Ministry of SMEs and Startups

KOSME Korea SMEs and Startups Agency

LIKELION

JOBKOREA

Theory 11
NextJS Inroduction, Routing, Pages

WEEK 04

Nội dung chính

Tổng quan về SSR

❑ **Khái niệm SSR:**

❍ **Server Side Rendering** (**SSR**) là khả năng của ứng dụng để chuyển đổi các tệp HTML trên máy chủ thành một trang HTML hiển thị đầy đủ nội dung cho máy người dùng.

❍ Trình duyệt web gửi yêu cầu thông tin từ máy chủ, máy chủ sẽ trả lời ngay lập tức bằng cách gửi một trang chứa đầy đủ thông tin đến máy khách bao gồm HTML, CSS, JavaScript và các nội dung khác cần thiết để hiển thị trang.

❍ Một số Framework sử dụng SSR phổ biến là **Next.js**, **Nuxt.js** và **Nest.js**

Tổng quan về SSR

❑ **Ưu điểm của SSR:**

❍ **Thời gian tải trang ban đầu nhanh hơn so với CSR**: điều này mang lại trải nghiệm tốt hơn cho người dùng.

❍ **SEO tốt hơn**: do thời gian tải ban đầu nhanh hơn, các bot của công cụ tìm kiếm có thể thu thập thông tin tin và lập chỉ mục các trang tốt hơn.

❍ **Tối ưu cho người dùng có kết nối internet chậm**: Họ có thể xem trước HTML trong khi JavaScript đang xử lý.

Tổng quan về SSR

❑ **Nhược điểm của SSR:**

❍ **Tốn tài nguyên máy chủ**: trình duyệt gửi yêu cầu đến máy chủ cho mỗi trang.

❍ **Tốn thời gian**: việc xử lý trên máy chủ có thể tốn nhiều thời gian cho việc tiếp nhận yêu cầu, xử lý logic và gửi lại phản hồi.

❍ **Có thể có độ trễ khi tương tác ở thời điểm tải trang**: Mặc dù trang HTML hiển thị cho người dùng nhưng họ không thể tương tác với trang đó vì JavaScript được xử lý sau.

❍ **TTFB chậm (Time to First Byte)**: Do cần thời gian để xử lý HTML được kết xuất để phản hồi yêu cầu đầu tiên.

Tổng quan về SSR

❑ So sánh CSR với SSR (Ưu điểm):

Tiêu chí so sánh	Client-Side Rendering (CSR)	Server-Side Rendering (SSR)
Tốn tài nguyên máy chủ	**Ít tốn hơn SSR** vì trình duyệt gửi yêu cầu đến máy chủ cho mỗi trang	**Tốn nhiều hơn CSR** vì mọi thứ render ở máy chủ, máy khách chỉ hiển thị nội dung
Thời gian render	Sau lần tải đầu tiên, từ lần tải thứ 2 trở đi, trang hiển thị rất nhanh	Lần tải đầu tiên có thể bằng hoặc nhanh hơn CSR một chút, từ lần tải thứ 2 trở đi, trang hiển thị chậm hơn CSR
Giảm tải trên máy chủ	JavaScript được thực thi trên máy khách, tạo ít tải hơn cho máy chủ	JavaScript vẫn được thực thi trên máy chủ, tạo nhiều tải hơn cho máy chủ

Tổng quan về SSR

❑ So sánh CSR với SSR (Nhược điểm):

Tiêu chí so sánh	Client-Side Rendering (CSR)	Server-Side Rendering (SSR)
Tốc độ tải trang lần đầu tiên	**Chậm hơn SSR** vì HTML, CSS, JS phải được hiển thị trên DOM ảo trước rồi mới hiển thị lên DOM thật cho người dùng xem, **làm tăng thời gian tải trang lần đầu tiên**	Lần tải đầu tiên có thể bằng hoặc nhanh hơn CSR một chút, từ lần tải thứ 2 trở đi, trang hiển thị chậm hơn CSR
Tối ưu về SEO	Gây khó khăn cho SEO hơn vì Bot của công cụ tìm kiếm phải đợi tải tất cả các tài nguyên của trang được render hoàn chỉnh mới tìm kiếm	Mọi thứ đã được render trên máy chủ, máy khách chỉ hiển thị nội dung, thuận lợi hơn cho Bot của công cụ tìm kiếm làm việc

Tổng quan về SEO

❑ **Khái niệm SEO:**

❍ **Search Enginee Optimization** (**SEO**) là quá trình tăng chất lượng và lưu lượng truy cập Website bằng cách tối ưu tệp trích xuất HTML, cấu trúc Website thân thiện với các công cụ tìm kiếm dữ liệu như Google, Bing, Yahoo,...

