Getting Started

Terms

Client components
Client-side Rendering (CSR)
Dynamic rendering
Node.js runtime
Server components
Server-side Rendering (SSR)
Static rendering
Static Site Generation (SSG)

Summary

- Next.js is a framework for building fast, and search-engine friendly applications.
- It includes a compiler for transforming and minifying JavaScript code, a Command-line Interface (CLI) for building and starting our application, and a Node.js runtime for running backend code. This enables full-stack development.
- With Next.js, we can render our components on the server and return their content to the client. This technique is called *Server-side Rendering* (SSR) and makes our applications search-engine friendly.
- To further improve performance, we can pre-render pages and components that have static data during the build and serve them whenever needed. This technique is called *Static Site Generation* (SSG).
- The new app router in Next.js 13 makes it incredibly easy to create routes. We can define route segments by creating directories. To make a route public, we add a page file (page.js, page.jsx, or page.tsx) in the corresponding directory.

- Next.js provides the Link component to enable *client-side navigation*. This means as the user navigates between pages, the new content is loaded quickly and smoothly without the entire page being reloaded.

- Next.js 13 supports *client and server components* introduced in React 18. Client components are rendered on the client within a web browser. This technique is called *Client-side Rendering (CSR)* and it's how traditional React apps work. Server components are rendered on the server within a Node.js runtime. This technique is called *Server-side Rendering (SSR)*.

- Server components lead to reduced bundle sizes, better performance, increased search engine visibility, and enhanced security. But they cannot handle browser events, access browser APIs, or use the state or effect hooks. These functionalities are only available in client components. So we should use them whenever possible unless we need interactivity.

- All the components and pages in the /app directory are server components by default. To make a component a client component, we add the **'use client'** directive on top of the component file.

- Server components are great for fetching data because they don't require extra server trips, making our application faster and more search-engine friendly.

- Next.js enhances the fetch() function by adding automatic caching. This boosts performance and reduces the need to retrieve the same piece of data twice.

- In Next.js, components can be rendered at build time (called *Static Rendering*) or at request time (called *Dynamic Rendering*). If we have pages or components with static data, we can pre-render them during build time to improve our application's performance.

Key Commands

```
# Creating a new project
npx create-next-app

# Running the dev server
npm run dev

# Building the application
npm run build

# Starting the application in production mode
npm start
```


Styling

Terms

CSS modules
Daisy UI
Global styles
PostCSS
Tailwind

Summary

- In Next.js projects, we define global styles in **/app/global.css**. Reserve this file for global styles that need to be applied across multiple pages and components. Avoid adding excessive styles to this file, as it can quickly grow out of hand and become difficult to maintain.

- In traditional CSS, if we define the same class in two different files, one will overwrite the other depending on the order in which these files are imported. *CSS modules* help us prevent this problem. A CSS module is a CSS file that is scoped to a page or component.

- During the build process, Next.js uses a tool called *PostCSS* to transform our CSS class names and generate unique class names. This prevents clashes between different CSS classes across the application.

- *Tailwind* is a widely-used CSS framework for styling application. It offers a comprehensive set of small, reusable utility classes. We can combine these classes to create beautiful user interfaces.

- DaisyUI is a component library built on top of Tailwind. It provides a collection of pre-designed and reusable components such as accordion, badge, card, etc.

