React JS

Handling Navigation in React

Agenda

01 Routing in Single Page Apps

02 React Router Dom

03 React Router Advantages

04 Components of react-router-dom

05 Navigation using Links

06 Handling 404 Route

07 URL Parameters

08 Nested Routes

Routing in Single Page Apps

Routing in Single Page Apps

A Single Page Application, is an application which uses a single html page for initial loading and after that all further interaction is managed by JavaScript

Routing in Single Page Apps

These Interactions include handling user events, data changes, moving between pages for navigation etc.

Routing in Single Page Apps

In React in order to handle navigation between pages which is caused by change in current url, is done by using routing libraries

Routing in Single Page Apps

Although there are many routing libraries available for react react-router is the most popular library

React Router Dom

React Router Dom

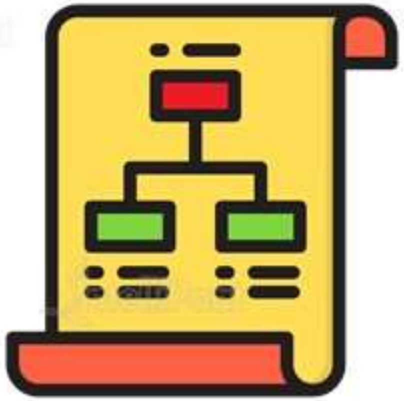
As React can be used to build applications for various platforms such as web and mobile we can also use react router for these platforms

React Router Dom

But routing on a native mobile application is quite different from routing in a Single Page Web Application

React Router Dom

Because of this reason there are multiple react router libraries corresponding to multiple platforms



React Router Dom

For the web we use react router dom as it is the package developed to work well with the web platform

React Router Advantages

React Router Advantages

React Router is quite popular because it is one of the

Easy

Declarative

Fast

Transitions

React router is incredibly easy to learn as it has a very intuitive API design so we can easily get started with it

React Router Advantages

React Router is quite popular because it is one of the

Easy

Declarative

Fast

Transitions

Just like ReactJS, React Router is also declarative so it is easier for us to describe how we want our routes to be structured

React Router Advantages

React Router is quite popular because it is one of the

Easy

Declarative

Fast

Transitions

React Router is very performant and switching between routes in a react app once they are loaded is blazingly fast

React Router Advantages

React Router is quite popular because it is one of the

Easy

Declarative

Fast

Transitions

In React Router animations and transitions can be easily implemented when switching between different components

Components of react-router-dom

Components of react-router-dom

Handling routing declaratively can be a bit difficult to solve this react-router-dom has built react components to be used in react code

Components of react-router-dom

There are several components in react-router-dom but there are three kinds of components that are most important to understand

Routers

Route Matchers

Navigation

Every React Router application should have a router component as the top level component

Components of react-router-dom

There are several components in react-router-dom but there are three kinds of components that are most important to understand

Routers

Route Matchers

Navigation

For web projects, react-router-dom provides <BrowserRouter> and <HashRouter> routers

Components of react-router-dom

There are several components in react-router-dom but there are three kinds of components that are most important to understand

Routers

Route Matchers

Navigation

A <BrowserRouter> uses regular URL paths, but needs the server to be configured correctly and also it only works with browsers supporting HTML5

Components of react-router-dom

There are several components in react-router-dom but there are three kinds of components that are most important to understand

Routers

Route Matchers

Navigation

A <HashRouter> stores the current location in the hash portion of the URL, so the URL looks something like http://example.com/#/your/page

Components of react-router-dom

There are several components in react-router-dom but there are three kinds of components that are most important to understand

Routers

Route Matchers

Navigation

There are two route matching components: Switch and Route, in which Route should be child component of Switch

Components of react-router-dom

There are several components in react-router-dom but there are three kinds of components that are most important to understand

Routers

Route Matchers

Navigation

When a <Switch> is rendered, it searches through its children <Route> elements to find one whose path matches the current URL

Components of react-router-dom

There are several components in react-router-dom but there are three kinds of components that are most important to understand

Routers

Route Matchers

Navigation

When it finds one, it renders that <Route> and ignores all others below it

Components of react-router-dom

There are several components in react-router-dom but there are three kinds of components that are most important to understand

Routers

Route Matchers

Navigation

Because of this we have to put <Route>s with more specific paths before less-specific ones.

Components of react-router-dom

There are several components in react-router-dom but there are three kinds of components that are most important to understand

Routers

Route Matchers

Navigation

For Navigation we have few specific components namely <Link>, <NavLink> and <Redirect>

Components of react-router-dom

There are several components in react-router-dom but there are three kinds of components that are most important to understand

Routers

Route Matchers

Navigation

The <Link> component to create links in your application. Wherever you render a <Link>, an anchor (<a>) will be rendered in your HTML document.

