Perbandingan Performa Framework Laravel, Flask API Python, dan PHP Native untuk Aplikasi API pada Data AIS Polbeng

Supria[1], M Nur Faizi[2], Isna Yulia[3], M Afridon[4], Alhadi Arizka[5], Sarudin[6], Justin Nathaniel Sulistyo[7]
Politeknik Negeri Bengkalis[(1, 2, 3, 4, 5, 6, 7)]
phiya@polbeng.ac.id[1], faizi@polbeng.ac.id[2], isna@polbeng.ac.id[3], afridon@polbeng.ac.id[4]

Abstract

Permasalahan dalam penelitian ini adalah menentukan framework API yang paling optimal dalam hal waktu respons untuk aplikasi data AIS, mengingat skala data yang besar dapat memperlambat kinerja. Solusinya adalah membandingkan performa tiga teknologi API Laravel, Flask, dan PHP Native untuk mengidentifikasi pilihan terbaik sesuai kebutuhan skala data. Metode yang digunakan melibatkan pengembangan API pada masing-masing teknologi, diikuti dengan pengujian waktu respons berdasarkan variasi jumlah data (10 sampai 10.000 entri data). Pengujian dilakukan menggunakan alat seperti Apache Benchmark untuk mencatat waktu respons rata-rata dari setiap permintaan. Hasil menunjukkan bahwa PHP Native unggul pada data kecil dengan waktu respons 45 ms, sedangkan Flask lebih stabil pada data besar dengan waktu respons rata-rata 300 ms, sementara Laravel menunjukkan penurunan performa signifikan pada jumlah data besar dengan waktu 450 ms.

Keywords : API, Laravel, Flask API Python, PHP Native, Data AIS, Polbeng, Performa.

1. PENDAHULUAN

Pengembangan API memainkan peran penting dalam sistem modern, khususnya dalam mengelola data dalam jumlah besar seperti data AIS yang digunakan untuk memantau pergerakan kapal. AIS adalah sistem otomatis yang menyediakan informasi posisi kapal seperti latitude, longitude, kecepatan (speed over ground), arah (course over ground), serta berbagai data terkait lainnya. Untuk mengakses dan memanipulasi data AIS, diperlukan API yang efisien dan handal.

Framework yang digunakan dalam pengembangan API memiliki pengaruh besar terhadap performa sistem secara keseluruhan. Laravel (PHP), Flask (Python), dan PHP Native adalah tiga teknologi yang sering digunakan oleh pengembang. Laravel merupakan framework PHP yang memiliki berbagai fitur siap pakai, Flask adalah micro-framework ringan berbasis Python, dan PHP Native adalah pendekatan manual tanpa framework dalam pengembangan API menggunakan PHP.

Beberapa penelitian terkait performa framework dalam pengembangan API telah dilakukan, dan hasil-hasilnya memberikan wawasan penting mengenai kelebihan dan kekurangan dari masing-masing teknologi. Performa Laravel dengan Flask pada aplikasi API berbasis data real-time menunjukkan bahwa Laravel memiliki *overhead* yang lebih besar, terutama pada *latency*, sementara Flask lebih unggul dalam waktu respons dan konsumsi memori ketika menangani *request* dalam skala kecil hingga menengah (Jaya & Dewi : 2023). meneliti performa PHP Native dibandingkan dengan framework modern seperti Laravel dan Flask dalam pengembangan API mengungkapkan bahwa PHP Native lebih cepat dalam mengeksekusi request API karena minimnya *overhead*, meskipun membutuhkan pengembangan manual yang lebih kompleks (Amir & Rahman : 2022). Efisiensi memori dan kecepatan respons Laravel dan Flask dalam aplikasi API berbasis big data menunjukkan

