Thursday 9-12 Next.js Assignment 04

Presented by Aleeza | GIAIC Student

What is Next.js ?

Next.js is a React framework for building full-stack web applications. It enhances React with features like automatic routing, server-side rendering, and optimization, making development easier. With Next.js, the focus shifts to creating applications without the hassle of complex configurations. It's ideal for solo developers and teams aiming to build fast, interactive applications.

What is the page.tsx file, and what is the layout.tsx file ?

- page.tsx: This file defines a specific page of the application. With each page corresponds to a route in the app. A React component is exported from this file to render the content for that route.
- Layout.tsx: This file is used to define a common layout for multiple pages. It wraps around the content from page.tsx, allowing for shared elements like headers, footers, or navigation across different pages.

What is the Link tag ?

The Link tag is a component provided by Next.js for navigating between different pages in a web application. It allows for client-side routing, which means users can move from one page to another without experiencing full page reloads.

Why do we use Link tag ?

Enhanced User Experience	The Link tag facilitates smooth transitions between pages, creating a responsive and fluid application.
SEO Benefits	It maintains a proper structure for search engines, enhancing indexing and improving visibility in search results.
Error Handling	It allows for built-in error handling for navigational links, providing fallback options if a page is not found or if there's a navigation issue.
Automatic Pre-fetching	Next.js automatically loads linked pages as they come into view, which helps to speed up loading times and makes navigating the site feel smooth and effortless.

What is the purpose of Link tag ?

The Link tag boosts SEO by allowing seamless navigation between pages without reloading the entire site, which enhances user experience (UX). It provides quicker transitions than traditional anchor tags in HTML, which cause a full page reload each time a navigation link is clicked. This leads to a smoother and more efficient browsing experience for users.

app/page.tsx — TypeScript

```tsx
import Link from 'next/link'

export default function Page() {
  return <Link href="/dashboard">Dashboard</Link>
}
```

How can we create nested pages in Next.js ?

Creating nested pages in Next.js is a great way to keep the application organized.

1.Create a Folder: Inside your app directory, create a new folder for your main route. For example, to create /about, create a folder named about.

2.Add a Page File: Inside the about folder, create a file named page.tsx. This file will handle the content for the /about route.

3.Create Nested Routes: To add a nested route, like /about/team, create another folder named team inside the about folder. Then, add a page.tsx file in the team folder to define the content for that route.

What are components, and why do we use them ?

Components in Next.js are essential parts of a web application. They allow for the creation and management of different sections of a site—like buttons, headers, and more—that can be reused throughout the project. This approach makes building and maintaining a website easier and more efficient!

Reusability: Create once, use anywhere! Design a button and deploy it on multiple pages without rewriting.
Organization: Keep your code neat. Break complex applications into smaller, manageable pieces.
Easy Maintenance: Change styles or fix bugs in one place—effortlessly update across the board!

How can we apply CSS in Next.js ?

Next.js offers two popular methods for styling: **CSS Modules** and **Tailwind CSS**.

CSS Modules:

CSS Modules provide a way to write CSS that is scoped locally to the component, preventing styles from affecting other parts of the application. This helps maintain clean and organized code.

app/dashboard/styles.module.css

```
1  .dashboard {
2    padding: 24px;
3  }
```

What is Tailwind CSS ?

TAILWIND CSS:

Tailwind CSS is a utility-first CSS framework that streamlines styling in web applications. It provides a collection of pre-defined classes, allowing for rapid and responsive design without writing custom CSS.

Importing Styles

Add the Tailwind CSS directives that Tailwind will use to inject its generated styles to a Global Stylesheet in your application, for example:

app/globals.css

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

app/page.tsx — TypeScript

```
export default function Page() {
  return <h1 className="text-3xl font-bold underline">Hello, Next.js!</h1>
}
```

tailwindcss

What are the differences between Tailwind CSS and standard CSS ?

Feature	Tailwind CSS	Standard CSS
Styling Approach	Uses utility-first classes directly in HTML (e.g., bg-blue-500).	Styles defined in separate CSS files with selectors.
Customization	Highly customizable via a config file for easy tweaks.	Involves creating or overriding rules, which can get complex.
Responsiveness	Built-in responsive utilities simplify styling for various screen sizes (e.g., md:bg-red-500).	Relies on media queries, often leading to more code.
Learning Curve	Steeper initially due to many utility classes to remember.	More familiar for those used to traditional methods.
File Size	Uses PurgeCSS to remove unused styles, keeping the bundle lightweight.	Can become bulky if not managed properly.

Thank you for taking the time to read!