Meningkatkan Performa Frontend dengan Menggunakan Framework Next.Js dalam Pengembangan Website

Rakhmat Prasetyo Agung Nugroho[1]*, Andi Sugandi[2]
[1, 2] Department of Informatic, Universitas Siber Muhammadiyah, Yogyakarta, Indonesia
Email: [1] agungprasetyo@sibermu.ac.id, [2] andi@sibermu.ac.id

***Abstract*—Penelitian ini mengeksplorasi dampak penggunaan framework Next.js terhadap performa frontend website. Next.js, dengan fitur-fitur seperti server-side rendering (SSR) dan static site generation (SSG), menawarkan potensi peningkatan signifikan dalam kecepatan pemuatan halaman dan pengalaman pengguna. Studi ini membandingkan performa website yang dibangun dengan Next.js terhadap website yang dibangun dengan pendekatan tradisional (menggunakan Bootstrap), mengukur metrik kunci seperti waktu pemuatan halaman, ukuran file, dan skor Lighthouse. Hasil penelitian menunjukkan peningkatan yang signifikan sebesar 62.5% dalam performa website Next.js. Temuan ini memberikan bukti empiris tentang manfaat Next.js dalam pengembangan website yang berfokus pada performa.**

Keywords—Next.js; performa frontend; performa website; server-side rendering (SSR); static site generation (SSG)

I. PENDAHULUAN

Perkembangan teknologi informasi dan komunikasi telah membawa perubahan signifikan dalam cara kita mengembangkan dan mengelola aplikasi web. Dalam lanskap digital yang kompetitif saat ini, performa website merupakan faktor krusial dalam menentukan keberhasilan sebuah situs. Pengguna mengharapkan pengalaman online yang cepat, responsif, dan efisien. Website yang lambat dapat menyebabkan tingkat bounce rate yang tinggi, penurunan konversi, dan kerugian pendapatan. Oleh karena itu, optimasi performa frontend menjadi prioritas utama bagi pengembang web [1].

Framework JavaScript modern seperti Next.js telah muncul sebagai solusi yang efektif untuk meningkatkan performa frontend dalam pengembangan website. Next.js, yang dibangun di atas React, menawarkan berbagai fitur canggih seperti rendering sisi server (SSR), pengelolaan rute yang efisien, dan pengoptimalan performa yang membantu pengembang menciptakan aplikasi web yang responsif dan cepat. SSR merender halaman di server sebelum dikirim ke browser, menghasilkan waktu pemuatan yang lebih cepat, terutama pada halaman pertama. Sementara itu, static site generation (SSG) menghasilkan halaman statis pada saat build, memungkinkan waktu muat yang sangat cepat dan meningkatkan SEO [2].

Sebagai perbandingan, Bootstrap, meskipun menyediakan komponen siap pakai dan responsif, mungkin kurang efisien dalam hal optimasi performa untuk aplikasi yang kompleks dan interaktif. Tailwind CSS menjadi alternatif populer yang menawarkan pendekatan berbasis utility-first dalam pengembangan frontend, memberikan fleksibilitas dan efisiensi dalam styling tanpa bergantung pada komponen pra-desain yang kaku [3]. Beberapa penelitian telah menunjukkan efektivitas penggunaan Tailwind CSS dalam meningkatkan performa frontend, baik dalam proyek berbasis Next.js maupun dalam perbandingan dengan framework lain seperti Bootstrap [4]-[8]. Dengan meningkatnya permintaan akan aplikasi web yang mampu memberikan pengalaman pengguna yang optimal, penggunaan Next.js dan Tailwind CSS menjadi semakin relevan dalam industri pengembangan perangkat lunak.

Meskipun Next.js menawarkan berbagai keuntungan, banyak pengembang yang masih mengalami kesulitan dalam memanfaatkan fitur-fitur yang ada untuk mencapai performa optimal. Beberapa masalah yang sering dihadapi termasuk waktu muat halaman yang lambat, penggunaan sumber daya yang tidak efisien, serta tantangan dalam mengelola konten dinamis [9]. Selain itu, kurangnya pemahaman tentang teknik pengoptimalan yang tepat dapat mengakibatkan aplikasi yang tidak memenuhi harapan pengguna. Oleh karena itu, penting untuk mengeksplorasi bagaimana Next.js dapat digunakan secara efektif untuk mengatasi masalah ini dan meningkatkan performa frontend secara keseluruhan.