❍ **Tối ưu mã nguồn chuẩn SEO** nhằm mục tiêu điều hướng các công cụ tìm kiếm một cách tốt nhất.

Tổng quan về SEO

❑ **Trang Web được đánh giá chuẩn SEO thì bao gồm các tiêu chí:**

❍ Web thân thiện với người dùng, thiết kế đẹp, nội dung tương tác tốt với người dùng.

❍ Web có cấu trúc thân thiện với các công cụ tìm kiếm, giúp việc tìm kiếm dễ dàng thu thập thông tin.

❍ Phần quản trị Website có đầy đủ các cơ chế quản trị thân thiện dễ dàng tùy biến SEO.

Tổng quan về SEO

❑ **Các loại hình SEO gồm có:**

❍ **SEO từ khóa** là loại hình thông dụng nhất hiện nay. Nhà quản trị dựa trên SEO từ khóa tiếng Việt có dấu và không dấu, nhằm mục đích cải thiện thứ hạng của Website trên công cụ tìm kiếm, tăng khả năng tiếp cận với người dùng.

❍ **SEO hình ảnh** là đưa hình ảnh trên Website ưu tiên hiển thị ở những vị trí đầu trên trang tìm kiếm hình ảnh của những công cụ như Google.

❍ **SEO video social** là nâng cao thứ hạng tìm kiếm của Website nhờ các trang mạng xã hội và tương tác người dùng

❍ ...

Lợi ích của SSR với SEO

- Việc render trên máy chủ sẽ tăng tốc thời gian tải trang, điều này không chỉ cải thiện trải nghiệm người dùng mà còn có thể giúp trang web của bạn xếp hạng tốt hơn trong kết quả tìm kiếm của Google.
- SSR tốt hơn cho Seo vì nó loại bỏ gánh nặng hiển thị JavaScript khỏi bot của công cụ tìm kiếm, giải quyết các vấn đề liên quan đến thu thập dữ liệu, tốc độ.

Tổng quan về NextJS

- Để xây dựng một ứng dụng Web hoàn chỉnh với ReactJS từ đầu, có nhiều chi tiết quan trọng bạn cần xem xét:
  - Code phải được đóng gói bằng Webpack và được chuyển đổi bằng trình biên dịch như Babel
  - Bạn có thể muốn render trước một trang cho SEO
  - Bạn muốn thực hiện tối ưu hóa cho trang Web chạy nhanh bằng cách tách mã
  - Bạn cũng có thể vừa muốn render phía máy chủ hoặc render phía máy khách
- **NextJS** sẽ **cung cấp giải pháp cho tất cả các vấn đề nêu trên**

Tổng quan về NextJS

- **NextJS** là một React Framework cung cấp những thành phần cần thiết để giúp bạn tạo ra các ứng dụng Web một cách nhanh chóng.
- **NextJS** có tính năng quan trọng là **Server-Side Rendering** và giúp tạo ra các trang Web tĩnh.

Tính năng của NextJS

- ❑ Pre-rendering, static generation (SSG) và Server-Side Rendering (SSR).
- ❑ Tách mã tự động để tải trang nhanh hơn, chỉ load những gì cần thiết mỗi page
- ❑ Hỗ trợ CSS, SASS hoặc bất kỳ thư việc CSS trong JS
- ❑ Hỗ trợ Refresh page nhanh chóng ở môi trường development
- ❑ Môi trường phát triển và hỗ trợ làm mới nhanh
- ❑ Các Router API xây dựng các API Endpoint với chức năng không có máy chủ
- ❑ Có thể mở rộng dễ dàng

