NEXT JS Essentials - App Router

Understanding File based Routing

→ NextJS uses files and folders to define routes

- Page.js is a reserved filename in Next
  - ↳ this simply tells next that it should render a page
  - ↳ which is simply a react component
  - ↳ (here) its called as server component
  - ↳ next js ensures this function is executed on the server so if you console log something you wont see it in your browser instead we can see it on our terminal

So its a regular react component but treated in a special way by NEXT js

Adding another route in file system

So in nextjs new paths can be created by adding new folders in app directory

eg:

So if we want www.localhost/about route we add

about folder in app folder

↳ additionally add page.js in about folder

inside page.js

```
export default function AboutPage(){
    return (
        <main>
            <p> About </p>
        </main>
    )
}
```

JSX code we want to render

Navigating between pages

now if we use traditional anchor tags our page will be refreshed
now
Next allows us to actually stay in single page application and update the UI with the help of client side Javascript code

what we can use is Link provided by Next

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

   <Link href='/about'> About Us </Link>
```

Working with page and layouts

page.js → defines content of the page
layout.js → defines shell around one or more pages

NEXT requires atleast one root layout.js file

c/

reserved keyword

```
export const metadata = {
    title: '...'
    discription: '...'
}
```

this is where we add all contents of <head> tag

```
export default function RootLayout({children}) {
    return (
        <html lang: 'en'>
            <body> {children} </body>
        </html>
    )
```

Reserved filenames, custom components and how to organise a nextjs project

→ `icon.png` : this will be used as favicon by Nextjs

→ you can still create normal component just like React and can store this file anywhere

```
// export default function Header () {

    return ...

}
```

→ `jsconfig.json`

```
{ "compilerOption": {
    "paths" : {
        "@/*" : [ "./*" ]
    }
  }
}
```

So now due to this configuration your imports will look like this

c/ import Header from '@/components/header'

@ signifies root directory of the project

→ route.js

Allows us to create API route i.e. page which returns data

→ not-found.js

Fallback page for "not found" errors

→ error.js

Fallback page for other errors

→ loading.js

Fallback page shown whilst sibling or nested pages are fetching data

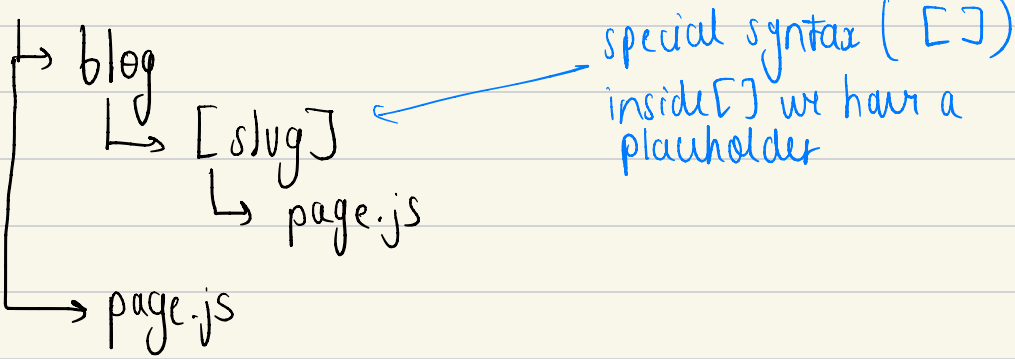
Configuring Dynamic Routes and route parameters

eg: Now suppose we have a blog page and nested inside we have various posts

so for rendering different posts we need dynamic routes

↳ A route which is defined only once but which is then capable of rendering different pages for different posts

so this is how folder structure looks for it



1) Page.js - outer

in outer js file we define links

```
<p>
    <Link href='/blog/post-1'>
        Post 1
    </Link>
</p>

<p>
    <Link href='/blog/post-2'>
        Post 2
    </Link>
</p>
```

value

2) page.js inside [slug]

this simply tells next js that we want to have some path segments but we dont know the exact value yet

this placeholder gives the exact value when the route is loaded

c/

```
export default function BlogPost({params}) {
    return (
        <main>
            <p> {params.slug} </p>
        </main>
    )
}
```

this is the prop set by next js which we can use

so this dynamic placeholder is the key and the value is dynamic route value of url

⟹ value : post-1, post-2