EPISODE 7 : FINDING THE PATH

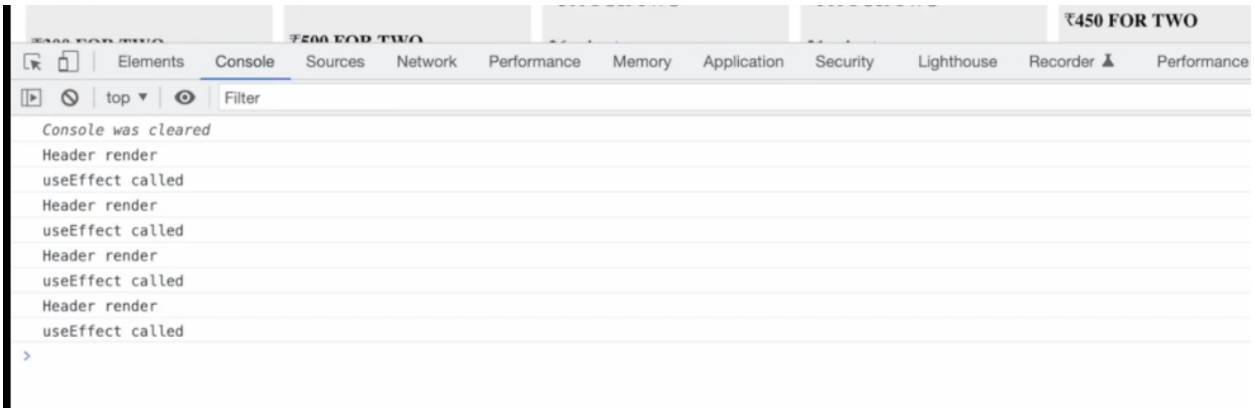
PART 01:BEHAVIOR OF USEEFFECT

Syntax of useEffect

UseEffect(()=>{
},[])

- **When there is no dependency array**
  UseEffect(()=>{
  })
  In this case the useEffect will be called at every render of component.

```
// if no dependency array => useEffect is called on every render
useEffect(() => {
  console.log("useEffect called");
});
```



- **When dependency array is empty.**
  In this case useEffect is called on initial render only. I.e just once when the component renders for the first time.

```
//if dependency array is empty = [] => useEffect is called on initial render(just once)
useEffect(() => {
  console.log("useEffect called");
}, []);
```

```
Elements  Console  Sources  Network  Performance  Memory  Application  Security  Lighthouse  Recorder  Performance
top ▾  Filter
Download the React DevTools for a better development experience: https://reactjs.org/link/react-devtools
Header render
useEffect called
9 Header render
>
```

When there is something in the Dependency array

```
// if dependency array is [btnNameReact] = > called everytime btnNameReact is updated
useEffect(() => {
  console.log("useEffect called");
}, [btnNameReact]);
```

```
Elements  Console  Sources  Network  Performance  Memory  Application  Security  Lighthouse  Recorder  Perfor
top ▾  Filter
Download the React DevTools for a better development experience: https://reactjs.org/link/react-devtools
Header render
useEffect called
Header render
useEffect called
Header render
useEffect called
Header render
useEffect called
>
```

PART 02 : best practices for useState

1.Never create state variables outside the components , if you do so you will end up with error.

```
...
import RestaurantCard from "./RestaurantCard";
import { useState, useEffect } from "react";
import Shimmer from "./Shimmer";

const [searchText, setSearchText] = useState("");

const Body = () => {
  // Local State Variable - Super powerful variable
```

TypeError: Cannot read properties of null (reading 'useState')

```
useState
node_modules/react/cjs/react.development.js:1622:20

  1619 | }
  1620 | function useState(initialState) {
  1621 |   var dispatcher = resolveDispatcher();
> 1622 |   return dispatcher.useState(initialState);
  1623 | }
  1624 | function useReducer(reducer, initialArg, init) {
  1625 |   var dispatcher = resolveDispatcher();

View compiled

8yaV8.react/jsx-dev-runtime
src/components/Body.js:5:45

  2 | import { useState, useEffect } from "react";
  3 | import Shimmer from "./Shimmer";
  4 |
> 5 | const [searchText, setSearchText] = useState("");
  6 |
  7 | const Body = () => {
  8 |   // Local State Variable - Super powerful variable

View compiled
```

