

Fetching News category wise in NewsMonkey React App | Complete React Course in Hindi #31

In the last video, we have passed the categories and Proptypes in the NewsMonkey application. In today's video, we would be setting up a react-router and would be fetching the News category wise in our NewsMonkey application.

Install React Router package

Let's see how to set up react-router in our code:

Procedure: The first thing we have to do is install the react-router package because it is not a part of the core react library.

```
npm install react-router-dom
```

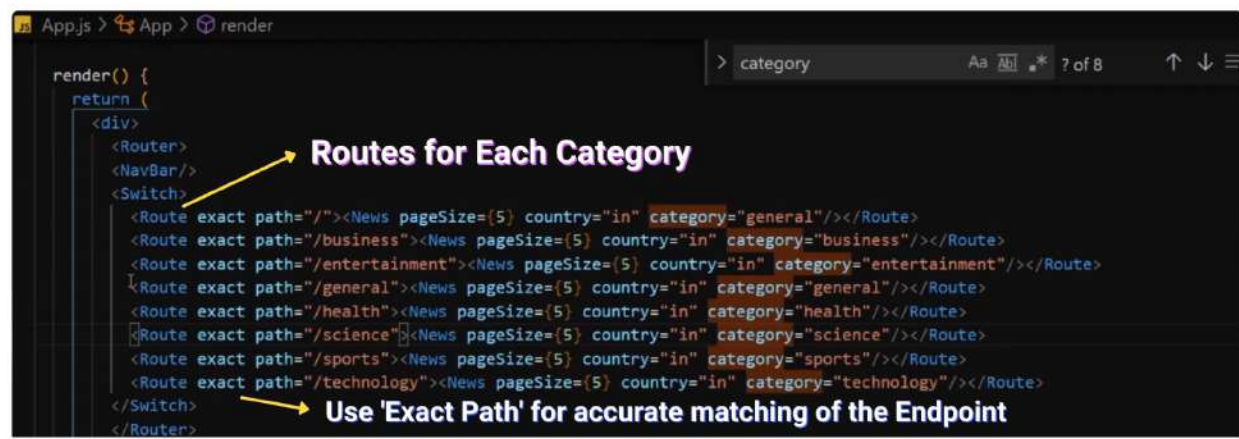
Set Up routing for our application

Procedure: After installing the package, The first thing you have to do is go to the root component, that is, your "app.js" file, and import a few things from the react-router package. We are going to import the Browser Router as Router, Switch, Route, and Link from react-router-dom. To import the following using the below command:

```
import {BrowserRouter as Router, Switch, Route, Link} from "react-router-dom"
```

In App.js

- 1. **Using Router:** We have to surround our entire application with the router component and it means that we can use the router in the entire application. As a result, all components that are nested inside this app component(app.js) get access to the router. Hence, To surround your app with the router component, use `<Router></Router>`
- 2. **Using Switch:** The next step is to decide where we want our page content to go when we go to different pages. Since we want to access the different News items with Update Props. So, we are going to use the switch component (`<switch></switch>`). The switch component makes sure that only one route shows at any one time. All of our routes go inside this switch component.
- 3. **Using Route:** Alright, we need to set up our individual routes for the pages of different Categories. So, we will create a route for each page, for which we will be using the route component(`<route></route>`). At this moment, we have General, Business, Entertainment, Health, Science, Sports, Technology pages of our application, and hence we are going to place the same number of routes inside this switch component.



The screenshot shows a code editor with the following content:

```
App.js > App > render
render() {
  return (
    <div>
      <Router>
      <NavBar/>
      <Switch>
        <Route exact path="/"><News pageSize={5} country="in" category="general"/></Route>
        <Route exact path="/business"><News pageSize={5} country="in" category="business"/></Route>
        <Route exact path="/entertainment"><News pageSize={5} country="in" category="entertainment"/></Route>
        <Route exact path="/general"><News pageSize={5} country="in" category="general"/></Route>
        <Route exact path="/health"><News pageSize={5} country="in" category="health"/></Route>
        <Route exact path="/science"><News pageSize={5} country="in" category="science"/></Route>
        <Route exact path="/sports"><News pageSize={5} country="in" category="sports"/></Route>
        <Route exact path="/technology"><News pageSize={5} country="in" category="technology"/></Route>
      </Switch>
    </Router>
  )
}
```

Annotations in the image:

- A yellow arrow points from the text "Routes for Each Category" to the `<Switch>` tag.
- A yellow arrow points from the text "Use 'Exact Path' for accurate matching of the Endpoint" to the `exact` attribute in the first `<Route>` tag.

Figure 1.1: Using React Router in NewsMonkey

Remember, that we are using the `<Route Exact path= "/" />` for the exact matching of the endpoint.

Point to be noted: React won't render the NewsComponent again while navigating through different categories as it will render the NewsComponent for the first request. But we want to rebound the News component with the Updated Props. To fix this issue we would add a unique key prop to every route as shown below:



```
render() {
  return (
    <div>
      <Router>
      <NavBar/>
      <Switch>
        <Route exact path="/"><News key="general" pageSize={5} country="in" category="general"/></Route>
        <Route exact path="/business"><News key="business" pageSize={5} country="in" category="business"/></Route>
        <Route exact path="/entertainment"><News key="entertainment" pageSize={5} country="in" category="entertainment"/></Route>
        <Route exact path="/general"><News key="general" pageSize={5} country="in" category="general"/></Route>
        <Route exact path="/health"><News key="health" pageSize={5} country="in" category="health"/></Route>
        <Route exact path="/science"><News key="science" pageSize={5} country="in" category="science"/></Route>
        <Route exact path="/sports"><News key="sports" pageSize={5} country="in" category="sports"/></Route>
        <Route exact path="/technology"><News key="technology" pageSize={5} country="in" category="technology"/></Route>
      </Switch>
    </Router>
  </div>
)
```

Figure 1.2: Using Key prop to Rebound NewsComponent

Result: We would be displaying the News of a specific Category while the selected path is used as an endpoint in the URL.