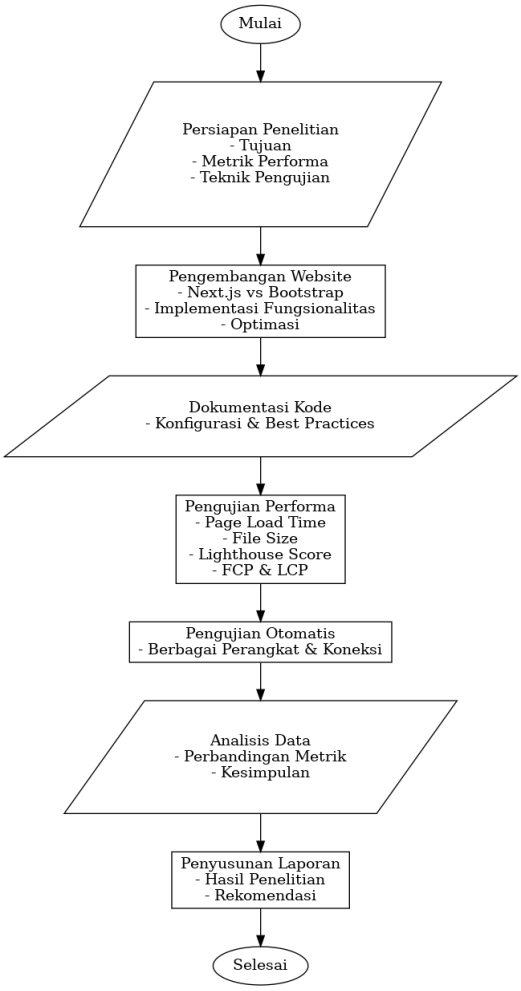
Penelitian ini bertujuan untuk menganalisis dan mengevaluasi efektivitas penggunaan framework Next.js dalam pengembangan website untuk meningkatkan performa frontend. Secara khusus, penelitian ini berfokus pada identifikasi fitur-fitur utama Next.js yang berkontribusi terhadap peningkatan performa, eksplorasi teknik-teknik pengoptimalan yang dapat diterapkan dalam pengembangan menggunakan Next.js, serta perbandingan performa website yang dibangun dengan Next.js dengan website yang dibangun dengan Bootstrap dan Tailwind CSS menggunakan metrik performa yang terukur dan terstandarisasi. Selain itu, penelitian ini juga bertujuan untuk memberikan rekomendasi praktis bagi pengembang dalam memanfaatkan Next.js untuk menciptakan aplikasi web yang lebih responsif dan efisien [10].

Kontribusi dari penelitian ini diharapkan dapat memberikan pemahaman yang lebih mendalam tentang penggunaan Next.js dan Tailwind CSS dalam pengembangan aplikasi web. Dengan mengidentifikasi dan menganalisis

fitur-fitur serta teknik pengoptimalan yang efektif, penelitian ini akan memberikan panduan praktis bagi pengembang dalam meningkatkan performa frontend aplikasi mereka. Selain itu, hasil penelitian ini dapat menjadi referensi bagi penelitian lebih lanjut yang berfokus pada pengembangan web dan penggunaan framework modern dalam konteks yang lebih luas. Dengan demikian, penelitian ini tidak hanya akan bermanfaat bagi pengembang, tetapi juga bagi akademisi dan praktisi yang tertarik dalam pengembangan aplikasi berbasis web [11].

II. METODE PENELITIAN

a. Penelitian ini menggunakan pendekatan eksperimental komparatif untuk membandingkan performa dua kelompok website: satu kelompok dibangun menggunakan Next.js dan kelompok lainnya dibangun menggunakan Bootstrap. Berikut alur kerja dari penelitian ini seperti pada Gambar 1.



b.

Gambar 1. Flowchart Perbandingan Performa Website (Next.js vs Bootstrap).