2. Always try to call these hooks on the top of the component.

```
You, 1 second ago | 1 author (You)
import RestaurantCard from "./RestaurantCard";
import { useState, useEffect } from "react";
import Shimmer from "./Shimmer";

const Body = () => {
  // Local State Variable - Super powerful variable      You, 3 months ago • episode-05
  const [listOfRestaurants, setListOfRestraunt] = useState([]);
  const [filteredRestaurant, setFilteredRestaurant] = useState([]);

  const [searchText, setSearchText] = useState("");

  // Whenever state variables update, react triggers a reconciliation cycle(re-renders the compo
  console.log("Body Rendered");

  useEffect(() => {
    fetchData();
```


3.Never use your usestate hook or create useState variable inside the if else block, for loops ,while loops or even inside a function .Because this can create inconsistency in the program.

```
if() {
  const [searchText, setSearchText] = useState("");
}
```

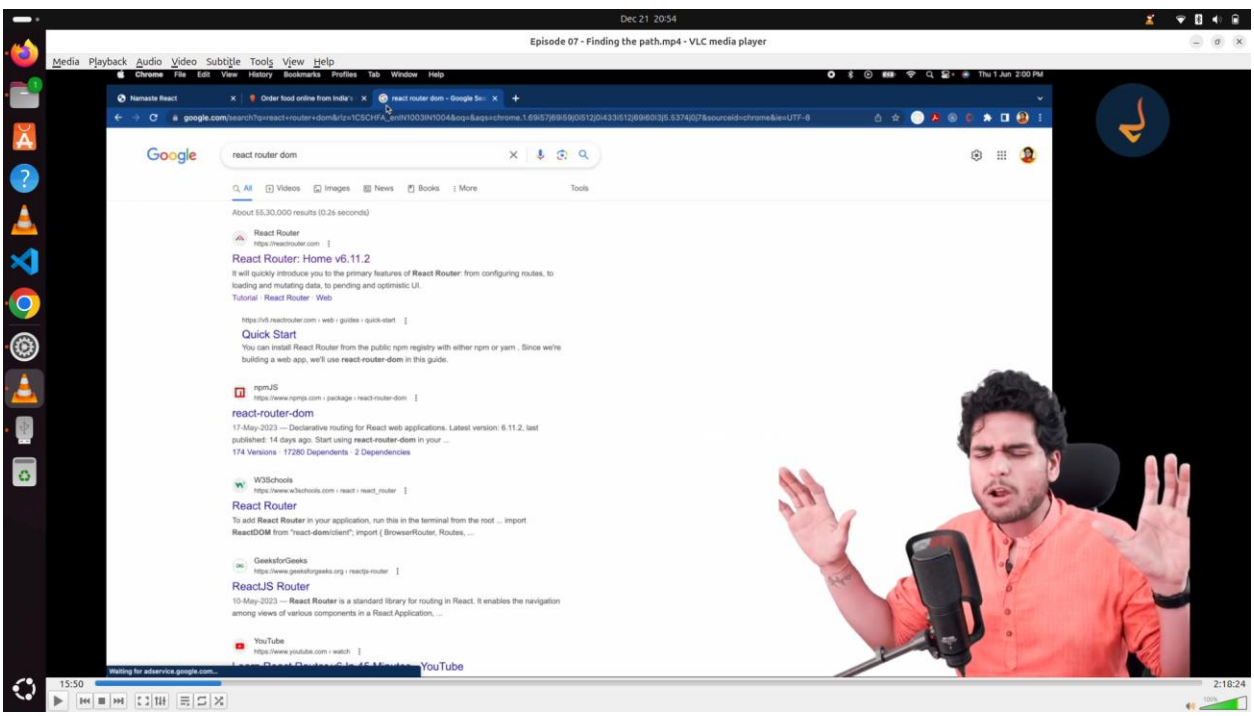
```
for() {        You, now • Uncommitted changes
  const [searchText, setSearchText] = useState("");
}
```

