[4] R. Susanti and N. Cahyono, "Analisis Dan Perbandingan Performa Website Penerimaan," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 8, no. 6, pp. 12044–12050, 2024.

[5] A. Suprapto and D. Sasongko, "Evaluasi Performa Website Berdasarkan Pengujian Beban Dan Stress Menggunakan Loadimpact (Studi Kasus Website Iain Salatiga)," *Netw. Eng. Res. Oper.*, vol. 6, no. 1, p. 31, 2021, doi: 10.21107/nero.v6i1.198.

[6] M. A. Aziz, "Vulnerability Assesment Untuk Mencari Celah Keamanan Web Aplikasi E-Learning Pada Universitas Xyz," *J. Eng. Comput. Sci. Inf. Technol.*, vol. 2, no. 1, 2023, doi: 10.33365/jecsit.v1i1.13.

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025

[7] I. G. N. P. D. D. D. Diastama, I. M. Sukarsa, and N. K. A. Wirdiani, “Pengembangan Test Script Untuk Load Testing Web Dengan Metode Software Testing Life Cycle,” *J. Ilm. Teknol. dan Komput.*, vol. 2, no. 1, pp. 311–318, 2021.
[8] F. J. Wangsa, Marlina, and Renny, “Optimasi Website Toko Kerja Menggunakan Uji Performa Google Pagespeed Insights,” *KHARISMA Tech*, vol. 18, no. 2, pp. 41–54, 2023, doi: 10.55645/kharismatech.v18i2.403.
[9] W. Tejaya, S. Rahman, and A. Munir, “Pengujian Website Invitees Menggunakan Metode Load Testing Dengan Apache Jmeter,” *KHARISMA Tech*, vol. 18, no. 1, pp. 99–112, 2023, doi: 10.55645/kharismatech.v18i1.305.
[10] F. Kristianto, S. Rahman, and S. Bahri, “Analisis Kerentanan Pada Website Servio Menggunakan Acunetix Web Vulnerability,” *Jtriste*, vol. 9, no. 1, pp. 46–55, 2022, doi: 10.55645/jtriste.v9i1.363.
[11] “View of ANALISIS KEAMANAN WEB NEW KUTA GOLF MENGGUNAKAN METODE VULNERABILITY ASSESSMENTS DAN PERHITUNGAN SECURITY METRIKS.pdf.”
[12] S. A. Sentosa, E. Subyantoro, and I. Asrowardi, “Pengukuran Kinerja Pada Aplikasi Video Pembelajaran UMKM Berbasis Web Dengan Metode Pengujian Beban,” *ROUTERS J. Sist. dan Teknol. Inf.*, vol. 2, no. 2, pp. 95–102, 2024, doi: 10.25181/rt.v2i2.3407.
[13] S. Yason, Sudirman, and A. Yunus, “Analisis Performa Website Sclean Menggunakan Pingdom Tools Dan Page Speed Insights,” *KHARISMA Tech*, vol. 17, no. 1, pp. 113–124, 2022, doi: 10.55645/kharismatech.v17i1.213.
[14] N. L. A. Sonia Ginasari, K. Suar Wibawa, and N. K. Ayu Wirdiani, “Pengujian Stress Testing API Sistem Pelayanan dengan Apache JMeter,” *JITTER J. Ilm. Teknol. dan Komput.*, vol. 2, no. 3, p. 552, 2021, doi: 10.24843/jtrti.2021.v02.i03.p14.
[15] M. K. Abdillah, Suprapto, and A. R. Perdanakusuma, “Analisis Kualitas Website XYZ.com menggunakan Model ISO/IEC 25010 Product Quality,” *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 8, no. 1, pp. 2548–964, 2024.
[16] Ruliansyah, Tukino, Baenil Huda, and April Lia Hananto, “Penerapan Software Testing Life Cycle Pada Pengujian Otomatisasi Platform Dzikra,” *CSRID (Computer Sci. Res. Its Dev. Journal)*, vol. 15, no. 1, pp. 01–11, 2023, doi: 10.22303/csrid.15.1.2023.01-11.

https://doi.org/10.33998/processor.2025.20.2.2507
Submitted: 09 Agustus 2025; Reviewed Round-1: 20 Agustus 2025; Reviewed Round-2: 28 Agustus 2025; Accepted: 02 September 2025; Published: 31 Oktober 2025