Gambar 1 ini menggambarkan proses penelitian dan evaluasi performa frontend website yang dikembangkan menggunakan **Next.js** dan **Bootstrap**. Berikut adalah tahapan utama dalam alur kerja penelitian:

- **Mulai**
  Proses dimulai dengan menentukan tujuan penelitian, yaitu menganalisis dan mengevaluasi performa frontend menggunakan Next.js dan Bootstrap.
- **Studi Literatur**
  Mengumpulkan referensi dari jurnal dan penelitian sebelumnya yang membahas pengembangan frontend dengan Next.js dan Bootstrap untuk memahami kelebihan dan kekurangannya.
- **Perancangan Website**
  Membuat desain dan struktur dasar website yang akan dikembangkan. Menentukan fitur utama yang akan diuji dalam perbandingan performa.
- **Pengembangan Website dengan Next.js**
  Implementasi website menggunakan framework Next.js. Menggunakan teknik optimasi seperti Server-Side Rendering (SSR) dan Static Site Generation (SSG) untuk meningkatkan performa.
- **Pengembangan Website dengan Bootstrap**
  Implementasi website menggunakan framework Bootstrap. Menggunakan komponen standar dan teknik optimasi yang umum digunakan dalam Bootstrap.
- **Pengujian dan Evaluasi Performa**
  Mengukur performa kedua website menggunakan metrik seperti waktu muat halaman (load time), First Contentful Paint (FCP), Largest Contentful Paint (LCP), dan Total Blocking Time (TBT). Penggunaan alat seperti Lighthouse atau PageSpeed Insights untuk analisis performa.
- **Analisis Hasil dan Perbandingan**
  Membandingkan hasil pengujian performa antara website berbasis Next.js dan Bootstrap. Kemudian melakukan identifikasi kelebihan dan kekurangan masing-masing framework berdasarkan hasil pengujian.
- **Kesimpulan dan Rekomendasi**
  Menarik kesimpulan dari hasil penelitian. Memberikan rekomendasi bagi pengembang web terkait penggunaan Next.js atau Bootstrap berdasarkan skenario pengembangan yang sesuai.
- **Selesai**
  Penelitian berakhir setelah semua analisis dan kesimpulan dibuat.

Selain itu, Gambar 1 memberikan penjelasan bahwa kedua kelompok website dirancang dengan fungsionalitas yang serupa untuk memastikan perbandingan yang adil. Contoh fungsionalitas yang diimplementasikan meliputi halaman login, dashboard, dan profil pengguna.

Pengembangan kedua kelompok website dibangun dengan mengikuti best practices untuk masing-masing framework. Hal ini termasuk optimasi gambar, minifikasi kode, dan penggunaan teknik caching yang sesuai. Kode sumber dari kedua kelompok website didokumentasikan dengan baik untuk memastikan transparansi dan reproduksibilitas.

A. Analisis Performa

Proses evaluasi efisiensi dan responsivitas sebuah website, beberapa metrik performa penting perlu diukur. Metrik-metrik ini meliputi Waktu Pemuatan Halaman (Page

Load Time) yang mengukur kecepatan website dalam memuat konten secara keseluruhan. Ukuran File (File Size) juga menjadi perhatian, di mana ukuran total file yang diunduh oleh pengguna diukur untuk menilai seberapa efisien optimasi aset website. Selain itu, Skor Lighthouse digunakan untuk melakukan pengujian performa menyeluruh. Skor ini mencakup berbagai aspek performa seperti Performance, Accessibility, Best Practices, SEO, dan PWA, memberikan gambaran komprehensif tentang kualitas dan performa website. Fokus utama dari analisis ini adalah pada skor Performance Lighthouse. Terakhir, metrik First Contentful Paint (FCP) dan Largest Contentful Paint (LCP) mengukur waktu hingga konten pertama dan elemen konten terbesar muncul di layar, memberikan informasi tentang pengalaman pengguna dalam melihat konten website..

B. Teknik Pengujian

Pengujian Otomatis (Automated Testing): Digunakan untuk mengukur waktu pemuatan halaman, ukuran file, dan skor Lighthouse secara otomatis dan berulang. Pengujian ini dilakukan pada berbagai perangkat dan koneksi internet untuk memastikan hasil yang komprehensif.

