# \*\* in next js we have two type of Router \*\*

1)app Router

2)Page Router

* app Router is used for server-side rendering and page Router is used for client-side rendering
* we can use both type of Router in next js but we have to use one type of Router in whole application
* we can use app Router by using getStaticProps and getServerSideProps method in pages
* But in latest version we use app Router

# \*\* In react 18 we have two type of component

1)Server component

2)Client component

1)Server component: -

* in Server component we can access data from database and also, we can read file.
* we can use server component to fetch data from server and also, we can use it to send data to server.
* Server component cannot interact with user.

2)Client component: -

* in Client component we can interact with user.
* we can use client component to render UI.
* Client component cannot access data from database and also, we cannot read file in client component.
* To create Client component, we use "use client" to the top of the file.

# \*\* next js has file base routing mechanism \*\*

## \*\* Routing convention \*\*

* All Routing place inside app folder.
* Every file that is coition routing name has to be page.tsx or page.jsx (if we are using JavaScript)

# \*\* next js provide catch all segment \*\*

* in catch all segment we can rename file to [...slug] in that we create page.tsx file in that we have to catch

{

params,

}: {

params: {

slug: string [];

};

}

# \*\* next js also provide privet folder \*\*

* in privet folder we cannot navigate or to make routing
* to create privet folder, we have to name that start with \_<folder name>

# \*\* next js provide Route group \*\*

* in Route group manse is that we have to create folder to group some action.
* for example, we have login, sign up and forget password.
* we can create folder name auth and inside that we can create login, sign up and forget password page.
* to access this page, we have to go to auth folder and that that file name.
* so next js provide Route group to create Route group we have to wrap folder name in(auth) so we don't have to write auth in
* order so access login. Sign up or forget password we don't have to write auth.

# \*\* next js provide routing group layout \*\*

* that means I have one layout in app folder that layout will be display in all file like layout.jsx in react.
* to create routing group layout, we have to create layout.tsx in folder that we want to display in.
* for example, we have auth folder and we want to display layout in all file in auth folder
* so, we have to create layout.tsx in auth folder and that layout will be display in all file in auth folder.
* so now if we access auth folder first control will go to layout.tsx in that first layout will be display and then child we load.
* child will come from page.tsx.
* if we only have layout.tsx and not page.tsx so in that case first layout.tsx will display and if search for page.tsx and if there is no page.tsx then they only display layout.

# \*\* next js provide Metadata \*\*

* next js is seo friendly because next js provide Metadata.
* for example, next js provide metadata on root layout in that we write title and description.
* we can also write metadata in page.tsx file.
* if we change page and if in that page.tsx or layout.tsx does not have metadata function (title and description) so in that case there will be root layout metadata
* if there is a metadata so in that case first priority will be that file metadata's and if there no metadata there will go to root layout and their metadata will be display.
* we can write metadata in file using

import {Metadata} from "next";

export const generateMetadata = ({params}: Props): Metadata=> {

return {

title: "Home Page",

description: "This is home page",

};

}

* metadata can be applying in object.
* for example:

import {Metadata} from "next";

export const metadata: Metadata = {

title: {

absolute:"",

default:"",

template:"%s | Next.js"

}

description: "This is home page",

}

* default: is child layout do not provide title in that case default will be apply as a title.
* template: if child provide title, and we have template in that case title replace with "%s", and coenact with | Next.js.
* absolute: and if child provide default in that case they will ignore template and display only default value.

# \*\* next js provide parallel routes \*\*

* Parallel routes are same as component in react but in some case, they are batter.
* for example, we have complex layout like left side we have two div and right side we have one.
* to create that in react we create three component and that three components import in page.tsx and give style.
* but in next js we can create component that folder name starts with "@" for example if have notification and that folder name will be @notification and that folder will have page.tsx.
* so, in page.tsx we can import @notification and that will be display in that page.
* we don't have to import component what start with @ they pass as Props in parent page.tsx.
* for example, if we have @notification and that have page.tsx and that page.tsx have notification: React.ReactNode

# \*\* Why we use parallel routes? \*\*

* Independent UI Sections: Parallel routes allow you to have different parts of the UI that dose not rely on the same route or path. For example, you might have a main content area, a sidebar, and a modal. Each of these can be controlled independently with parallel routes.
* Enhanced User Experience: With parallel routes, you can provide a richer and more interactive user experience. For instance, if you have a complex dashboard with various sections (like a main view, notifications, and user profile details), you can update each section independently without reloading the entire page or affecting other sections.
* Modular Design: They help in creating a modular and maintainable codebase. Each parallel route can manage its own state and rendering logic, which helps in organizing the code and separating concerns.
* in simple way if there is error in one component we have dot get crash full website next.js provide error.tsx in that we can make over error bounder that will prevent form crashing full website and give that error.tsx.
* and similarly, we can give loading.tsx that will automatic give loading state individual that we can show there will show in that page.tsx.

# \*\* Next.js provide Intercepting Routes \*\*

* that means we can create some UI that will display first and if user reload that page then user can see that folder page.tsx
* example if user in on http://localhost:3000/ and user click on login and link will be http://localhost:3000/login
* but in case of Intercepting Routes we can create with (.)<folder name> in case of login Intercepting Routes folder name will be (.)login and if we create Intercepting Routes we can display that Intercepting folder
* to creating Intercepting Routes purees is that is we have 6 images can user lick one of that so I want to open that image in model and if user share that URL to other and I don't want to show in modal and show in individual image or different style in that case we use Intercepting Routes.
* there are some rules that we have to flowed like
* we want to create login Intercepting Routes we have to create in same directory as login and name same as login but start with (.)
* if we want to create parents Intercepting Routes in child we can create like if we have auth>login and we also have auth > sign up and we want create Intercepting Routes of sign up in login folder we can create with (..)signup folder name
* in short if we want to create Intercepting Routes of one level above we have to give name with (..).
* if we want to create Intercepting Routes of two level above we have to give name with (..)(..).
* if we want to create Intercepting Routes of root level above we have to give name with (...).
* Like in below we have example: -