Prosiding Seminar Nasional Ilmu Teknik
Volume 2 Nomor 2 Desember 2025
e-ISSN: 3063-5713; p-ISSN: 3063-4709, Hal 667-675
DOI: https://doi.org/10.61132/prosemnasproit.v2i2.79
Tersedia: https://prosiding.aritekin.or.id/index.php/PROSEMNASPROIT/

Evaluasi Perbandingan *Laravel Blade* dan *Livewire* pada Sistem Presensi Akademik

Ali Sadikin[1*], Abdul Rahim[2], Muhammad Wardani[3], Irawan[4]
[1-4] Ilmu Komputer, Universitas Dinamika bangsa, Indonesia
Email: alisadikin@unama.ac.id[1], abdulrahim@unama.ac.id[2], ardan@unama.ac.id[3], irawan@unama.ac.id[4]*

Alamat: Jalan Jendral Sudirman Thehok, Jambi, Indonesia
**Penulis Korespondensi: alisadikin@unama.ac.id*

Abstract. *The increasing demand for interactive web applications has encouraged the adoption of server-driven approaches such as Livewire as an alternative to building Single Page Applications (SPAs) without complex client-side JavaScript. However, the performance implications of this approach compared to conventional methods remain insufficiently explored. This study presents an empirical comparison between Laravel Blade with AJAX and Livewire in an academic attendance system scenario. Performance evaluation was conducted using k6 on the same web server, complemented by manual browser-based testing to observe actual communication patterns. The results indicate that Livewire exhibits approximately 2.7× higher average response time and up to 6× greater bandwidth consumption than Laravel Blade, primarily due to its snapshot mechanism and state synchronization process. Conversely, Livewire demonstrates better stability, reflected by lower maximum response times and a 0% error rate. These findings highlight a clear trade-off between resource efficiency and development convenience, where Livewire favors stability and developer productivity, while Laravel Blade provides superior efficiency in terms of latency and bandwidth usage.*

Keywords: *Laravel Livewire; Load Testing; Performa Web; K6; Web Performance.*

Abstrak. Perkembangan aplikasi web interaktif mendorong penggunaan pendekatan *server-driven* seperti *Livewire* sebagai alternatif pengembangan *Single Page Application* (SPA) tanpa ketergantungan JavaScript yang kompleks. Namun, implikasi performa dari pendekatan ini dibandingkan metode konvensional masih memerlukan evaluasi empiris. Penelitian ini membandingkan kinerja aplikasi berbasis *Laravel Blade* dengan AJAX dan *Livewire* pada skenario sistem presensi akademik. Pengujian dilakukan menggunakan k6 pada web server yang sama, disertai pengujian manual melalui browser untuk mengamati pola komunikasi aktual. Hasil menunjukkan bahwa *Livewire* memiliki waktu respons rata-rata sekitar 2,7× lebih tinggi dan konsumsi *bandwidth* hingga 6× lebih besar dibandingkan *Laravel Blade*, yang disebabkan oleh mekanisme snapshot dan sinkronisasi state. Sebaliknya, *Livewire* menunjukkan stabilitas lebih baik dengan waktu respons maksimum lebih rendah dan tingkat kesalahan 0%. Temuan ini mengindikasikan adanya *trade-off* antara efisiensi sumber daya dan kemudahan pengembangan aplikasi interaktif, di mana *Livewire* unggul dalam stabilitas dan produktivitas pengembangan, sementara *Laravel Blade* lebih efisien dari sisi latensi dan bandwidth.

Kata kunci: *Laravel Livewire*; *Load Testing*; Performa Web; K6; Performa Web

1. LATAR BELAKANG

Laravel, sebagai kerangka kerja PHP dengan pangsa pasar 50% (JetBrains 2023), menyediakan dua pendekatan utama: Blade dengan pemuatan ulang halaman penuh, dan *Livewire* yang menawarkan komponen reaktif dengan *rendering* sisi server (Porzio 2024). *Livewire* menggunakan mekanisme snapshot untuk mempertahankan status komponen antar permintaan HTTP (Stauffer 2023). Meskipun menyederhanakan pengembangan, pertanyaan kritis muncul mengenai kompromi kinerja dalam aplikasi dengan interaksi sering dan operasi intensif data.