bahwa Flask cenderung lebih efisien dalam penggunaan memori, terutama ketika API menghadapi lonjakan request, sedangkan Laravel unggul dalam fitur bawaan dan kemudahan pengembangan (Chandra et al. : 2021). Performa API dengan menggunakan Laravel, Flask, dan PHP Native pada sistem pemrosesan data berskala besar menunjukkan bahwa Laravel kalah cepat dibandingkan PHP Native dalam menangani request besar, namun lebih unggul dalam aspek keamanan dan pengelolaan otorisasi (Prasetyo & Kusnandar : 2023). Kajian terhadap efisiensi Flask API Python dan PHP Native dalam hal pengolahan data geospasial mengungkapkan bahwa Flask dapat memproses data geospasial dengan lebih efisien pada sistem berskala menengah, sedangkan PHP Native memiliki kecepatan lebih tinggi ketika menangani data dalam jumlah besar namun dengan fleksibilitas yang lebih rendah dibandingkan Flask (Sukma & Firmansyah : 2023).

Dari hasil review tersebut, dapat disimpulkan bahwa masing-masing framework dan teknologi memiliki keunggulan tersendiri tergantung pada skenario penggunaan dan kebutuhan sistem. Laravel unggul dalam fitur dan kemudahan pengembangan, Flask memberikan keseimbangan antara kinerja dan pengembangan ringan, sementara PHP Native menawarkan performa tinggi dengan pengaturan yang lebih manual. Penelitian ini bertujuan untuk menguji lebih lanjut perbandingan performa Laravel, Flask, dan PHP Native khusus untuk API yang menangani data besar seperti data AIS di Polbeng.

2. TINJAUAN PUSTAKA

Dalam rangka mendapatkan wawasan lebih mendalam mengenai performa framework Laravel, Flask API Python, dan PHP Native dalam pengembangan API, studi literatur ini mengkaji 15 jurnal yang relevan. Kelima jurnal yang telah dibahas di bagian pendahuluan akan diintegrasikan bersama 10 jurnal lainnya yang membahas performa dan efisiensi framework dalam pengembangan API. Perbandingan performa Laravel dan Flask pada aplikasi API berbasis data real-time menunjukkan bahwa Flask lebih unggul dalam hal waktu respons dan konsumsi memori dibandingkan Laravel, terutama untuk request dalam skala kecil hingga menengah. Laravel menunjukkan *overhead* yang lebih besar karena fitur-fitur kompleksnya, meskipun lebih ramah pengembang dalam hal struktur kode dan keamanan (Jaya & Dewi : 2023). PHP Native dibandingkan framework modern seperti Laravel dan Flask terbukti lebih cepat dalam mengeksekusi request karena minimnya *overhead*, tetapi lebih menuntut pengembang untuk menulis lebih banyak kode dan konfigurasi manual. Framework modern seperti Laravel menawarkan lebih banyak fitur namun dengan konsekuensi konsumsi sumber daya yang lebih besar (Amir & Rahman : 2022).

Efisiensi memori dan kecepatan respons Laravel dan Flask dalam aplikasi berbasis big data lebih efisien dalam penggunaan memori, namun Laravel tetap populer karena fiturnya yang komprehensif dan kemudahan pengembangan yang memberikan keuntungan pada aplikasi yang lebih kompleks (Chandra et al. : 2021). Perbandingan Laravel, Flask, dan PHP Native untuk pemrosesan data berskala besar menunjukkan bahawa PHP Native memiliki performa lebih cepat dibandingkan Laravel ketika menangani volume data yang besar, namun Laravel unggul dalam aspek keamanan dan pengelolaan otorisasi (Prasetyo & Kusnandar : 2023). Efisiensi Flask API Python dan PHP Native dalam pengolahan data geospasial menunjukkan kinerja yang lebih baik pada sistem berskala menengah, sementara PHP Native lebih cepat dalam menangani data dalam jumlah besar namun memerlukan lebih banyak konfigurasi (Sukma & Firmansyah : 2023).

