

# **PRESENTATION ON NEXT JS**

**ASSIGNMENT 4  
ASSIGN BY SIR HAMZA ALVI  
THURSDAY 9 TO 12 SLOT**

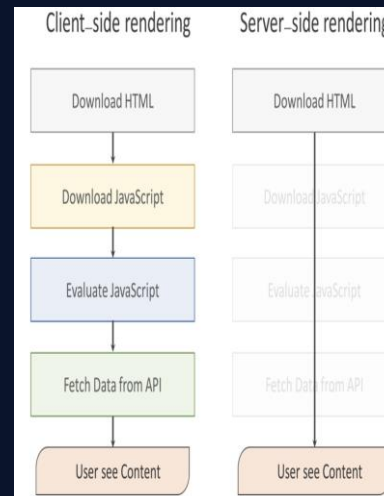
## WHAT IS **NEXT JS** ?

Next.js is a **framework built on React** that helps create web applications quickly with built-in tools for routing, server-side rendering, and static site generation.

The logo for Next.js, featuring the word "NEXT" in a large, black, sans-serif font, followed by ".JS" in a smaller, black, sans-serif font. A thin black diagonal line crosses through the "X" and extends upwards and to the right.

**OOPS, what is CSR (Client-Side Rendering) or SSR (Server-Side Rendering) ?  
Let me define it.**

✓ **CSR**  
**(Client-Side-Rendering)**  
📱 Web pages are rendered in the browser using JavaScript. Offers interactivity but can lead to longer load times and SEO challenges.



✓ **SSR**  
**(Server-Side-Rendering)**  
🌐 Web pages are rendered on the server, delivering complete HTML to the client. Better for SEO and initial load times.



## Next.js is used for:

### ✓ Better SEO

Definition: Improves indexing with server-side rendering.

Example: Blog posts indexed before loading.

### ✓ API Routes Definition:

Easy creation of API endpoints.

Example: `api/hello.js` → `/api/hello`.

### ✓ Easy Routing

Definition: Simple routing via file structure.

Example: `about.js` → `/about`.

### ✓ Enhanced Developer Experience

Definition: Quick updates and Typescript support.

Example: Live browser refresh as you code.

### ✓ Automatic Code Splitting

Definition: Loads only needed code for each page.

Example: Active page code only.



# LETS COME TO THE TOPICS WHICH IS VERY IMPORTANT

*Here is the question of Assignment.....*

*lets solve it .....*

- What is the **page.tsx** file, and what is the **layout.tsx** file?
- What is the **Link** tag, why do we use this tag, and what is its purpose?
- How can we create **nested pages** in Next.js?
- What are **components**, and why do we use them?
- How can we apply **CSS** in Next.js?
- What is **Tailwind CSS**, and what are the differences between **Tailwind CSS** and **standard CSS**?
- Create a single presentation that includes clear answers to these questions.....

# What is the `page.tsx` file?

- `page.tsx` is a special Next.js file that exports a React component and is a necessary for a route to be accessible. It acts as the main page, rendering the primary content, and should start with a parent element for proper structure.



# What is the `layout.tsx` file?

A `Layout.tsx` file defines UI elements that are shared across multiple pages in an application and wraps the entire app, ensuring consistency.

To display common components like a header, navbar, sidebar, or footer on every page, import them into `Layout.tsx`, avoiding code repetition on individual pages.

```
1  import Navbar from './navbar'
2  import Footer from './footer'
3
4  export default function Layout({ children }) {
5    return (
6      <>
7        <Navbar />
8        <main>{children}</main>
9        <Footer />
10     </>
11   )
12 }
```



# What is the **Link** tag in next js?

The **Link** tag in Next.js is used for client-side navigation between pages.

We use the **Link** tag to improve the user experience by enabling faster page transitions without reloading the entire page. Its purpose is to create links that leverage Next.js routing capabilities, making navigation seamless and efficient.

Link component : `<Link href="/page">New Page</Link>`





# Here's a real-world comparison

Imagine a **library** with different sections. If you had to exit and re-enter every time you wanted to switch sections, it would be frustrating.

Now, imagine being able to walk between sections without leaving the library. That's how the **Link** tag in Next.js works. It allows you to **navigate between pages** on a website without reloading, making your experience faster and smoother.



# HOW I CREATE A **NESTED PAGES** IN NEXTJS?

**Nested Pages** in Next.js refer to pages organized in a hierarchical structure within the **app** directory, where each folder represents a route. For example, a folder **about/team** would create a nested route accessible via **/about/team**. This organization helps manage complex applications by grouping related pages together, allowing for better structure and routing.



# HERE'S A REAL-WORLD COMPARISON

**Nested Pages** in Next.js can be compared to a university campus:

The main building represents the university's homepage (accessible at [/university](#)). Inside, the Arts department represents the [/university/arts](#) route. Within the Arts department, the Music section details music programs (accessible at [/university/arts/music](#)).

This structure helps organize information clearly, similar to how Next.js allows for nested pages, creating a logical hierarchy for easier navigation.



# HOW I CREATE A **COMPONENTS** IN NEXT JS?

✓ components

Footer.tsx

Header.tsx

Hero.tsx

- ✓ **Components** in Next.js are reusable snippets of code written in TSX that combine HTML, JSX, and JavaScript logic for UI elements. Examples include:

**Header** : Navigation across pages.

**Hero** : A prominent banner for key content.

**Footer** : Consistent information at the bottom.

Storing components in separate files and capitalizing their names enhances organization and improves load times and rendering efficiency.

**CSS Modules:** Use component-specific CSS by creating `.module.css` files. Import these in the relevant components for localized styles without global conflicts.

**Global CSS:** For site-wide styles, create a `global.css` file and import it into the main layout (e.g., `_app.js`). This applies styles across all pages without needing repeated imports.

**Tailwind CSS:** Utilize utility-first classes directly in your HTML for rapid, inline styling without custom CSS files.


**Styled Components:** Write CSS within JavaScript files for dynamic styling using libraries like `Styled Components`



# WHAT IS TAILWIND CSS ?

Tailwind CSS is a utility-first framework that enables developers to create custom designs directly in HTML using utility classes. This method offers flexibility and precise control over styling, making it ideal for modern applications built with React and Next.js. By streamlining the styling process, Tailwind promotes faster development of responsive, high-performance interfaces.

```
src > app > page.tsx > Home
1  export default function Home() {
2    return (
3      <div>
4        <h1 className="text-center bg-[#ffff] font-mono font-semibold">NIRMA QURESHI</h1>
5      </div>
6    );
7  }
```



# Differences between Tailwind CSS and Standard CSS

## TAILWIND CSS

- Utility classes applied in markup.
- Reusable utility classes reduce custom styles.
- Faster with inline utility classes

## STANDARD CSS

- Custom styles in separate files.
- Requires unique styles for components.
- Slower due to context switching



**Presentation created by NIRMA QURESHI**  
**THANKS FOR WATCHING**