BAB I PENDAHULUAN

I.1 Latar Belakang

Perkembangan teknologi web mendorong berbagai sektor, termasuk pendidikan, untuk menghadirkan aplikasi yang mampu memberikan pengalaman pengguna yang efisien dan responsif. Studi mengenai interaksi manusia dan komputer menegaskan bahwa kecepatan waktu muat halaman dan kualitas interaksi antarmuka menjadi faktor kunci dalam meningkatkan kenyamanan dan produktivitas pengguna (Wang dkk., 2021). Berdasarkan ambang toleransi respon pengguna, respons aplikasi idealnya berada di bawah sepuluh detik agar tidak menimbulkan frustrasi (Nielsen, 2025).

Di ranah arsitektur antarmuka pengguna, terdapat dua pendekatan utama, yakni *server-side rendering* (SSR) dan *client-side rendering* (CSR). SSR melakukan *rendering* aplikasi pada sisi server dan mengirim dokumen HTML utuh kepada klien, sedangkan CSR merender aplikasi langsung di dalam browser menggunakan JavaScript, sehingga *Document Object Model* dimodifikasi di sisi klien (Osmani & Miller, 2019). *Server-side rendering* dapat mempercepat waktu muat awal serta meningkatkan keterjangkauan mesin pencari karena berkas HTML dikirim utuh dari sisi server, sedangkan *client-side rendering* memindahkan proses *rendering* ke browser melalui JavaScript sehingga interaksi pengguna menjadi sangat dinamis. Namun, pendekatan ini umumnya membutuhkan waktu muat awal lebih lama (Pavić & Brkić, 2021).

Menurut dokumentasi resmi Laravel (*Laravel*, 2025), framework ini menyediakan dua jalur resmi untuk membangun antarmuka pengguna, yakni memanfaatkan PHP bersama Blade atau JavaScript modern seperti React dan Vue melalui Inertia. Pendekatan pertama, *server-side rendering* dengan Blade, melakukan *rendering* HTML sepenuhnya di sisi server sehingga setiap interaksi biasanya memicu pemuatan ulang halaman. Pendekatan kedua memanfaatkan JavaScript modern seperti React atau Vue melalui Inertia yang menghubungkan *route* dan *controller* Laravel dengan komponen satu halaman React atau Vue dalam satu repositori kode.

Penelitian sebelumnya menegaskan bahwa penggunaan framework modern seperti React dan Vue dalam aplikasi berbasis Laravel secara signifikan meningkatkan performa dan pengalaman pengguna dibandingkan pendekatan tradisional seperti Blade. Studi menunjukkan bahwa Vue menghasilkan skor tinggi pada pengujian performa menggunakan Lighthouse, khususnya dengan First Contentful Paint (FCP) sekitar 0,870 detik dan waktu Time to First Byte (TTFB) tercepat yaitu 0,179 detik, yang mengindikasikan efisiensi yang tinggi dalam *rendering* awal halaman (Cen & Nusantara, 2024).

Lebih lanjut, penelitian yang membandingkan pendekatan *server-side rendering* dengan *client-side rendering* secara langsung menemukan bahwa SSR cenderung unggul dalam kecepatan render awal, ukuran dokumen HTML yang lebih besar akibat tersematnya *script* JavaScript justru membantu mempercepat proses rendering di browser, meningkatkan skor Google Lighthouse secara keseluruhan dibanding CSR (Bhanuartha dkk., 2025). Namun, CSR dengan React dan Vue tetap unggul dalam interaktivitas dan responsivitas setelah halaman dimuat sepenuhnya, meskipun memerlukan waktu lebih lama pada muatan awal karena proses render berlangsung di sisi klien (Pavić & Brkić, 2021).

Penelitian komparatif yang dilakukan oleh (Rathinam, 2023) menegaskan keunggulan React yang didukung oleh *Virtual Document Object Model* atau *Virtual* DOM dalam mengelola interaksi pengguna secara dinamis dengan efisiensi penggunaan memori yang lebih baik. Hal ini selaras dengan temuan (Ollila dkk., 2022) yang menunjukkan bahwa strategi rendering berbasis *Virtual DOM* seperti yang digunakan React dan Vue cenderung memiliki skalabilitas performa yang baik seiring meningkatnya kompleksitas aplikasi, dengan efisiensi optimal dalam manajemen perubahan elemen *DOM*.

Selain itu, studi kasus implementasi Laravel bersama Vue dalam aplikasi registrasi pasien membuktikan bahwa pendekatan ini efektif dalam mengelola data secara dinamis dengan responsivitas tinggi, mendukung pengembangan aplikasi dengan efisiensi tim dan biaya yang optimal (Seputra & Dewi, 2022). Dengan demikian, keputusan memilih pendekatan *rendering* dalam Laravel antara menggunakan Blade atau Vue dan React harus mempertimbangkan secara cermat

jenis interaksi pengguna, karakteristik data, serta skala aplikasi yang dikembangkan guna mencapai keseimbangan antara kecepatan muat awal dan dinamisme interaksi pengguna.