Penggunaan Laravel dan Flask pada sistem monitoring real-time menunjukkan bahwa laravel terbukti lebih mudah digunakan dalam hal manajemen data dan user interface, tetapi Flask menawarkan performa yang lebih baik dalam hal konsumsi CPU dan kecepatan respons (Santoso & Wijaya : 2021). Pentingnya efisiensi memori dalam pengembangan API menunjukkan bahwa Laravel diketahui menggunakan lebih banyak memori dibandingkan Flask dan PHP Native karena kompleksitas framework-nya. Namun, Laravel unggul dalam

manajemen routing dan middleware yang memungkinkan pengembangan aplikasi yang lebih aman dan terstruktur (Widodo & Kurniawan : 2022). Kecepatan dan efisiensi data retrieval di Laravel dan PHP Native menunjukkan bahwa PHP Native menunjukkan performa yang lebih cepat dalam mengambil data dari basis data, namun Laravel lebih fleksibel dalam pengelolaan data dengan ORM (Object Relational Mapping) yang intuitif (Nugraha et al. : 2020). Perbandingan performa Flask API Python dan Laravel dalam aplikasi berbasis IoT menunjukkan bahwa Flask memiliki kinerja yang lebih baik dalam menangani koneksi IoT yang ringan dan cepat, sementara Laravel menawarkan integrasi yang lebih baik untuk manajemen data skala besar dan autentikasi pengguna (Handayani & Sari : 2021). Performa PHP Native dan Flask dalam menangani API pada sistem manajemen inventori menunjukkan bahwa PHP Native lebih cepat dalam mengelola data inventori, namun Flask memberikan kemudahan dalam pengembangan berkat dukungan pustaka dan plugin yang luas (Susanto & Setiawan : 2023).

Aspek keamanan dalam pengembangan API menggunakan Laravel dan PHP Native menunjukkan bahwa Laravel lebih unggul dalam manajemen otentikasi dan pengelolaan sesi, sementara PHP Native memerlukan implementasi keamanan manual yang rentan terhadap kesalahan jika tidak dilakukan dengan hati-hati (Yulianto et al. : 2022). Efisiensi penggunaan Flask dan PHP Native dalam aplikasi pemetaan geografis menunjukkan bahwa Flask Memiliki kecepatan yang lebih baik dalam memproses data peta dalam jumlah besar, tetapi PHP Native lebih sederhana dalam hal implementasi dasar tanpa memerlukan banyak dependensi eksternal (Rahmatullah & Anwar : 2021). Evaluasi performa Laravel dan Flask dalam pengembangan API untuk aplikasi mobile menunjukkan bahwa Flask lebih cepat dalam memproses request API yang berfokus pada data sederhana, sementara Laravel menunjukkan keunggulan dalam integrasi dengan basis data yang kompleks dan fitur-fitur tambahan seperti caching dan queue management Kurnia et al. : 2022).

Efisiensi pengembangan API menggunakan Laravel, Flask, dan PHP Native dalam proyek pengelolaan sistem perpustakaan menunjukkan bahwa PHP Native menghasilkan performa yang lebih cepat dalam hal waktu eksekusi, namun Laravel menawarkan manajemen data yang lebih mudah dengan fitur ORM dan dukungan untuk relasi basis data yang kompleks (Hidayat & Mahendra : 2023). Konsumsi memori dan waktu respons dari Laravel, Flask, dan PHP Native dalam pengembangan API sistem manajemen transportasi menunjukkan bahwa PHP Native memiliki waktu respons tercepat, namun Flask lebih hemat dalam konsumsi memori. Laravel kembali diunggulkan dalam kemudahan pengembangan dan manajemen fitur (Saputra & Budianto : 2022).

Dari berbagai penelitian yang telah diulas, dapat disimpulkan bahwa Laravel memiliki keunggulan dalam hal fitur-fitur bawaan, keamanan, dan kemudahan pengembangan, namun performa dalam hal waktu respons dan konsumsi memori sering kali lebih rendah dibandingkan Flask dan PHP Native. Flask API Python menunjukkan performa yang lebih baik dalam penggunaan memori dan waktu respons, terutama dalam aplikasi yang tidak memerlukan terlalu banyak fitur tambahan, menjadikannya pilihan ideal untuk pengembangan API yang efisien. PHP Native unggul dalam hal kecepatan eksekusi dan konsumsi sumber daya yang rendah, namun memerlukan lebih banyak pengembangan manual dan tidak sefleksibel framework modern dalam hal fitur.