Components of react-router-dom

There are several components in react-router-dom but there are three kinds of components that are most important to understand

Routers

Route Matchers

Navigation

A <NavLink> is a special type of <Link> that can add a class or styles to itself when its to prop matches the current location

Components of react-router-dom

There are several components in react-router-dom but there are three kinds of components that are most important to understand

Routers

Route Matchers

Navigation

Any time that you want to force navigation, you can render a <Redirect>. When a <Redirect> renders, it will navigate using its to prop

Hands On: Using React Router Dom

Navigation using Links

Navigation using Links

In React Router Links are used to navigate inside a single page application

Navigation using Links

React Router support two types of link components: <Link> and <NavLink>

Navigation using Links

A Link Provides declarative, accessible navigation around your application.

Navigation using Links

<NavLink> is a special version of the <Link> that will add styling attributes to the rendered element when it matches the current URL.

Handling 404 Route

Handling 404 Route

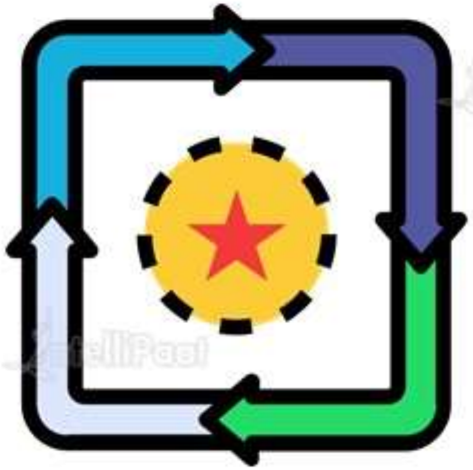
In a production build of a single page application we have to handle all possible errors we might come across

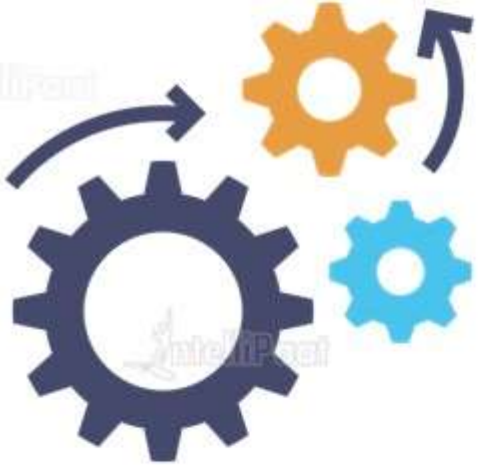
Handling 404 Route

One of these errors occurs if we navigate to a page that does not exists

Handling 404 Route

To do this we use Switch and Route Component and make use of the fact that Switch renders the first Route that matches - even if there’s more than one match otherwise it looks through all routes looking for a match





Handling 404 Route

So we have to render Route Components inside a Switch Component with last route rendering a page saying page not found and also not having any path prop passed so it matches every route

Hands On: Handling 404 Routes

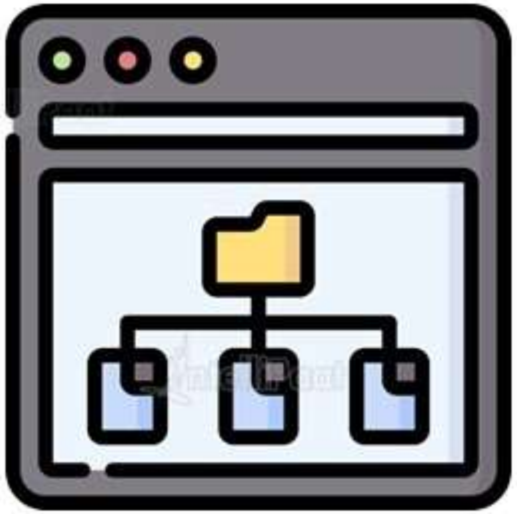
URL Parameters

URL Parameters

A URL Parameter is placeholder in our URL that needs to be replaced by value

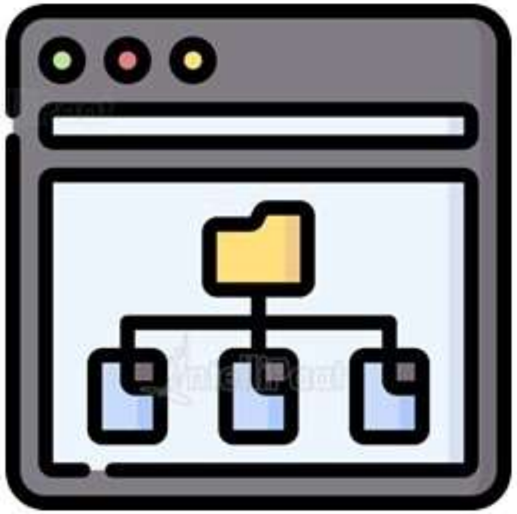
URL Parameters

A URL Parameter in a URL in react router begins with a colon and an identifier representing that parameter e.g. /user/:id