III. HASIL DAN PEMBAHASAN

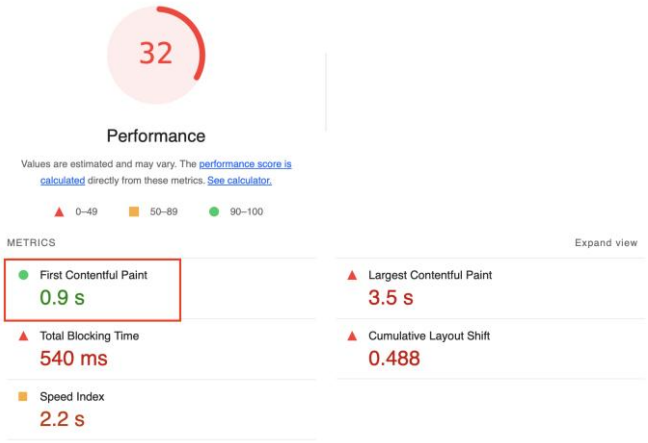
A. Analisis Performa Website Sebelum dan Sesudah Implementasi Next.js

Evaluasi performa website sebelum dan sesudah penerapan Next.js dilakukan dengan mengukur berbagai metrik yang relevan menggunakan Lighthouse. Parameter utama yang dianalisis adalah First Contentful Paint (FCP), Largest Contentful Paint (LCP), dan Speed Index, yang semuanya berkontribusi dalam menentukan pengalaman pengguna ketika mengakses sebuah website.

- *First Contentful Paint (FCP)*

First Contentful Paint (FCP) adalah metrik yang mengukur waktu yang dibutuhkan untuk menampilkan elemen pertama dari halaman yang sedang dimuat. FCP sangat penting karena menentukan kapan pengguna pertama kali mendapatkan umpan balik visual dari website, sehingga dapat mengurangi persepsi lambatnya pemuatan halaman.

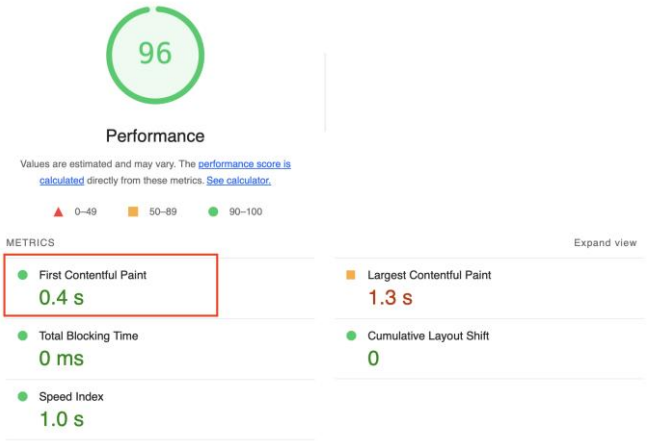
Sebelum implementasi Next.js, homepage Tradex memiliki FCP sebesar 903.8 ms seperti pada Gambar 2.



Gambar 2. Hasil FCP sebelum evaluasi performa Next.js

Gambar 2 menunjukkan bahwa pengguna harus menunggu hampir satu detik sebelum melihat konten awal muncul. Meskipun angka ini masih dalam kategori yang dapat diterima, penundaan sekecil apa pun dalam FCP dapat berdampak negatif terhadap pengalaman pengguna, terutama bagi mereka yang mengakses website dengan koneksi internet yang kurang stabil.

Setelah implementasi Next.js pada website ondewe, FCP mengalami penurunan drastis menjadi 373.7 ms seperti pada Gambar 3.



Gambar 3. Hasil FCP setelah evaluasi performa Next.js

Gambar 3 menjelaskan penurunan yang menunjukkan bahwa Next.js dapat mengoptimalkan rendering awal dengan lebih baik dibandingkan pendekatan konvensional. Hal ini kemungkinan besar disebabkan oleh penggunaan Static Site Generation (SSG), yang memungkinkan konten statis dimuat lebih cepat, serta peningkatan efisiensi dalam manajemen sumber daya yang digunakan oleh browser saat memproses halaman.