Tính năng của NextJS

- Từ các tính năng trên, NextJS có nhiều lợi ích như sau:
  - Cải thiện quy trình phát triển bằng chi phí và thời gian mang lại lợi ích cho khách hàng.
  - Cải thiện hiệu suất bằng ứng dụng nhanh hơn.
  - Cải thiện SEO bằng các ứng dụng thân thiện với SEO

Cài đặt dự án NextJS

- Khởi tạo dự án Next.js:
  - **Cách 1** (create-next-app -> best practice): npx create-next-app@latest [app-name]
  - **Cách 2** (npm manual): npm i next@latest react@latest react-dom@latest [app-name]
  - Khi cài đặt **cách 1**, lưu ý các lựa chọn sau:

    What is your project named? **my-app**
    Would you like to use TypeScript? No / Yes (Yes: dự án quy mô lớn, No: dự án quy mô nhỏ, vừa)
    Would you like to use ESLint? No / **Yes**
    Would you like to use Tailwind CSS? No / **Yes**
    Would you like your code inside a `src/` directory? No / **Yes**
    Would you like to use App Router? (recommended) No / **Yes**
    Would you like to use Turbopack for `next dev`? **No** / Yes
    Would you like to customize the import alias (`@/*` by default)? **No** / Yes
  - Chạy development server: npm run dev

Điều hướng trong NextJS

- Trong **Next.js**, một trang là một **React Component** được xuất từ một file trong thư mục **pages**.
- Các trang được liên kết với một route dựa trên tên file của chúng, ví dụ:
  - **pages/index.js** được liên kết với route **“/”**
  - **pages/posts/first-post.js** được liên kết với route **“/posts/first-post”**

Điều hướng trong NextJS

- Cách tạo một trang trong Next.js:
  - Tạo fiel JS bên trong thư mục pages
  - **pages/posts/first-post.js** được liên kết với route **"/posts/first-post"**

```
export default function FirstPost() {
    return <h1>First Post</h1>;
}
```

Điều hướng trong NextJS

❑ **Link Component**:

❍ **Link component** cho phép bạn thực hiện điều hướng phía máy khách đến một trang khác trong ứng dụng.

❍ Import **Link** component từ **“next/link”**

❍ **Link component** tương tự như sử dụng thẻ **<a>**, nhưng thay vì **<a href=“..”>**, chúng ta sẽ sử dụng **<Link href=“..”>** và đặt thẻ **<a>** bên trong.

Điều hướng trong NextJS

❑ **Client-Side Navigation**:

❍ **Client-Side Navigation** có nghĩa là quá trình chuyển đổi trang diễn ra bằng JavaScript, nhanh hơn điều hướng mặc định do trình duyệt thực hiện.

❍ **Next.js** thực hiện phân tách mã tự động, vì vậy mỗi trang chỉ tải những gì cần thiết cho trang đó -> đảm bảo rằng trang chủ tải nhanh chóng ngay cả khi bạn có hàng tram trang.

❍ Bạn sẽ tạo **route** dưới dạng file nằm dưới các trang **(/pages)** và sử dụng **Link component**, mà không cần sử dụng các thư viện route.

Static, Dynamic Pages

- **Các loại Route trong Next.js**:
  - **Index routes:** là route sẽ tự động định tuyến các tệp đến thư mục gốc.
  - **Ví dụ:**
    pages/index.js -> /
    pages/blog/index.js -> /blog

Static, Dynamic Pages

❑ **Các loại Route trong Next.js**:

❍ **Nested routes:** là route hỗ trợ các tệp được lồng vào nhau. Nếu bạn tạo cấu trúc thư mục lồng nhau, các tệp sẽ tự động được định tuyến theo cùng một cách.

❍ **Ví dụ:**

pages/blog/first-post.js -> /blog/first-post

pages/dashboard/settings/username.js -> /dashboard/settings/username

Static, Dynamic Pages

❑ **Các loại Route trong Next.js**:

❍ **Dynamic routes:** là các route cho phép chúng ta thêm các tham số tùy chọn vào URL. Next.js sử dụng dấu ngoặc [] trong tên của file, điều này cho phép bạn so khớp các tham số được đặt tên.