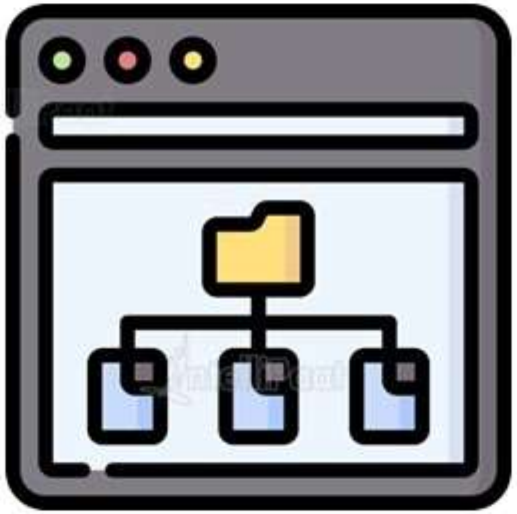
URL Parameters

A URL is not restricted to one parameter as it can have multiple parameters just make sure that they are all distinct and necessary



URL Parameters

To access url parameters value we use the useParams hook which returns a JavaScript object with key as parameter name and value is parameter value

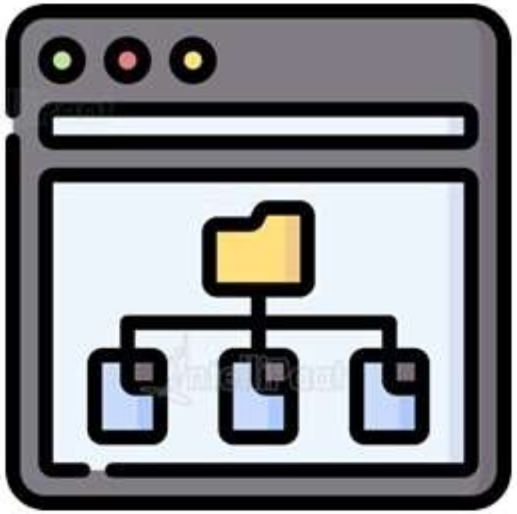


Hands On: Capturing URL Parameters

Nested Routes

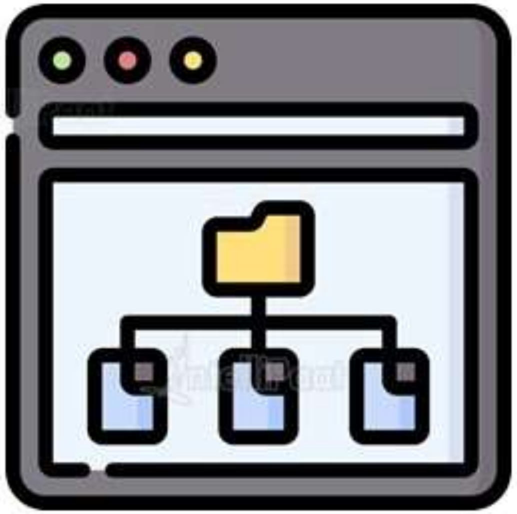
Nested Routes

In a single page react application, it often happens that we need to deal with routes that are nested into other routes



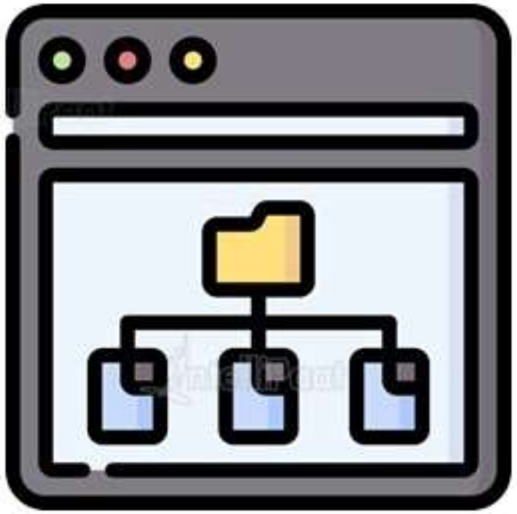
Nested Routes

A Nested Route basically is just a route that has that share common prefix from a parent route for example, /user/1 is parent route of /user/1/posts/1



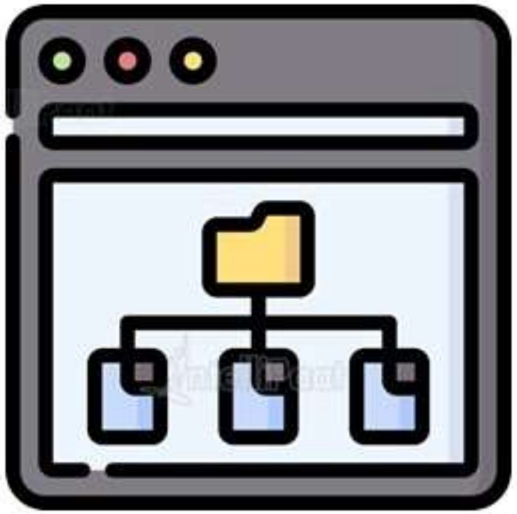
Nested Routes

To render nested routes we need to get access to the current route and add a few parameters at their end



Nested Routes

To get access to current URL we need to use useRouteMatch hook which returns an Object with path and current url which can be used for nesting



Hands On: Nested Routes

IntelliPaat

India: +91-7847955955

US: 1-800-216-8930 (TOLL FREE)

support@intellipaat.com

24/7 Chat with Our Course Advisor