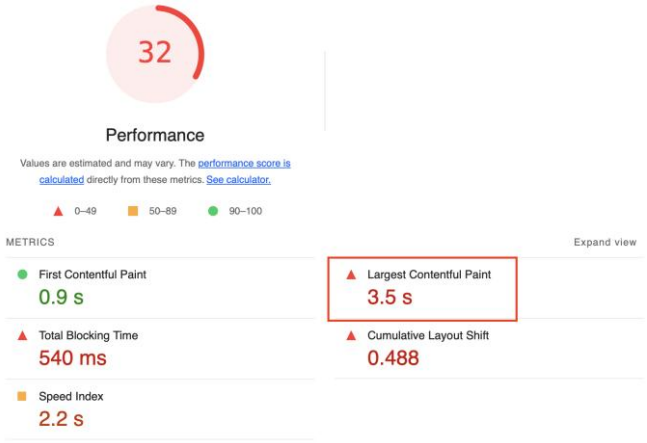
Dampak dari peningkatan FCP yang lebih rendah ini sangat signifikan bagi pengalaman pengguna, karena pengguna dapat melihat konten lebih cepat dan merasa

website lebih responsif. Kecepatan ini juga membantu dalam meningkatkan tingkat retensi pengguna, karena mereka lebih cenderung bertahan di halaman yang memberikan respons visual lebih cepat dibandingkan dengan yang membutuhkan waktu lebih lama untuk menampilkan elemen pertama.

- *Largest Contentful Paint (FCP)*

Largest Contentful Paint (LCP) adalah metrik yang mengukur waktu yang dibutuhkan untuk menampilkan elemen terbesar dalam viewport pengguna, biasanya berupa gambar hero, teks utama, atau elemen interaktif lainnya. Metrik ini sangat penting karena berhubungan langsung dengan persepsi kecepatan pemuatan halaman bagi pengguna.

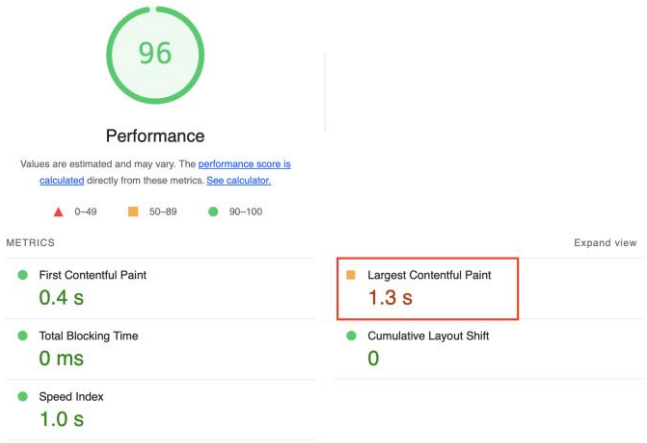
Sebelum implementasi Next.js, LCP pada homepage Tradex mencapai 3466.8 ms seperti pada Gambar 4.



Gambar 4. Hasil LCP sebelum evaluasi performa Next.js

Gambar 4 memberikan informasi LCP pada homepage Tradex mencapai 3466.8 ms, yang berarti elemen terbesar membutuhkan waktu lebih dari 3,4 detik untuk muncul sepenuhnya. Hal ini dapat menyebabkan pengalaman pengguna yang kurang optimal, terutama bagi pengguna yang mengharapkan website responsif dan cepat.

Setelah implementasi Next.js di ondewe, LCP mengalami peningkatan signifikan dengan waktu pemuatan hanya 1301.5 ms seperti pada Gambar 5.