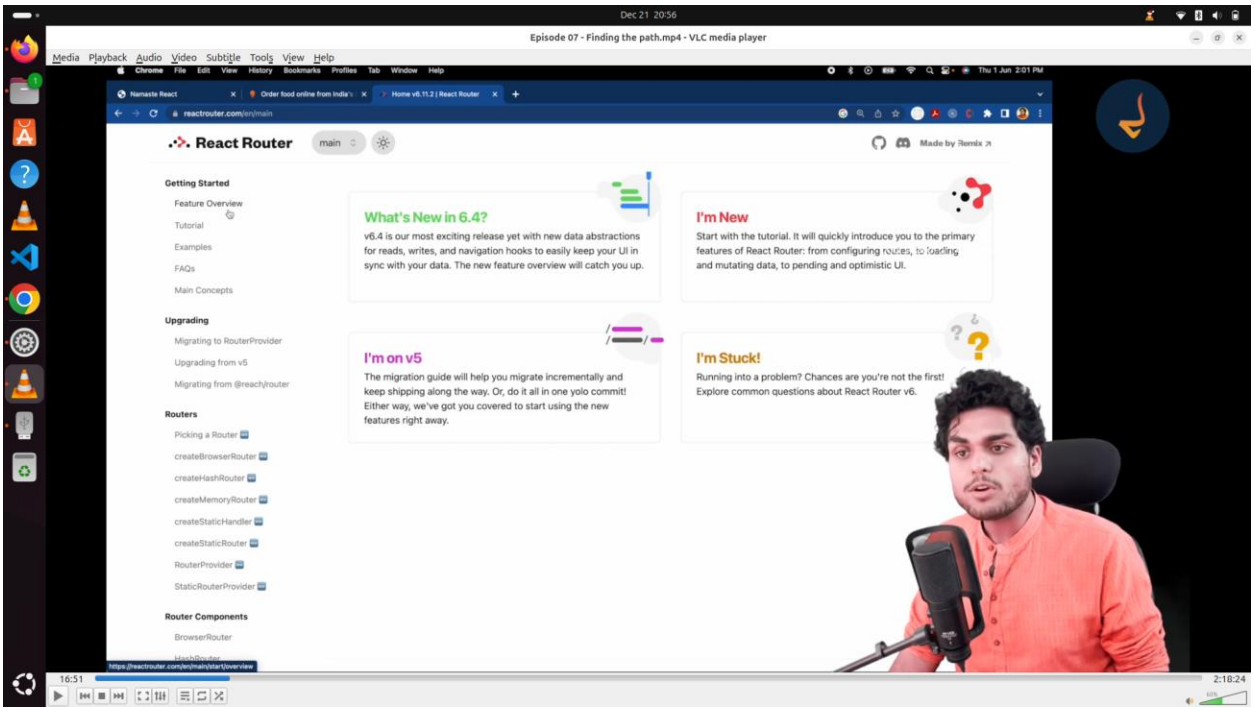
```
function () {
  const [searchText, setSearchText] = useState("");
}
```

So useState are meant to be created only in the functional component in the top at the higher level.

PART 03 :

Learning about react router





Here we are using the version 6.5 for that

e need to install react router library into the code.

Enter the below command to install react router.

npm i react-router-dom

Then create routes as follows.

For creating routes we need to import createBrowserRouter to create routes and RouterProvider to provide routes to the browser , we need to do in the App.js only

```
import { createBrowserRouter,RouterProvider } from "react-router-dom";
```

src > JS App.js > ...

You, 6 hours ago | 2 authors (You and one other)

```
import React from "react";
import ReactDOM from "react-dom/client";
import Header from './components/Header'
import Body from './components/Body'
import { createBrowserRouter,RouterProvider } from "react-router-dom";
import About from "./components/About";
import Contact from "./components/Contact";
import { ImportContacts } from "@mui/icons-material";
const AppLayout = () => {
  return(
    <div>
      <Header/>
      <Body/>
    </div>
  )
}
const appRouter = createBrowserRouter([
  {
    path:"/",
    element:<AppLayout />,
    errorElement:<Error/>
  },
  {
    path:"/about",
    element:<About />,
  },
  {
    path:"/contact",
    element:<Contact />,
  }
])
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(<RouterProvider router={appRouter}/>)
```