Naskah Masuk: 09 Oktober 2025; Revisi: 20 Oktober 2025; Diterima: 27 November 2025; Terbit: 30 Desember 2025;

Penelitian terkait menunjukkan *Laravel* memiliki waktu respons lebih tinggi namun menawarkan fitur matang. Penelitian (Amarulloh, Kurniasih, and Muchlis 2023)membandingkan *Laravel*, Django, dan Node.js, menemukan Express.js memiliki *execution speed* tercepat sedangkan Laravel berada di posisi ketiga. Penelitian (Purwanto, Nugroho, and Wijayanto 2018) menganalisis *Flask*, *Laravel*, dan Express.js menggunakan *Postman*, menemukan Express.js unggul dengan response time 53-58 ms dan error rate 0%, Laravel 556-683 ms dengan error rate 1,63-2,58%, dan Flask 723-1757 ms tanpa error. [6]membandingkan Laravel dengan Express.js menggunakan JMeter, menemukan Express.js 48,68 ms lebih cepat. Penelitian (Kansha 2023) menemukan *CodeIgniter* lebih cepat (1369 ms vs 1984 ms) namun Laravel unggul dalam maintainability. Penelitian (Hendayun, Ginanjar, and Ihsan 2023) melakukan pengujian API dengan load testing, mengidentifikasi batasan kapasitas dan pola degradasi kinerja. Meskipun penelitian ada memberikan wawasan tentang performa framework, belum terdapat studi empiris yang mengevaluasi karakteristik kinerja Livewire menggunakan pengujian beban sistematis, khususnya *overhead* snapshot dan implikasi bandwidth.

Kesenjangan ini penting mengingat adopsi *Livewire* yang meningkat dalam aplikasi produksi. Sistem presensi dosen, dengan pembaruan sering dan kebutuhan umpan balik *real-time*, menjadi kasus ideal untuk evaluasi kinerja. Pengembang memerlukan data empiris untuk keputusan arsitektur yang tepat (Pradana 2024).

Penelitian ini mengevaluasi kinerja *Livewire* pada sistem presensi dosen melalui studi komparatif dengan pendekatan laravel menggunakan K6. Tujuan penelitian: mengukur metrik kinerja kunci, mengidentifikasi akar penyebab *overhead*, dan menyusun rekomendasi praktis pemilihan pendekatan. Kontribusi penelitian mencakup data kinerja empiris, analisis mekanisme snapshot, dan strategi optimasi dengan kuantifikasi peningkatan yang diharapkan.

2. KAJIAN TEORITIS

Laravel dan *Livewire*

Laravel mengadopsi arsitektur MVC dengan penekanan pada produktivitas pengembang (Otwell 2024). Pendekatan *laravel* menggunakan *Blade* untuk rendering HTML dengan siklus permintaan-respons stateless. *Laravel Livewire* memungkinkan pengembang membangun antarmuka dinamis tanpa JavaScript ekstensif (Porzio 2024). Livewire bekerja dengan mekanisme snapshot yang menserialisasi status komponen dan mengirimkannya bersama setiap permintaan AJAX, menghasilkan *overhead bandwidth* bergantung pada kompleksitas status (Stauffer 2023).

Metrik Evaluasi Kinerja

Waktu respons merupakan indikator utama kinerja yang dipersepsikan pengguna (Liu et al. 2025). Standar Nielsen Norman Group (2020) menyatakan <100ms dipersepsikan instan, 100-300ms cepat, >1000ms lambat (Hikmat 2024). *Throughput* (req/s) mengindikasikan kapasitas sistem dalam menangani permintaan konkuren (Ambara et al. 2025). *Bandwidth* mencakup data terkirim dan diterima, berdampak pada biaya infrastruktur dan pengalaman pengguna mobile. Tingkat kesalahan mencerminkan keandalan sistem under beban (Hendayun et al. 2023).