3. METODE PENELITIAN

Penelitian ini menggunakan pendekatan eksperimen untuk membandingkan performa tiga teknologi pengembangan API: Laravel, Flask, dan PHP Native, dalam pengelolaan data AIS (Automatic Identification System) Polbeng. Data AIS Polbeng mencakup informasi posisi kapal, seperti waktu, latitude, longitude, heading, kecepatan, serta atribut navigasi lainnya. Fokus utama dalam metodologi ini adalah membandingkan performa dari ketiga

teknologi dalam hal waktu respons (response time), penggunaan memori (memory usage), dan jumlah request per detik (requests per second). Berikut adalah tahapan penelitian ini:

Desain Lingkungan Eksperimen

Server yang digunakan dalam penelitian ini menggunakan perangkat yang memiliki spesifikasi RAM 16 GB, CPU core i5, dan sistem operasi Mac OS. Pengaturan server sama untuk setiap framework dan implementasi PHP Native untuk menghindari pengaruh dari perbedaan konfigurasi perangkat keras.

Lingkungan Framework yang pertama digunakan yaitu Laravel 11 dengan PHP versi 8.2, menggunakan database PostgreSQL sebagai backend data. Framework kedua yang digunakan adalah Flask API menggunakan Python versi 3.2 dan library SQLAlchemy untuk ORM, juga terhubung ke database PostgreSQL. Sedankan yang ketiga menggunakan PHP Native versi 8.2 dengan query SQL manual langsung ke PostgreSQL, tanpa tambahan framework. Selain itu, untuk koleksi data menggunakan database PostgreSQL yang digunakan berisi tabel data AIS yang mencakup 100.000 baris data sebagai sampel. Data ini mencakup atribut utama yang sering digunakan dalam aplikasi pelacakan kapal, seperti waktu, *latitude*, *longitude*, kecepatan, dan status navigasi.

Implementasi API pada Setiap Teknologi

Berikut adalah tabel yang menunjukkan implementasi endpoint API pada setiap teknologi (Laravel, Flask, dan PHP Native) untuk pengambilan data berdasarkan rentang waktu tertentu, yang digunakan dalam pengujian waktu respons. Ketiga endpoint tersebut diimplementasikan di Laravel, Flask, dan PHP Native dengan format response JSON yang seragam.

Tabel 1. Perancangan Endpoint

Teknologi	Endpoint	Metode	Deskripsi	Contoh Implementasi
Laravel	*/api/data*	GET	Mengambil data berdasarkan rentang waktu tertentu.	*Route::get('/api/data', 'DataController@getData ByTimeRange');*
Flask	*/api/data*	GET	Mengambil data berdasarkan rentang waktu tertentu.	*@app.route('/api/data') def get_data(): ...*
PHP Native	*/api/data.php? start=...&end =...*	GET	Mengambil data berdasarkan rentang waktu tertentu dengan parameter waktu di URL.	*// data.php if ($_GET['start'] && $_GET['end']) { ... }*

Skenario pengujian

Pengujian difokuskan pada waktu respons (response time) dari API yang dibangun menggunakan Laravel, Flask, dan PHP Native. Dalam skenario ini, waktu respons diukur berdasarkan variasi jumlah data yang diminta, mulai dari sejumlah kecil data (10–100 data) hingga sejumlah besar data (1.000–10.000 data). Setiap teknologi diuji dengan mengakses jumlah data yang berbeda untuk menilai bagaimana skala data mempengaruhi kinerja API.