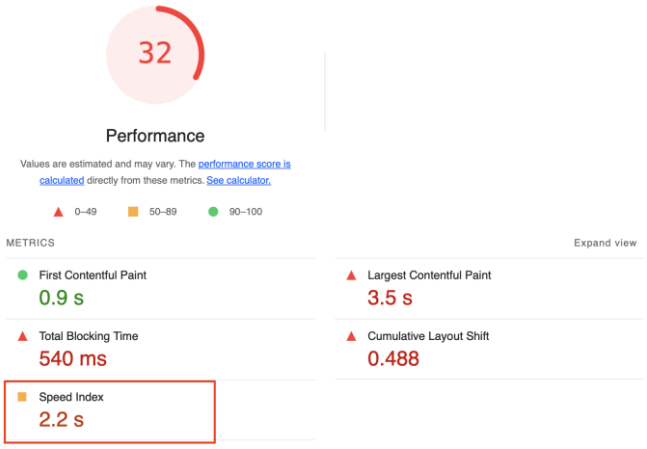
Gambar 5. Hasil LCP setelah evaluasi performa Next.js

Pada Gambar 5 dapat disimpulkan bahwa percepatan yang terjadi sebesar 62.5%. Peningkatan ini terutama disebabkan oleh optimasi server-side rendering (SSR) dan teknik image optimization, yang memungkinkan gambar dimuat lebih efisien tanpa mengorbankan kualitas tampilan.

Dengan LCP yang lebih rendah, pengguna dapat melihat elemen utama lebih cepat, yang meningkatkan persepsi kecepatan website dan mengurangi tingkat bounce rate. Website dengan LCP tinggi sering kali menyebabkan pengguna meninggalkan halaman sebelum elemen utama selesai dimuat, sehingga meningkatkan LCP merupakan faktor penting dalam optimasi UX/UI.

- *Speed Index*

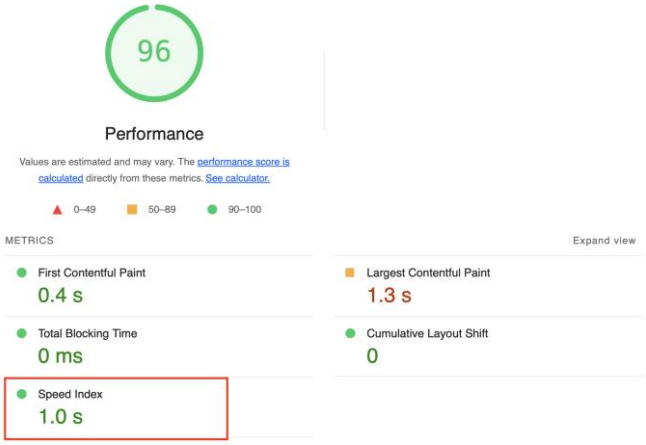
Speed Index adalah metrik yang mengukur seberapa cepat konten halaman terlihat oleh pengguna selama proses pemuatan. Metrik ini menggambarkan pengalaman pengguna dalam melihat tampilan halaman yang sedang dimuat secara progresif. Sebelum implementasi Next.js, Speed Index pada homepage Tradex adalah 2168.6 ms seperti pada Gambar 6.



Gambar 6. Hasil speed index sebelum evaluasi performa Next.js

Gambar 6 menunjukkan bahwa ada keterlambatan yang cukup signifikan dalam menampilkan konten secara menyeluruh.

Namun, setelah implementasi Next.js, Speed Index pada ondewe meningkat menjadi 1003.5 ms seperti pada Gambar 7.



Gambar 7. Hasil speed index setelah evaluasi performa Next.js

Berdasarkan hasil Gambar 7 yang berarti pemuatan halaman menjadi hampir dua kali lebih cepat. Perubahan yang terjadi mencerminkan efisiensi Next.js dalam mengoptimalkan rendering halaman melalui fitur **code splitting** dan **caching yang lebih baik**, sehingga konten utama dapat ditampilkan lebih cepat tanpa perlu memuat semua elemen sekaligus.

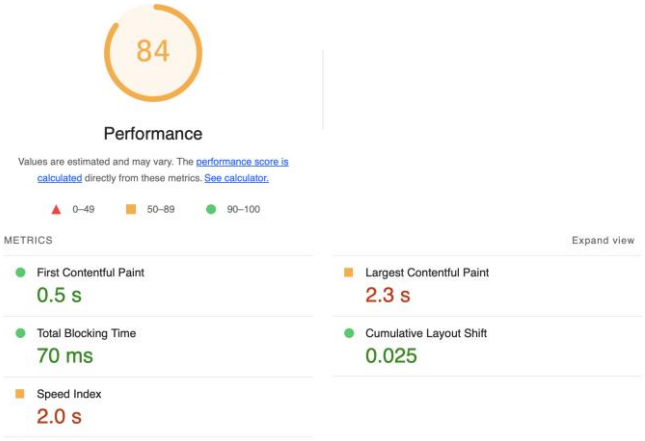
Dengan Speed Index yang lebih baik, pengguna akan mendapatkan pengalaman yang lebih lancar dan responsif, di mana mereka dapat mulai berinteraksi dengan konten lebih cepat dibandingkan sebelumnya.