Permasalahan yang diangkat dalam penelitian ini diperkuat oleh kondisi kritis yang terjadi pada aplikasi Sidang Online FRI (SOFI) terutama pada periode puncak penggunaan sistem di tengah tahun, saat intensitas penggunaannya meningkat drastis seiring dengan aktivitas akademik. Situasi ini semakin kompleks dengan tingginya frekuensi *troubleshooting*, yang tercatat sebanyak lima kali selama periode 2022 hingga 2024, menunjukkan bahwa arsitektur yang ada tidak mampu mempertahankan performa yang stabil ketika menghadapi beban data besar dalam waktu bersamaan. Kondisi ini secara langsung berdampak negatif terhadap produktivitas pengguna, terutama bagi administrator yang secara intensif menggunakan halaman dokumen sidang. Permasalahan tersebut menuntut adanya analisis teknis yang mendalam terhadap alternatif pendekatan *rendering* yang lebih efisien guna mengatasi kelambatan sistem yang telah terbukti menghambat operasional akademik secara signifikan.

Untuk memvalidasi temuan kualitatif tersebut dan mengukur tantangan ini secara kuantitatif, sebuah pengujian dilakukan dengan mereplikasi kondisi beban data aktual. Hasil pengujian performa pada sistem berbasis Blade menggunakan Google Lighthouse menunjukkan adanya masalah yang serius dan terukur, dengan First Contentful Paint (FCP) tercatat pada 8,6 detik, Largest Contentful Paint (LCP) mencapai 8,8 detik, Total Blocking Time (TBT) sebesar 3.690 milidetik, dan Speed Index pada 16,4 detik. Data empiris ini membuktikan adanya masalah performa yang serius. Meskipun waktu muat konten utama (LCP 8,8 detik) nyaris menyentuh batas kritis toleransi pengguna 10 detik, metrik lain menunjukkan gambaran yang lebih mengkhawatirkan. Nilai Total Blocking Time yang sangat tinggi (3.690 ms) mengindikasikan bahwa halaman menjadi tidak responsif untuk waktu yang lama setelah konten terlihat, sementara Speed Index (16,4 detik) menegaskan bahwa persepsi visual pengguna terhadap kecepatan muat halaman secara keseluruhan sangat lambat.

Temuan kuantitatif di atas menjadi justifikasi kuat untuk mengevaluasi pendekatan teknik rendering alternatif. Oleh karena itu, penelitian ini dirancang untuk menguji apakah *client-side rendering* menggunakan React dan Vue dapat memberikan performa muat halaman (diukur dengan FCP, LCP, Speed Index) dan stabilitas serta efisiensi interaktivitas awal halaman (diukur dengan Total Blocking Time) yang lebih baik dibandingkan Blade, khususnya pada halaman dengan volume data besar.

I.2 Rumusan Masalah

Berdasarkan permasalahan yang telah diuraikan pada latar belakang, maka rumusan masalah dalam penelitian ini dapat dijabarkan sebagai berikut:

a. Pendekatan *server-side rendering* (SSR) menggunakan Blade pada aplikasi Sidang Online FRI (SOFI) menunjukkan penurunan performa signifikan ketika memuat halaman dengan volume data besar, khususnya pada halaman dokumen sidang, yang berdampak pada lamanya waktu muat dan rendahnya stabilitas interaktivitas.
b. Hasil pengujian performa aplikasi *existing* menggunakan Google Lighthouse memperlihatkan bahwa metrik-metrik seperti First Contentful Paint (FCP), Largest Contentful Paint (LCP), Total Blocking Time (TBT), dan Speed Index (SI) melebihi batas toleransi kenyamanan pengguna, sehingga mengindikasikan kebutuhan akan arsitektur rendering yang lebih efisien.
c. Alternatif pendekatan *client-side rendering* (CSR) melalui framework React dan Vue menjanjikan peningkatan responsivitas, namun belum terdapat pembuktian empiris secara langsung dalam konteks aplikasi manajemen sidang akademik dengan data berskala besar.
d. Diperlukan pengujian komparatif yang sistematis untuk mengetahui apakah pendekatan CSR mampu memberikan performa waktu muat dan interaktivitas awal yang lebih baik dibanding SSR, sehingga dapat dijadikan dasar pemilihan arsitektur antarmuka pengguna yang paling sesuai untuk sistem akademik seperti SOFI.

I.3 Tujuan Tugas Akhir

Berdasarkan latar belakang dan rumusan masalah diatas, penelitian ini bertujuan untuk:

a. Mengidentifikasi dan mengukur performa aktual halaman dokumen sidang pada aplikasi SOFI yang dibangun menggunakan server-side rendering (Blade) dalam menghadapi volume data besar, khususnya terkait waktu muat awal dan stabilitas interaktivitas.
b. Menganalisis hasil pengujian metrik performa seperti First Contentful Paint (FCP), Largest Contentful Paint (LCP), Total Blocking Time (TBT), dan Speed Index (SI) sebagai indikator untuk mengevaluasi tingkat efisiensi dan kenyamanan pengguna.
c. Membangun dua varian alternatif sistem dengan pendekatan client-side rendering menggunakan React dan Vue, serta menguji performanya secara kuantitatif dalam konteks aplikasi sidang berbasis Laravel.
d. Membandingkan ketiga pendekatan rendering (Blade, React, dan Vue) secara komprehensif untuk menentukan strategi antarmuka pengguna yang paling optimal dalam menangani halaman dengan data skala besar pada sistem manajemen sidang akademik.

