# BAB I: PENDAHULUAN

## 1.1 Latar Belakang

Perkembangan teknologi web modern telah membawa perubahan signifikan dalam cara aplikasi web dirancang dan dikembangkan. Framework seperti Next.js menyediakan berbagai pendekatan rendering, seperti Server-Side Rendering (SSR), Client-Side Rendering (CSR), Static Site Generation (SSG), dan Incremental Static Regeneration (ISR), yang memungkinkan pengembang untuk menyesuaikan metode rendering sesuai dengan kebutuhan aplikasi. Setiap metode memiliki kelebihan dan kekurangannya masing-masing, baik dalam aspek performa, efisiensi sumber daya, maupun optimasi Search Engine Optimization (SEO).

Teknologi rendering ini sangat relevan dalam konteks peningkatan pengalaman pengguna dan pengoptimalan SEO, terutama di era di mana kecepatan akses dan visibilitas mesin pencari menjadi faktor utama dalam kesuksesan aplikasi web. Penelitian sebelumnya telah menunjukkan bahwa SSR unggul dalam hal optimasi SEO dan waktu muat awal (*initial page load*) dibandingkan CSR. Di sisi lain, CSR menawarkan fleksibilitas tinggi untuk aplikasi yang sangat dinamis. Metode terbaru seperti SSG dan ISR memperkenalkan efisiensi tinggi dalam rendering halaman statis dan pembaruan konten dinamis tanpa kehilangan kecepatan akses.

Namun, terdapat kesenjangan penelitian terkait evaluasi kuantitatif dari keempat metode ini, terutama dalam konteks framework Next.js yang mendukung pendekatan hybrid. Oleh karena itu, penelitian ini bertujuan untuk menganalisis performa SSR, CSR, SSG, dan ISR, serta memberikan panduan kepada pengembang dalam memilih metode yang paling sesuai untuk kebutuhan aplikasi web modern. Hasil dari penelitian ini diharapkan dapat memberikan kontribusi dalam pengembangan aplikasi web yang lebih efisien dan responsif terhadap kebutuhan pengguna.

## 1.2 Rumusan Masalah

1. Bagaimana perbandingan performa SSR, CSR, SSG, dan ISR pada framework Next.js ditinjau dari metrik seperti waktu muat halaman, konsumsi sumber daya, dan pengalaman pengguna?
2. Sejauh mana SSG dan ISR dapat mengoptimalkan SEO dibandingkan SSR dan CSR?
3. Apa saja skenario penggunaan terbaik untuk setiap metode rendering dalam pengembangan aplikasi web?

## 1.3 Tujuan Penelitian

* Mengevaluasi kelebihan dan kekurangan SSR, CSR, SSG, dan ISR dalam konteks performa, efisiensi, dan optimasi SEO.
* Memberikan panduan bagi pengembang dalam memilih metode rendering berdasarkan kebutuhan aplikasi web modern.
* Mengidentifikasi pengaruh metode rendering terhadap pengalaman pengguna dan efisiensi sumber daya server.

## 1.4 Manfaat Penelitian

1. **Manfaat Teoritis**: Menambah literatur ilmiah terkait evaluasi metode rendering pada framework modern seperti Next.js, memberikan wawasan baru terkait optimalisasi kinerja aplikasi web.
2. **Manfaat Praktis**: Memberikan rekomendasi praktis kepada pengembang untuk menentukan metode rendering terbaik berdasarkan kebutuhan spesifik aplikasi mereka. Penelitian ini juga diharapkan dapat membantu pengembang dalam merancang aplikasi web yang lebih responsif, efisien, dan sesuai dengan kebutuhan bisnis.

# BAB II: TINJAUAN PUSTAKA

## 2.1 Teknologi Web Modern

Framework modern seperti Next.js dirancang untuk mengatasi tantangan dalam pengembangan aplikasi web yang cepat, responsif, dan ramah SEO. Pendekatan rendering yang didukung Next.js, yaitu SSR, CSR, SSG, dan ISR, menjadi solusi utama dalam berbagai skenario pengembangan aplikasi. Framework ini mengintegrasikan fleksibilitas dengan performa tinggi, menjadikannya pilihan utama bagi pengembang web saat ini.