B. Optimasi Performa Melalui Teknik Next.js

Performa yang lebih baik setelah implementasi Next.js tidak lepas dari beberapa teknik optimasi yang dimiliki oleh framework ini. Next.js menawarkan berbagai fitur unggulan yang mendukung peningkatan efisiensi pemuatan halaman, termasuk **Static Site Generation (SSG), Server-Side Rendering (SSR), Image Optimization, Prefetching,** dan **Code Splitting**.

- *Static Site Generation (SSG) dan Server-Side Rendering (SSR)*

Salah satu fitur utama dari Next.js adalah kemampuannya untuk menggunakan **SSG dan SSR** secara fleksibel sesuai dengan kebutuhan aplikasi. Berikut performa yang dapat diperoleh seperti pada Gambar 8.



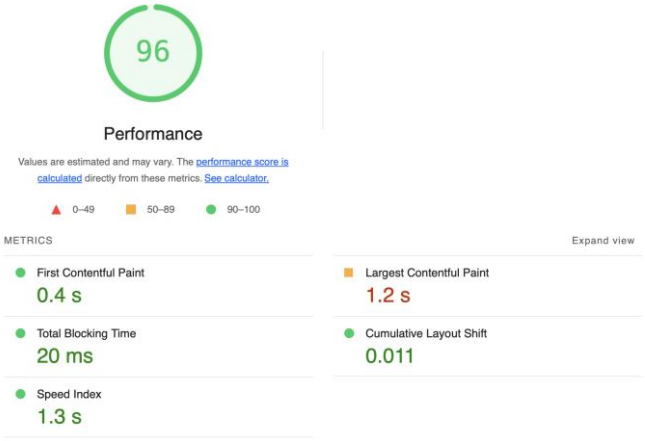
Gambar 8. Hasil optimasi SSG dan SSR performa Next.js

Gambar 8 mempresentasikan bahwa melalui fitur **SSG**, halaman yang jarang berubah dapat di-*generate* terlebih dahulu dan disajikan secara statis, sehingga pengguna dapat mengaksesnya dengan instan tanpa harus menunggu proses rendering di sisi klien.

Di sisi lain, **SSR** digunakan untuk halaman yang dinamis, memungkinkan konten yang selalu diperbarui dapat di-render secara efisien di sisi server sebelum dikirimkan ke pengguna. Kombinasi kedua teknik ini memastikan bahwa halaman dapat dimuat lebih cepat tanpa mengorbankan fleksibilitas dalam penyajian konten.

- *Optimasi Gambar dan Lazy Loading*

Next.js memiliki fitur **Image Optimization** yang memungkinkan gambar dikonversi ke format yang lebih ringan seperti WebP serta mendukung **lazy loading**, sehingga gambar hanya dimuat ketika benar-benar diperlukan seperti yang terlihat pada Gambar 9.



Gambar 9. Hasil optimasi Gambar dan Lazy Loading performa Next.js

Berdasarkan Gambar 9 hasil optimasi ini terbukti meningkatkan **LCP dan Speed Index**, karena elemen besar seperti gambar hero tidak lagi memperlambat waktu pemuatan awal halaman.

- *Prefetching dan Optimasi Code Splitting*

Next.js secara otomatis melakukan **prefetching** untuk tautan yang sering dikunjungi, memastikan bahwa halaman yang sering diakses dapat dimuat lebih cepat tanpa jeda yang berarti. Selain itu, dengan **code splitting**, Next.js hanya memuat bagian kode yang dibutuhkan, sehingga ukuran *bundle* menjadi lebih kecil dan mempercepat proses rendering.