I.4 Manfaat Tugas Akhir

Manfaat penelitian ini:

1. Bagi institusi pendidikan tinggi yang mengelola sistem akademik, penelitian ini bermanfaat dalam memberikan dasar evaluatif untuk memilih pendekatan rendering yang paling efisien dalam menangani data besar, sehingga dapat meningkatkan performa sistem dan kepuasan pengguna
2. Bagi pengembang aplikasi, penelitian ini bermanfaat sebagai referensi teknis dalam menentukan strategi *frontend* yang sesuai berdasarkan kebutuhan performa, skala data, dan karakteristik sistem yang dikembangkan
3. Bagi peneliti lain yang bergerak di bidang performa aplikasi web, penelitian ini bermanfaat sebagai studi awal yang mengkaji pengaruh

pendekatan rendering terhadap performa sistem dalam konteks arsitektur monolitik Laravel.

I.5 Batasan dan Asumsi Tugas Akhir

Batasan Tugas Akhir:

1. Penelitian ini hanya dilakukan pada satu sistem studi kasus, yaitu aplikasi SOFI yang digunakan di lingkungan Fakultas Rekayasa Industri.
2. Pendekatan rendering yang diuji terbatas pada Blade (server-side rendering), React, dan Vue melalui Inertia.js (client-side rendering), tanpa melibatkan framework *frontend* lain.
3. Sistem yang diuji merupakan replikasi dari satu fitur inti aplikasi SOFI, yaitu halaman dokumen sidang pada antarmuka administrator, dan bukan replikasi dari keseluruhan fungsionalitas aplikasi.
4. Pengujian dilakukan dalam skenario pengguna tunggal dengan memuat data statis (10 hingga 10.000 entri).
5. Evaluasi performa hanya berdasarkan metrik kuantitatif dari Google Lighthouse (FCP, LCP, TBT, Speed Index) dan tidak mencakup analisis kualitatif seperti persepsi pengguna, maupun analisis sisi server seperti penggunaan memori atau CPU.

Untuk menjaga validitas hasil penelitian, beberapa asumsi telah ditetapkan. Diasumsikan bahwa ketiga versi sistem (Blade, React, dan Vue) beroperasi di atas arsitektur backend Laravel yang sepenuhnya identik dan stabil, sehingga setiap variasi performa yang terukur dapat diatribusikan secara eksklusif pada teknologi *frontend* yang dievaluasi. Selain itu, konsistensi infrastruktur pengujian, termasuk hardware, browser, dan jaringan, dijaga secara ketat untuk mengeliminasi potensi bias pada hasil akhir. Sebagai tolok ukur evaluasi, penelitian ini mengadopsi standar waktu muat halaman maksimal 10 detik sebagai ambang toleransi kenyamanan pengguna, sebuah prinsip yang merujuk pada studi usability oleh Jakob Nielsen.

I.6 Sistematika Laporan

Laporan tugas akhir ini disusun secara sistematis dalam enam bab utama yang saling terintegrasi guna mempermudah pembaca dalam memahami alur pemikiran, proses penelitian, serta hasil analisis yang diperoleh. Adapun sistematika penulisan laporan ini adalah sebagai berikut:

1. Bab I Pendahuluan

   Bab ini menjelaskan latar belakang masalah, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan dan asumsi, serta sistematika penulisan laporan.
2. Bab II Landasan Teori

   Bab ini menguraikan teori-teori dan konsep yang relevan dengan pendekatan rendering pada aplikasi web, termasuk perbandingan antara *server-side rendering* dan *client-side rendering*. Selain itu, dibahas pula framework yang digunakan seperti Blade, React, dan Vue.
3. Bab III Metode Penyelesaian Masalah

   Bab ini menjelaskan pendekatan sistematis yang digunakan untuk menyelesaikan permasalahan penelitian. Dibahas secara rinci mengenai metode pengumpulan data, pengolahan data, dan metode evaluasi.
4. Bab IV Penyelesaian Masalah

   Bab ini membahas pelaksanaan penelitian secara detail berdasarkan tahapan sistematika penelitian yang telah dirancang sebelumnya. Uraian mencakup tahap pengumpulan dan analisis data, perancangan eksperimen, serta tahap pengembangan eksperimen.
5. Bab V Analisis Hasil dan Implikasi

   Bab ini membahas hasil pengujian secara lebih mendalam. Analisis difokuskan pada stabilitas dan efisiensi tiap pendekatan serta implikasi teknis yang relevan bagi pengembangan aplikasi serupa di masa depan.
6. Bab VI Kesimpulan dan Saran

   Bab ini menyajikan kesimpulan dari penelitian dan menjawab rumusan masalah yang telah ditetapkan. Selain itu, disampaikan pula saran-saran untuk pengembangan selanjutnya serta peluang studi lanjutan yang dapat dikembangkan dari hasil penelitian ini.