❍ **Ví dụ:**

pages/blog/[slug].js -> /blog/:slug (/blog/hello-world)
pages/[username]/settings.js -> /:username/settings (/foo/settings)
pages/post/[...all].js -> /post/* (/post/2025/id/title)

Static, Dynamic Pages

❑ **Cách tạo Dynamic Route trong Next.js**:

❍ Trong Next.js, bạn có thể thêm dấu ngoặc vào một trang ([param]) để tạo dynamic route.

❍ Cùng xem page sau: **pages/post/[pid].js**

```
import { useRouter } from ‘next/router’;
const Post = () => {
    const router = useRouter();
    const { pid } = router.query;
    return <p>Post: {pid}</p>
}
```

Static, Dynamic Pages

❑ **Cách tạo Dynamic Route trong Next.js**:

❍ Qua ví dụ trên, lưu ý:

➢ Bất kỳ route nào như **/post/1, /post/abc**, v.v… sẽ được khớp với các **pages/post/[pid].js**

➢ Ví dụ: route **/post/abc** sẽ có đối tượng truy vấn như sau: **{“pid”: “abc”}**

➢ Tương tự, route **/post/abc?foo=bar** sẽ có đối tượng truy vấn như sau: **{“foo”: “bar”, “pid”: “abc”}**

➢ Tuy nhiên, các **route parameter** sẽ ghi đè các tham số truy vấn có cùng tên. Ví dụ: route **/post/abc?pid=123** sẽ có đối tượng truy vấn: **{“pid”: “abc”}**

Static, Dynamic Pages

❑ **Cách tạo Dynamic Route trong Next.js**:

❍ Qua ví dụ trên, lưu ý:

➢ Tuy nhiên, các **route parameter** sẽ ghi đè các tham số truy vấn có cùng tên. Ví dụ: route **/post/abc?pid=123** sẽ có đối tượng truy vấn: **{“pid”: “abc”}**

➢ Nhiều dynamic route hoạt động cùng một cách. Các trang **pages/post/[pid]/[comment].js** sẽ khớp với route **/post/abc/a-comment** và đối tượng truy vấn của route này là: **{“pid”: “abc”, “comment”: “a-comment”}**

Static, Dynamic Pages

❑ **Sử dụng Client-Side Navigation với Dynamic Route, next/link**:

```
import Link from 'next/link'

function Home() {
    return (
        <ul>...</ul>
    )
}

export default Home
```

Static, Dynamic Pages

❑ **Sử dụng Client-Side Navigation với Dynamic Route, next/link**:

```
<ul>
    <li>
        <Link href="/post/abc"><a>Go to pages/post/[pid].js</a></Link>
    </li>
    <li>
        <Link href="/post/abc?foo=bar"><a>Also goes to pages/post/[pid].js</a></Link>
    </li>
    <li>
        <Link href="/post/abc/a-comment"><a>Go to pages/post/[pid]/[comment].js</a></Link>
    </li>
</ul>
```

Static, Dynamic Pages

❑ **Sử dụng Link với dynamic path** – với chuỗi nội suy (**string interpolation**):

```
function Posts({ posts }) {
    return (
      <ul>
        {posts.map((post) => (
          <li key={post.id}>
            <Link href={`/blog/${encodeURIComponent(post.slug)}`}>
              <a>{post.title}</a>
            </Link>
          </li>
        ))}
      </ul>
    )
}
```

Static, Dynamic Pages

❑ **Sử dụng Link với dynamic path** - sử dụng **Object URL**:

```
function Posts({ posts }) {
    return (
      <ul>
        {posts.map((post) => (
          <li key={post.id}>
            <Link href={{ pathname: ‘blog/[slug]’, query: { slug: post.slug }, }}>
              <a>{post.title}</a>
            </Link>
          </li>
        ))}
      </ul>
    )
}
```

Tóm tắt bài học

- ❑ Tổng quan về SSR
- ❑ Tổng quan về SEO
- ❑ Lợi ích của SSR với SEO
- ❑ Tổng quan về NextJS
- ❑ Tính năng của NextJS
- ❑ Cài đặt dự án NextJS
- ❑ Điều hướng trong NextJS
- ❑ Static, Dynamic Pages