Pengujian Beban dengan K6

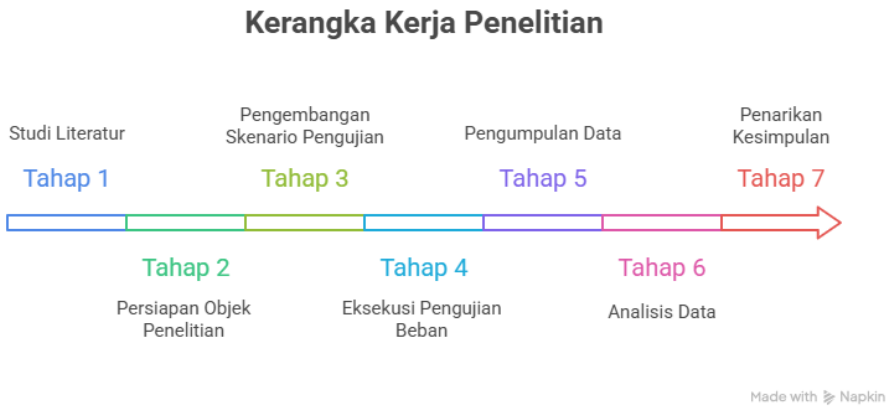
K6 merupakan alat pengujian beban modern dengan *scripting* JavaScript dan eksekusi Go untuk kinerja optimal (Labs 2024). *Virtual Users* (VUs) mengeksekusi skenario secara simultan dengan pola *ramp-up* untuk mensimulasikan lalu lintas realistis (Teslenko 2025). K6 menonjol sebagai alat pengujian beban modern karena pendekatan scripting berbasis JavaScript yang sederhana, ekspresif, dan mudah diintegrasikan dengan alur pengembangan perangkat lunak. Selain itu, implementasi K6 menggunakan bahasa *Go* memberikan efisiensi eksekusi dan stabilitas tinggi, sehingga mampu menangani simulasi beban secara andal pada skenario lalu lintas yang kompleks dan berskala besar (Pratama and Farisi 2025).

Penelitian Terkait

Penelitian (Amarulloh et al. 2023) membandingkan *Laravel*, Django, dan Node.js dalam REST API, menemukan *Laravel* memiliki pemrosesan lebih tinggi namun ekosistem matang. (Hadinata and Stianingsih 2024) menganalisis *Laravel* vs Express.js menggunakan JMeter, menunjukkan Express.js memiliki *throughput* lebih tinggi. (Kansha 2023) menemukan CodeIgniter lebih cepat (1369 ms vs 1984 ms) namun Laravel unggul dalam maintainability. (Hendayun et al. 2023) melakukan pengujian API dengan *load testing*, mengidentifikasi batasan kapasitas. Meskipun penelitian ada memberikan wawasan, terdapat kesenjangan dalam evaluasi empiris *Livewire* menggunakan pengujian beban sistematis terhadap overhead snapshot.

3. METODE PENELITIAN

Penelitian menggunakan pendekatan kuantitatif dengan desain studi komparatif, dilakukan melalui tahapan:



Gambar 2. Kerangka Kerja Penelitian

Gambar 2 merupakan kerangka kerja penelitian ini, berikut adalah penjelasannya:

1. Studi Literatur

   Kajian terhadap literatur terkait performa kerangka kerja PHP, *Laravel Livewire*, metrik evaluasi kinerja aplikasi web, dan metodologi pengujian beban menggunakan K6.

2. Persiapan Objek Penelitian

   Penyiapan sistem presensi dosen dalam dua versi: versi *laravel* menggunakan *Laravel* 6 dengan template *Blade*, dan versi *Livewire* menggunakan *Laravel* 11 dengan *Livewire* 3. Kedua versi mengimplementasikan logika bisnis identik dengan database yang sama.