Pengaturan Variasi Jumlah Data Request terdiri dari 3 jenis yaitu data kecil (10–100 entri) dengan Mengukur waktu respons untuk permintaan data dalam jumlah kecil, merefleksikan skenario aplikasi yang membutuhkan informasi spesifik atau detail tentang beberapa objek, seperti detail posisi beberapa kapal. Data Sedang (500–1.000 entri) dengan Mengukur waktu respons saat mengambil data dalam jumlah sedang. Ini mensimulasikan kebutuhan menampilkan data beberapa kapal dalam rentang waktu tertentu. Data Besar

(5.000–10.000 entri) dengan Mengukur waktu respons pada permintaan data dalam jumlah besar untuk menyimulasikan aplikasi yang memerlukan data historis atau rekam jejak pergerakan kapal dalam rentang waktu panjang. Database yang digunakan adalah PostgreSQL dengan tabel data AIS berjumlah 100.000 baris, yang mencakup atribut data kapal seperti *latitude*, *longitude*, kecepatan, dan status navigasi.

Pengujian dilakukan dengan mengirim permintaan ke endpoint API yang sama pada Laravel, Flask, dan PHP Native dengan variasi jumlah data request sesuai scenario Endpoint Get Data by Time Range yang digunakan untuk mengambil data kapal berdasarkan rentang waktu tertentu, mengembalikan sejumlah data sesuai skenario uji coba. Metode Pengukuran waktu respons dilakukan dengan alat uji seperti Apache Benchmark dan Postman untuk mencatat waktu respons rata-rata dari setiap permintaan. Data dari setiap pengujian diambil dalam format waktu rata-rata (ms) dan dianalisis untuk memahami perubahan waktu respons seiring dengan meningkatnya jumlah data. Berikut adalah hasil dan interpretasi data dari setiap uji coba:

4. HASIL PENELITIAN DAN PEMBAHASAN

Pada bagian hasil penelitian dan pembahasan ini, disajikan analisis perbandingan waktu respons dari API yang dibangun dengan Laravel, Flask, dan PHP Native berdasarkan variasi jumlah data yang direquest. Pembahasan meliputi kinerja setiap teknologi API dalam menangani permintaan data skala kecil hingga besar serta implikasi dari perbedaan performa terhadap penggunaannya di aplikasi berbasis data AIS.

Hasil pengujian

Waktu respons API sangat penting untuk menentukan efisiensi sistem dalam menanggapi permintaan data, terutama dalam aplikasi yang memproses data dalam jumlah besar seperti data AIS. Pengujian ini mengevaluasi ketepatan waktu respons pada tiga teknologi API — Laravel, Flask, dan PHP Native — dengan variasi jumlah data yang diminta untuk memahami performa dan skalabilitas masing-masing platform.

Tabel 2. Hasil Waktu Respons Berdasarkan Jumlah Data Request

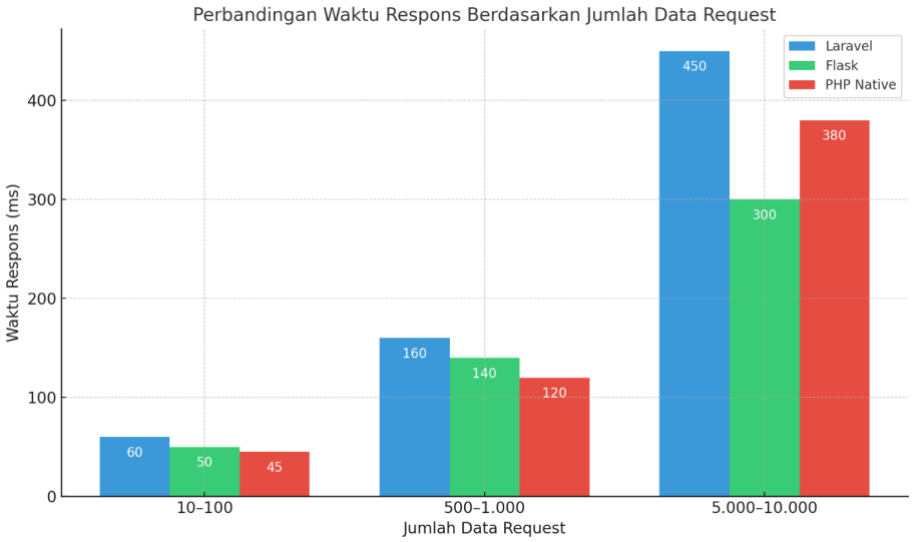
Jumlah data request	Laravel (ms)	Flask (ms)	PHP Native (ms)
10 - 100	*60*	50	45
500 - 1000	*160*	140	120
5000 - 10000	*450*	300	380