You, 6 hours ago • Uncommitted changes

JS Header.js M | JS Body.js | JS About.js U | JS Error.js U | JS Contact.js U | JS App.js M

src > components > JS Error.js > ...

```
import { useRouteError } from "react-router-dom";

const Error = () => {
  const err = useRouteError();
  console.log(err);
  return (
    <div>
      <h1>Oops!!!</h1>
      <h2> Something went wrong!!</h2>
      <h3>
        {err.status}: {err.statusText}
      </h3>
    </div>
  );
};

export default Error;
```

Actually after creating the paths , if we give wrong path (ex: http://localhost:1234/xyz) we get an error page which is handled by react router dom itself, but we can customize our own Error page as creating Error component like above.

In the above way of creating paths , if we use ” / ” we get page with header and body component

, but for about, contact us pages the pages will be rendered but we cannot see the header by adding Outlet.

To achieve header for each page we need to make them as child routes like below.

```
const AppLayout = () => {
    <div>
      <Header/>
      <Outlet/>

    </div>
  )
}
const appRouter = createBrowserRouter([
  {
    path: "/",
    element: <AppLayout/>,
    children: [
      {
        path: "/",
        element: <Body/>
      },
      {
        path: "/about",
        element: <About/>
      }
    ],
    ErrorElement: <Error/>
  },
  {
    path: "/about",
    element: <About/>
  }
])
```

```
const AppLayout = () => {
    <div>
      <Header/>
      <Outlet/>

    </div>
  )
}
const appRouter = createBrowserRouter([
  {
    path: "/",
    element: <AppLayout/>,
    children: [
      {
        path: "/",
        element: <Body/>
      },
      {
        path: "/about",
        element: <About/>
      }
    ],
    ErrorElement: <Error/>
  },
  {
    path: "/about",
    element: <About/>
  }
])
```

By this when we browse:

http://localhost:1234/ -> goes to Body

http://localhost:1234/about -> goes to about

Now we can make to go to contact us, about pages when we click on contactus and about us pages in Header

If we want to make home , About Us, contact us etc., as links we can generally make those unordered list items as links using <a> anchor tags

```
import { useState } from "react";
import { LOGO } from "../utils/constants";
const Header = ()=>{
  const [login,setLogin] = useState("Login")
  return(
    <div className="header">
        <div className="logo">
          <img id="image" src={LOGO} alt="logo"/>
        </div>
        <div className="nav-items">
          <ul>
            <li><a href="Body">Home</a></li>
            <li><a href="Contact">Contact us</a></li>
            <li><a href="about">About</a></li>
            <li>Cart</li>
            <button className="login-btn"
              onClick={()=>(
                login=="Login"?setLogin('Logout'):setLogin('Login')

              )}
            >
              {login}
            </button>
          </ul>

        </div>
    </div>
```

Now they became clickable links

But using anchor tags is not a good idea because when you switch from one page to another the entire page will be loaded.

To avoid this we can follow the below approach

So instead of using
 Contact Us

We can use

<Link to=” ContactUs”> Contact Us </Link>

To use Link we need to import it

```
import { Link } from "react-router-dom"
```

```
import { useState } from "react";
import { LOGO } from "../utils/constants";
import { Link } from "react-router-dom"
const Header = ()=>{
  const [login,setLogin] = useState("Login")
  return(
    <div className="header">
      <div className="logo">
        <img id="image" src={LOGO} alt="logo"/>
      </div>
      <div className="nav-items">
        <ul>
          <li><Link to="Body">Home</Link></li>
          <li><Link to="Contact">Contact us</Link></li>
          <li><Link to="About">About</Link></li>
          <li>Cart</li>
          <button className="login-btn"
            onClick={()=>(
              login=="Login"?setLogin('Logout'):setLogin('Login')

            )}
          >
            {login}
          </button>
        </ul>

      </div>
```

By using this method whole page not reloads only the components refreshes .

That is why we call react applications as single page applications

There are 2 types of Routing in Web apps

1. Client-side Routing
2. Server-side Routing- for example if you have index.html, contact.html, about.html . If you click on anchor tag /about it reloads the whole page fetches that html and renders it on to the web page

But what we are currently using is a Client-side Routing, Because when we load the app for the first time all the components are loaded into our app , it just loads the components