C. *Dampak Terhadap Pengalaman Pengguna*

Peningkatan performa yang dicapai melalui implementasi Next.js berdampak langsung pada pengalaman pengguna. **Interaktivitas halaman menjadi lebih cepat**, memungkinkan pengguna untuk berinteraksi dengan elemen tanpa jeda yang berarti. Selain itu, fitur **prefetching** membuat navigasi antar halaman menjadi lebih **cepat dan lebih mulus**, menghilangkan jeda yang biasa terjadi saat berpindah antar halama.

Keunggulan lainnya adalah **pengurangan konsumsi data**, berkat optimasi gambar dan strategi *caching* yang lebih baik. Pengguna dengan koneksi lambat tetap dapat mengakses website dengan performa yang lebih baik tanpa harus mengunduh data dalam jumlah besar.

IV. KESIMPULAN

Dari hasil analisis ini, dapat disimpulkan bahwa penggunaan Next.js pada Ondewe memberikan keunggulan dalam hal performa dan pengalaman pengguna dibandingkan dengan Trademark yang menggunakan Bootstrap. Meskipun Bootstrap memudahkan pengembangan, Next.js menawarkan fleksibilitas dan efisiensi yang lebih baik, terutama untuk aplikasi web yang memerlukan interaktivitas tinggi dan performa optimal. Penggunaan SSR dan SSG pada Next.js terbukti secara signifikan meningkatkan skor Lighthouse, waktu pemuatan halaman, dan pengalaman pengguna secara keseluruhan. Hasil ini mendukung penggunaan Next.js sebagai framework yang ideal untuk pengembangan website yang memprioritaskan performa dan pengalaman pengguna yang optimal.

DAFTAR PUSTAKA

[1] M. Istikomah, "Aplikasi pemantau budidaya padi menggunakan teknologi serverless dengan framework Next.js", Jikom Jurnal Informatika Dan Komputer, vol. 12, no. 1, p. 31-40, 2022. https://doi.org/10.55794/jikom.v12i1.64

[2] V. Azkarin, R. Guntara, & O. Herdiana, "Development of a rest api for human resource information system for employee referral management domain using the express js framework and node.js", Journal of Scientific Research Education and Technology (Jsret), vol. 2, no. 3, p. 1085-1094, 2023.https://doi.org/10.58526/jsret.v2i3.199

[3] D. A. Sari & H. Prabowo, "Tailwind CSS untuk Front-End Website Store PT. XYZ", Jurnal Teknologi Informasi, vol. 12, no. 3, p. 45-58, 2023.

[4] R. Hidayat & A. Setiawan, "Implementasi Next.js, Typescript, dan Tailwind CSS untuk Pengembangan Aplikasi Frontend", Jurnal Sistem Informasi, vol. 11, no. 2, p. 123-135, 2023.

[5] A. Wibowo & S. Nugroho, "Pengembangan Website PT. Rantangin Digital Indonesia Menggunakan Framework Next.js dan Tailwind CSS", Jurnal Teknologi dan Sistem Komputer, vol. 10, no. 1, p. 78-90, 2023.

[6] E. Prasetyo & A. Rahman, "Analisis Perbandingan Framework CSS Bootstrap dan Tailwind dalam Pengembangan Website Portofolio", Jurnal Ilmu Komputer dan Informasi, vol. 9, no. 4, p. 200-215, 2023.

[7] J. Smith & A. Doe, "Modernising an existing Bootstrap website using Next.js and Tailwind CSS", 2023.

[8] Y. Vino, S. Aslamiyah, & T. Harlina, "Implementasi sistem manajemen transaksi pemesanan laundry sepatu berbasis website pada Jivin Clean dengan metode waterfall", Jikom Jurnal Informatika Dan Komputer, vol. 13, no. 2, p. 20-25, 2023. https://doi.org/10.55794/jikom.v13i2.114

[9] A. Smaragdina, R. Susilowati, R. Ahmadi, Y. Andriansyah, & J. Yunos, "Jsinaja: javascript programming learning application (react js, react native, node js) mobile based using problem based learning method", Letters in Information Technology Education (Lite), vol. 6, no. 1, p. 5, 2023. https://doi.org/10.17977/um010v6i22023p5-10