Tabel 1. Perbandingan Implementasi Teknis

Halaman/Fitur	**Laravel Blade**	**Laravel Livewire**
Login	POST *form* + *full page reload*	POST *form* + *full page reload*
***Dashboard* Presensi**	GET + *full page reload*, render 36 mahasiswa	*Wire event no reload, render 36 mahasiswa + mount snapshot*
***Update* Status**	AJAX POST + JSON *response* (status, *message*)	*Wire event no page reload + Snapshot response*
Teknologi *Request*	XMLHttpRequest / Fetch API	Wire request (AJAX wrapper)
Payload Response	JSON	JSON + *Snapshot* + Checksum
DOM *Update*	Manual JavaScript *manipulation*	*Morphdom (automated patching)*

3. Pengembangan Skenario Pengujian

   Pembuatan skrip pengujian K6 yang mensimulasikan perjalanan pengguna: autentikasi, navigasi ke halaman presensi, dan pembaruan 6 status kehadiran mahasiswa dari total 36 mahasiswa. Konfigurasi menggunakan 5 *Virtual Users* konkuren dengan durasi pengujian 5 menit per eksekusi.

Tabel 2. Skenario Pengujian

Item	Detail
K6 Version	v0.48.0
VUs	5 concurrent
Duration	5 menit
Flow	Login → Load (36 data) → Update 6x → Repeat
Metrics	Response time, Throughput, Bandwidth, Error rate
Runs	3× per implementasi

4. Eksekusi Pengujian Beban

   Pelaksanaan load testing menggunakan K6 versi 0.48.0 pada lingkungan lokal (Intel i7-10700K, 32GB RAM, PHP 7.4/8.2, MySQL 8.0, Apache 2.4). Setiap implementasi diuji 3 kali untuk memastikan konsistensi hasil.

5. Pengumpulan Data

   Pengumpulan metrik kinerja meliputi: waktu respons (rata-rata, median, p95, maksimum), *throughput* (req/s), *bandwidth* (data terkirim/diterima), tingkat kesalahan, dan tingkat penyelesaian iterasi. Data di-ekspor dalam format JSON dari K6.

6. Analisis Data

   Analisis menggunakan statistik deskriptif untuk perbandingan kinerja antar implementasi. *Root cause analysis* dilakukan melalui inspeksi lalu lintas jaringan menggunakan *Developer Tools* dan profiling ukuran *snapshot* komponen *Livewire*.

7. Penarikan Kesimpulan

   Penyusunan kesimpulan berdasarkan hasil analisis data, identifikasi *trade-off* antara kedua pendekatan, dan perumusan rekomendasi praktis untuk pemilihan pendekatan yang sesuai dengan karakteristik aplikasi.

Keterbatasan penelitian meliputi perbedaan versi *Laravel* (v6 vs v11), lingkungan lokal yang tidak sepenuhnya representatif terhadap produksi, dan tingkat konkurensi terbatas (5 VUs).

4. HASIL DAN PEMBAHASAN

Hasil Pengujian Beban

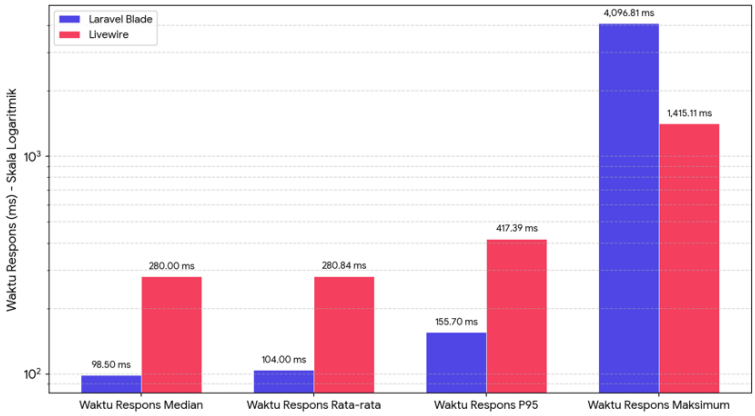
Tabel 3. Perbandingan Hasil Pengujian Beban

Metrik	*Laravel Blade*	*Livewire*	Rasio
Total Permintaan	866	981	**1,13×**
***Throughput* (req/s)**	4,07	4,61	**1,13×**
Tingkat Kesalahan	6 (0,69%)	0 (0%)	0×
Waktu Respons Rata-rata	104,00 ms	280,84 ms	**2,70×**
Waktu Respons Median	98,50 ms	280,00 ms	**2,84×**
Waktu Respons P95	155,70 ms	417,39 ms	**2,68×**
Waktu Respons Maksimum	4096,81 ms	1415,11 ms	**0,35×**
Data Diterima	43,45 MB	258,43 MB	**5,95×**
Data Dikirim	0,64 MB	2,30 MB	**3,59×**