Note: Since we have already learned about react-router so we won't be discussing the React router in detail.

In Navbar.js:

We would like to show the specific news when the user selects the desired category. For example, when the user visits the sports category then we want to display the news articles related to sports, to do this we have to refactor the navbar component. We would replace the "/about" endpoint with different endpoints related to those specific news categories, as shown below:

```
<div className="collapse navbar-collapse" id="navbarSupportedContent">
  <ul className="navbar-nav me-auto mb-2 mb-lg-0">
    <li className="nav-item">
      <a className="nav-link" aria-current="page" href="/">Home</a>
    </li>
    <li className="nav-item"><a className="nav-link" href="/business">Business</a></li>
    <li className="nav-item"><a className="nav-link" href="/entertainment">Entertainment</a></li>
    <li className="nav-item"><a className="nav-link" href="/general">General</a></li>
    <li className="nav-item"><a className="nav-link" href="/health">Health</a></li>
    <li className="nav-item"><a className="nav-link" href="/science">Science</a></li>
    <li className="nav-item"><a className="nav-link" href="/sports">Sports</a></li>
    <li className="nav-item"><a className="nav-link" href="/technology">Technology</a></li>
  </ul>
</div>
```

In Navbar.js

Changing the Endpoints of the Categories

Figure 1.3: Changing the Endpoints of the Categories

Fixing the Reloading issue

You might have noticed that the page is reloading while switching between different categories so to solve this we would replace the "a href" keyword with 'link to' in "Navbar.js". To do so firstly, Make sure to import the link from react-router-dom.

```
<div className="collapse navbar-collapse" id="navbarSupportedContent">
  <ul className="navbar-nav me-auto mb-2 mb-lg-0">
    <li className="nav-item">
      <Link className="nav-link" aria-current="page" to="/">Home</Link>
    </li>
    <li className="nav-item"><Link className="nav-link" to="/business">Business</Link></li>
    <li className="nav-item"><Link className="nav-link" to="/entertainment">Entertainment</Link></li>
    <li className="nav-item"><Link className="nav-link" to="/general">General</Link></li>
    <li className="nav-item"><Link className="nav-link" to="/health">Health</Link></li>
    <li className="nav-item"><Link className="nav-link" to="/science">Science</Link></li>
    <li className="nav-item"><Link className="nav-link" to="/sports">Sports</Link></li>
    <li className="nav-item"><Link className="nav-link" to="/technology">Technology</Link></li>
  </ul>
</div>
```

Fixing the Reloading Page Issue

Using 'link to' in place of a href tag

Figure 1.4: Using 'Link to' instead of 'a href'

Result: Now, we can switch between the News of different categories without reloading the page. Here's a look at the NewsMonkey application.

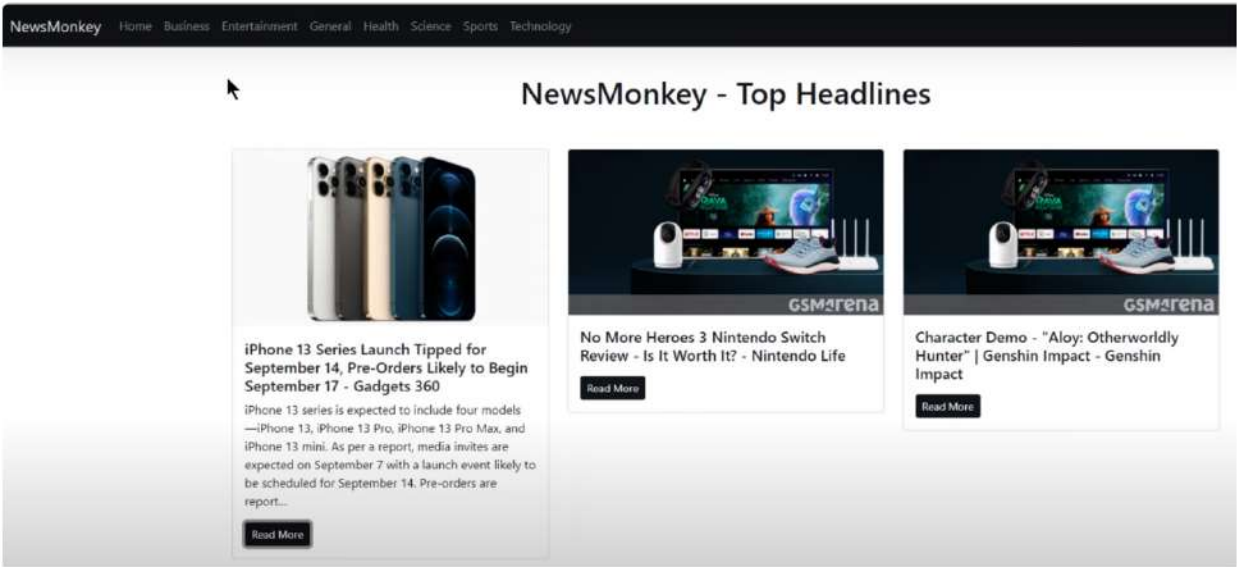


Figure 1.5: The NewsMonkey Application