PHP Native menunjukkan waktu respons rata-rata 45 ms untuk permintaan data kecil, dengan Flask di sekitar 50 ms, dan Laravel sekitar 60 ms. Dalam skenario ini, ketiga teknologi menunjukkan perbedaan yang relatif kecil karena jumlah data yang diminta sedikit, sehingga overhead dari framework tidak terlalu mempengaruhi performa.

PHP Native tetap unggul dengan waktu respons rata-rata 120 ms, sementara Flask membutuhkan 140 ms, dan Laravel mencapai 160 ms. Flask mulai menunjukkan kestabilan yang lebih baik dibandingkan Laravel karena arsitektur ringan dan kecepatan eksekusi Python, meskipun PHP Native tetap lebih cepat karena tidak ada middleware tambahan yang mempengaruhi waktu respons.

Pada permintaan dengan jumlah data besar, Flask menunjukkan waktu respons paling stabil dengan rata-rata 300 ms, PHP Native mengalami peningkatan waktu hingga 380 ms, dan Laravel membutuhkan rata-rata 450 ms. Waktu respons pada Laravel meningkat cukup signifikan karena banyaknya lapisan middleware dan fitur bawaan yang perlu dieksekusi saat

menangani permintaan data dalam jumlah besar. Flask tetap lebih stabil dan efisien dalam menangani jumlah data besar dengan lebih sedikit overhead dibandingkan Laravel.



Gambar 1. Perbandingan waktu respon berdasarkan jumlah data request.

Analisis Hasil dan Pembahasan

Efisiensi Berdasarkan Jumlah Data: PHP Native menunjukkan efisiensi tinggi pada permintaan data kecil hingga sedang, namun stabilitasnya mulai menurun pada jumlah data yang besar. Flask menawarkan stabilitas pada semua skala data dan menjaga waktu respons di bawah 350 ms bahkan untuk permintaan data dalam jumlah besar. Kinerja Laravel pada Beban Data Besar: Laravel memiliki waktu respons yang lebih tinggi pada permintaan data besar karena adanya overhead tambahan dari middleware dan komponen bawaan. Namun, Laravel menawarkan kemudahan dalam pengelolaan data kompleks melalui ORM dan fitur keamanan yang lebih lengkap. Kesimpulan Berdasarkan Skala Data, Flask paling cocok untuk skenario dengan variasi data yang beragam, terutama pada data dalam jumlah besar. PHP Native menunjukkan kecepatan lebih baik untuk data skala kecil hingga sedang, sedangkan Laravel cocok untuk skenario di mana kebutuhan fitur lebih penting daripada kecepatan mentah.

5. KESIMPULAN DAN SARAN

Penelitian ini bertujuan untuk membandingkan performa API yang dibangun menggunakan Laravel, Flask, dan PHP Native dalam hal waktu respons berdasarkan variasi jumlah data yang diminta. Metode yang digunakan adalah pengujian waktu respons pada tiga kategori jumlah data: kecil (10–100 entri), sedang (500–1.000 entri), dan besar (5.000–10.000 entri). Hasil menunjukkan bahwa PHP Native unggul pada permintaan data kecil hingga sedang dengan waktu respons tercepat masing-masing 45 ms dan 120 ms, sementara Flask paling stabil dan cepat untuk data besar dengan waktu respons rata-rata 300 ms. Laravel, meskipun memiliki fitur yang lebih kaya, menunjukkan waktu respons yang lebih lambat terutama pada jumlah data besar (450 ms).