Sumber: Hasil Pengujian K6, Januari 2026

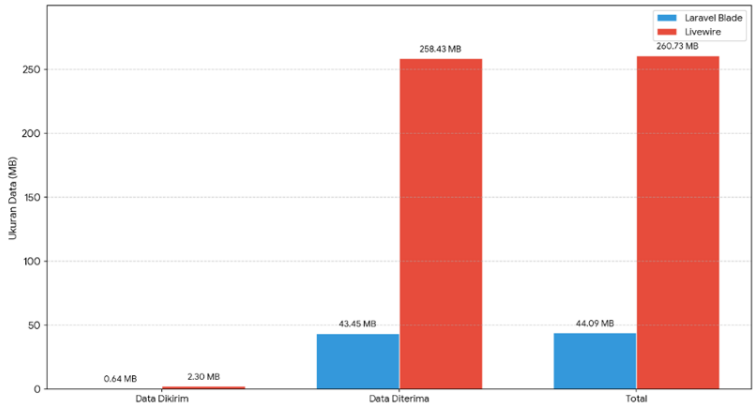
Pendekatan *Laravel Blade* menunjukkan throughput sebesar 4,07 req/s dengan waktu respons rata-rata 104,00 ms, yang mencerminkan efisiensi pemrosesan *server-side rendering*. Waktu *respons* maksimum mencapai 4.096,81 ms, mengindikasikan adanya *occasional slowdowns* yang kemungkinan dipicu oleh kontensi sumber daya pada sejumlah kecil *request*. Tingkat kesalahan yang tercatat sebesar 0,69% menunjukkan bahwa meskipun performanya tinggi, pendekatan *laravel blade* masih mengalami kegagalan *sporadis* di bawah beban bersamaan. Sebaliknya, *Livewire* memproses 981 permintaan dengan *throughput* 4,61 req/s (peningkatan sekitar 13%), namun dengan konsekuensi waktu respons rata-rata 280,84 ms, atau sekitar 2,7 kali lebih tinggi dibandingkan *Blade*, sehingga berpotensi memengaruhi persepsi responsivitas pengguna. Meskipun demikian, *Livewire* menunjukkan keandalan sempurna (0% error) sepanjang pengujian. Dari sisi konsumsi *bandwidth*, perbedaan terlihat signifikan, di mana *Livewire* menerima 258,43 MB data (sekitar 6 kali lebih besar) dan mengirim 2,30 MB data (sekitar 3,6 kali lebih besar) dibandingkan pendekatan laravel blade.

Analisis Komparatif



Gambar 2. Perbandingan Waktu Respon Aplikasi.

Berikut perbandingan penggunaan bandwidth:



Gambar 3. Perbandingan Penggunaan Bandwidth

Livewire menunjukkan waktu respons rata-rata sekitar 2,7× lebih tinggi dibandingkan *Laravel Blade*, dengan konsistensi pada distribusi latensi (median 2,8× dan P95 2,7×). Sebaliknya, waktu *respons* maksimum *Livewire* sekitar 0,35× lebih rendah, yang mengindikasikan kestabilan kinerja pada kondisi terburuk yang lebih baik. Perbedaan ini mencerminkan karakteristik arsitektural masing-masing pendekatan, di mana *Laravel Blade* memiliki jalur pemrosesan yang lebih ringan sehingga latensinya rendah pada kondisi normal, tetapi lebih rentan terhadap *outlier* ekstrem, sementara *Livewire* melibatkan tahapan tambahan seperti sinkronisasi state dan server-side rendering yang menghasilkan baseline latensi lebih tinggi namun perilaku eksekusi yang lebih deterministik.