## 2.2 Server-Side Rendering (SSR)

SSR adalah proses di mana halaman web dirender di server sebelum dikirimkan ke klien. Keuntungan utama SSR adalah waktu muat awal yang cepat dan optimasi SEO yang baik karena konten lengkap dapat diindeks oleh mesin pencari. Namun, pendekatan ini dapat membebani server jika terdapat lonjakan permintaan pengguna.

* **Referensi**: Artikel di Towards Data Science menyebutkan bahwa SSR sangat efektif untuk aplikasi yang membutuhkan optimasi SEO tinggi【18†source】.

## 2.3 Client-Side Rendering (CSR)

CSR merender halaman web di sisi klien menggunakan JavaScript. Pendekatan ini memungkinkan pengalaman pengguna yang dinamis, tetapi waktu muat awal sering kali lebih lambat karena harus memuat JavaScript terlebih dahulu sebelum halaman ditampilkan.

* **Referensi**: Artikel Flatirons menguraikan bahwa CSR lebih cocok untuk aplikasi interaktif dengan konten dinamis【20†source】.

## 2.4 Static Site Generation (SSG)

SSG merender halaman web selama waktu build, menghasilkan konten statis yang siap dikirim ke klien. Ini meningkatkan waktu muat halaman untuk konten yang jarang berubah dan mengurangi beban server secara signifikan.

* **Referensi**: Dokumentasi resmi Next.js menyebutkan bahwa SSG sangat cocok untuk situs dengan konten statis.

## 2.5 Incremental Static Regeneration (ISR)

ISR memungkinkan pembaruan konten dinamis pada situs statis dengan interval tertentu, menjembatani kelemahan SSG tanpa kehilangan performa. Fitur ini memungkinkan halaman statis untuk tetap relevan meskipun terdapat perubahan konten.

* **Referensi**: ISR adalah fitur unik Next.js untuk mendukung pembaruan konten dinamis secara efisien.

## 2.6 Penelitian Terkait

Beberapa penelitian telah mengevaluasi SSR dan CSR, tetapi sedikit yang membahas perbandingan SSG dan ISR dalam konteks framework modern. Penelitian ini bertujuan mengisi gap tersebut dengan mengevaluasi keempat metode rendering secara bersamaan untuk mendapatkan gambaran yang lebih komprehensif.

# BAB III: METODOLOGI PENELITIAN

## 3.1 Metode Penelitian

Penelitian ini menggunakan metode eksperimental untuk mengevaluasi performa dan optimasi SEO dari SSR, CSR, SSG, dan ISR pada framework Next.js. Metode ini dipilih karena memungkinkan pengukuran kuantitatif yang akurat terhadap metrik yang relevan.

## 3.2 Tahapan Penelitian

1. **Studi Literatur**: Mengkaji referensi terkait teknologi rendering.
2. **Implementasi**: Membuat prototipe aplikasi dengan keempat metode rendering menggunakan Next.js.
3. **Pengujian**: Melakukan pengujian kuantitatif terhadap performa dan SEO.
4. **Analisis Data**: Mengolah hasil pengujian untuk memberikan wawasan yang dapat digunakan dalam pengembangan aplikasi web.

## 3.3 Alat dan Bahan

* **Framework**: Next.js
* **Alat Pengujian**: Lighthouse, Google Search Console, WebPageTest
* **Server Hosting**: Vercel
* **Perangkat Pendukung**: Alat monitoring server seperti New Relic.

## 3.4 Metrik Pengujian

1. **Waktu Muat Halaman**: Diukur menggunakan Lighthouse.
2. **Efisiensi SEO**: Skor SEO diukur menggunakan Google Search Console.
3. **Konsumsi Sumber Daya**: Analisis beban server menggunakan alat monitoring seperti New Relic.
4. **Pengalaman Pengguna**: Analisis klik dan waktu interaksi pengguna.

## 3.5 Rencana Implementasi

Penelitian ini akan dilakukan dalam beberapa tahap dengan fokus pada pengujian performa dan efisiensi masing-masing metode rendering. Setiap metode akan diuji dalam kondisi yang sama untuk memastikan hasil yang konsisten dan dapat dibandingkan.