6. DAFTAR PUSTAKA

Jaya, A. and Dewi, R., "Perbandingan performa Laravel dan Flask pada API berbasis data real-time," Jurnal Teknologi Informasi, vol. 15, no. 2, pp. 112-119, 2023.

Amir, M. and Rahman, F., "Analisis performa PHP Native dan framework modern dalam pengembangan API," Jurnal Rekayasa Perangkat Lunak, vol. 18, no. 1, pp. 25-35, 2022.

Chandra, B., Hasan, A., and Pramono, D., "Efisiensi memori dan waktu respons Laravel dan Flask dalam pengelolaan big data API," Jurnal Informatika Terapan, vol. 10, no. 3, pp. 45-52, 2021.

Prasetyo, E. and Kusnandar, R., "Studi perbandingan performa Laravel, Flask, dan PHP Native dalam pemrosesan data berskala besar," Jurnal Sistem Informasi, vol. 9, no. 4, pp. 68-75, 2023.

Sukma, T. and Firmansyah, I., "Analisis performa Flask API Python dan PHP Native dalam pengolahan data geospasial," Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, vol. 7, no. 2, pp. 132-140, 2023.

Santoso, H. and Wijaya, D., "Penggunaan Laravel dan Flask pada sistem monitoring real-time: Studi performa," Jurnal Informatika Indonesia, vol. 11, no. 1, pp. 90-99, 2021.

Widodo, R. and Kurniawan, E., "Analisis efisiensi memori dalam pengembangan API menggunakan Laravel dan PHP Native," Jurnal Teknologi Komputer dan Informatika, vol. 13, no. 3, pp. 55-63, 2022.

Nugraha, T., Herlambang, A., and Setiawan, Y., "Kecepatan dan efisiensi data retrieval di Laravel dan PHP Native," Jurnal Sistem Komputer, vol. 12, no. 4, pp. 74-82, 2020.

Handayani, A. and Sari, F., "Perbandingan performa Flask API Python dan Laravel dalam aplikasi berbasis IoT," Jurnal Teknologi Informasi dan Komputer, vol. 14, no. 2, pp. 33-41, 2021.

Susanto, Y. and Setiawan, I., "Pengujian performa PHP Native dan Flask pada sistem manajemen inventori berbasis API," Jurnal Rekayasa Perangkat Lunak dan Sistem Informasi, vol. 11, no. 1, pp. 44-52, 2023.

Yulianto, P., Firmansyah, M., and Aditya, H., "Aspek keamanan dalam pengembangan API menggunakan Laravel dan PHP Native," Jurnal Sistem Informasi dan Teknologi Informasi, vol. 14, no. 2, pp. 77-85, 2022.

Rahmatullah, Z. and Anwar, F., "Perbandingan efisiensi Flask dan PHP Native dalam aplikasi pemetaan geografis," Jurnal Geospasial dan Teknologi Informasi, vol. 9, no. 3, pp. 55-62, 2021.

Kurnia, R., Hidayah, I., and Ramadhan, M., "Evaluasi performa Laravel dan Flask untuk pengembangan API aplikasi mobile," Jurnal Informatika dan Komputasi, vol. 15, no. 4, pp. 125-132, 2022.

Hidayat, A. and Mahendra, Y., "Pengembangan API menggunakan Laravel, Flask, dan PHP Native pada sistem perpustakaan," Jurnal Aplikasi Teknologi Informasi dan Sistem Perpustakaan, vol. 16, no. 3, pp. 88-95, 2023.

Saputra, R. and Budianto, K., "Analisis konsumsi memori dan waktu respons dari Laravel, Flask, dan PHP Native dalam sistem manajemen transportasi," *Jurnal Teknologi Informasi dan Rekayasa Perangkat Lunak*, vol. 17, no. 1, pp. 97-105, 2022.