Dari sisi konsumsi *bandwidth*, *Livewire* menerima data sekitar 6× lebih besar dibandingkan *Laravel Blade* (258,43 MB vs 43,45 MB), dengan total transfer meningkat lebih dari 5,9×. Secara rata-rata per permintaan, *Laravel Blade* mentransfer sekitar 50 KB, sedangkan *Livewire* mencapai sekitar 260 KB, menunjukkan *overhead* komunikasi sekitar 5×. Hasil ini diperkuat oleh pengujian manual melalui browser, di mana proses *update* presensi menggunakan Laravel Blade + AJAX hanya mentransfer sekitar 1 kB, sementara Livewire tetap mentransfer lebih dari 17 kB meskipun tanpa komponen tambahan, yang mengindikasikan bahwa *overhead* tersebut berasal dari mekanisme *snapshot* dan *sinkronisasi state*.

Meskipun memiliki *overhead* latensi dan bandwidth yang lebih tinggi, *Livewire* menunjukkan keandalan yang lebih baik dengan tingkat kesalahan 0%, sedangkan *Laravel Blade* mencatat 0,69%. Dalam skenario sistem presensi dengan puluhan interaksi per sesi, perbedaan ini berpotensi meningkatkan risiko kegagalan kumulatif pada *Laravel Blade*, sementara *Livewire* menawarkan stabilitas eksekusi yang lebih konsisten dengan konsekuensi peningkatan konsumsi sumber daya.

Diskusi Hasil

Livewire menunjukkan waktu respons rata-rata sekitar 2,7× lebih tinggi dibandingkan *Laravel Blade*, dengan konsistensi pada distribusi latensi (median 2,8× dan P95 2,7×). Sebaliknya, waktu respons maksimum *Livewire* sekitar 0,35× lebih rendah, yang mengindikasikan kestabilan kinerja pada kondisi terburuk yang lebih baik. Perbedaan ini mencerminkan karakteristik arsitektural masing-masing pendekatan, di mana *Laravel Blade* memiliki jalur pemrosesan yang lebih ringan sehingga latensinya rendah pada kondisi normal, tetapi lebih rentan terhadap *outlier* ekstrem, sementara *Livewire* melibatkan tahapan tambahan seperti sinkronisasi state dan *server-side rendering* yang menghasilkan baseline latensi lebih tinggi namun perilaku eksekusi yang lebih deterministik.

Dari sisi konsumsi *bandwidth*, *Livewire* menerima data sekitar 6× lebih besar dibandingkan *Laravel Blade* (258,43 MB vs 43,45 MB), dengan total transfer meningkat lebih dari 5,9×. Secara rata-rata per permintaan, *Laravel Blade* mentransfer sekitar 50 KB, sedangkan *Livewire* mencapai sekitar 260 KB, menunjukkan *overhead* komunikasi sekitar 5×. Hasil ini diperkuat oleh pengujian manual melalui browser, di mana proses update presensi menggunakan *Laravel Blade* + AJAX hanya mentransfer sekitar 1 kB, sementara *Livewire* tetap mentransfer lebih dari 17 kB meskipun tanpa komponen tambahan, yang mengindikasikan bahwa *overhead* tersebut berasal dari mekanisme *snapshot* dan sinkronisasi state.

Meskipun memiliki *overhead* latensi dan bandwidth yang lebih tinggi, *Livewire* menunjukkan keandalan yang lebih baik dengan tingkat kesalahan 0%, sedangkan *Laravel Blade* mencatat 0,69%. Dalam skenario sistem presensi dengan puluhan interaksi per sesi, perbedaan ini berpotensi meningkatkan risiko kegagalan kumulatif pada *Laravel Blade*, sementara *Livewire* menawarkan stabilitas eksekusi yang lebih konsisten dengan konsekuensi peningkatan konsumsi sumber daya.

5. KESIMPULAN DAN SARAN

Hasil pengujian menunjukkan bahwa pendekatan *Laravel Blade* memiliki keunggulan pada waktu *respons* yang lebih cepat dan penggunaan *bandwidth* yang lebih efisien, sehingga lebih cocok untuk sistem presensi yang membutuhkan performa tinggi dan konsumsi jaringan rendah. Sementara itu, *Livewire* menawarkan stabilitas interaksi yang baik, namun dengan konsekuensi latensi lebih tinggi dan *overhead* data yang signifikan. Kemudahan *Livewire* dalam membangun aplikasi web yang interaktif menyerupai *Single Page Application* (SPA) terbukti menyederhanakan proses pengembangan, tetapi mekanisme snapshot yang digunakan meningkatkan beban pemrosesan dan transfer data. Hal ini menunjukkan adanya *trade-off* yang jelas antara kemudahan pengembangan dan efisiensi performa. Dengan demikian, pemilihan pendekatan harus disesuaikan dengan kebutuhan sistem dan keterbatasan infrastruktur. Untuk aplikasi presensi dengan frekuensi pembaruan tinggi, pendekatan *laravel blade* lebih efisien, sementara *Livewire* lebih relevan ketika interaktivitas dan kemudahan pengembangan menjadi prioritas utama.

DAFTAR REFERENSI

Amarulloh, A., K. Kurniasih, and M. Muchlis. 2023. "Analisis Perbandingan Performa Web Service REST Menggunakan Framework Laravel, Django, Dan Node JS Pada Aplikasi Berbasis Website." *Jurnal Teknik Informatika* 9(1):12–17.

Ambara, M. P., M. Sudiarta, S. M. Suryaniadi, and I. G. T. S. Dharma. 2025. "Implementation of the CodeIgniter Framework in the Development of a Cloud-Based Competency Assessment System." Pp. 421–30 in *International Conference on Sustainable Green Tourism Applied Science-Engineering Applied Science 2025 (ICOSTAS-EAS 2025)*. Atlantis Press.

Hadinata, M. S., and R. W. Stianingsih. 2024. "Analisis Perbandingan Performa RESTful API Antara Express.Js Dengan Laravel Framework." *Jurnal Informatika Dan Teknik Elektro Terapan (JITET)* 7(2):545–54. doi:10.35313/jitet.v7i2.3845.

Hendayun, M., A. Ginanjar, and Y. Ihsan. 2023. "Analysis of Application Performance Testing Using Load Testing and Stress Testing Methods in API Service." *Jurnal Sisfotek Global* 13(1):28. doi:10.38101/sisfotek.v13i1.2656.

Hikmat, A. N. 2024. "Evaluasi Desain Interface Pada Aplikasi Ipusnas Berdasarkan Teori Evaluasi Heuristik Nielsen." *LIBRIA* 16(1):16–36.

JetBrains. 2023. "Developer Ecosystem Survey 2023."

Kansha, W. M. 2023. "Analisis Perbandingan Struktur Dan Performa Framework CodeIgniter Dan Laravel Dalam Pengembangan Web Application." *Jurnal Teknik Informatika STMIK Antar Bangsa* 9(1):25–30.

Labs, Grafana. 2024. "K6 Load Testing Documentation."

Liu, W., L. Guo, Y. W. Heng, C. Li, A. E. Hassan, and others. 2025. "Screencast-Based Analysis of User-Perceived GUI Responsiveness." *ArXiv Preprint ArXiv:2508.01337*.

Otwell, Taylor. 2024. "Laravel Documentation."

Porzio, Caleb. 2024. "Laravel Livewire Documentation."

Pradana, F. A. 2024. "Perancangan Sistem Presensi Deteksi Wajah Berbasis Website (Studi Kasus Laboratorium Sistem Manufaktur Terintegrasi UII)." Universitas Islam Indonesia.

Pratama, F., and A. Farisi. 2025. "Analisis Perbandingan Kinerja Backend API Menggunakan PHP, Golang, Dan JavaScript." *Techno Com* 24(1).

Purwanto, B., H. Nugroho, and A. Wijayanto. 2018. "Pengembangan Dashboard Akademik Berbasis Data Warehouse Di Perguruan Tinggi." *Jurnal Teknologi Informasi Dan Komputer* 4(1):12–20.

Stauffer, Matt. 2023. "Understanding Livewire: Server-Side Framework for Laravel."

Teslenko, Y. 2025. "Performance Percentile Analysis for API-Based Testing." *Авторський Колектив* 235.