Reuse Component in Loop

App.js

import './App.css';

import React from 'react';

import User from './User';

import {Table} from 'react-bootstrap'

function App(){

    const user=[

        {

            name:"ravi",email:"ravi@123",id:10

        },

        {

            name:"kavi",email:"kavi@123",id:20

        },

        {

            name:"bhavi",email:"bhavi@123",id:30

        }

    ]

    return(

        <div className="App">

            <Table striped variant="dark">

                <tbody>

                    <tr>

                        <th className="text-uppercase">Name</th>

                        <th className="text-uppercase">email</th>

                        <th className="text-uppercase">id</th>

                    </tr>

            {

                user.map((items,i)=>

                <User data={items} />

                )

            }

            </tbody>

            </Table>

        </div>

    );

}

export default App;

User.js

import React from "react";

function User(props){

    return(

           <tr>

               <td>{props.data.name}</td>

               <td>{props.data.email}</td>

               <td>{props.data.id}</td>

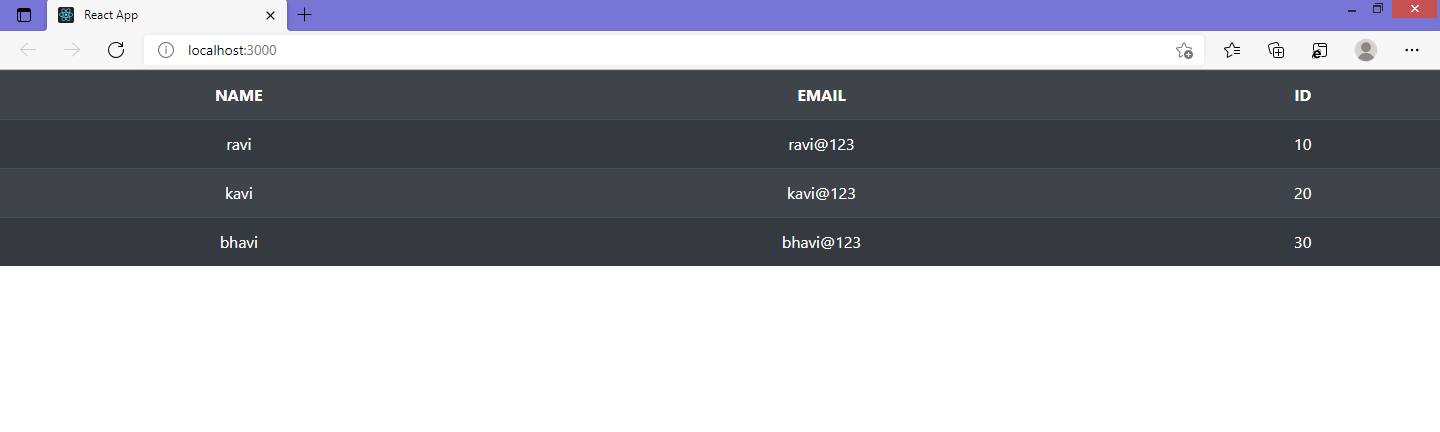
           </tr>

    );

}

export default User;

Like This:-



Fragment

It’s used to wrapping error to < Selector >..!< Selector /> solution and multiple children component for using the parent component it’s error to solve.

App.js

import './App.css';

import React, { Fragment } from 'react';

import User from './User';

function App(){

    return(

        <div className="App">

        <h1>baagi</h1>

        <h1>baagi</h1>

        <User/>

        </div>

    );

}

export default App;

User.js

import React, { Fragment } from "react";

function User(){

    return(

        <Fragment> // this is advance feature on react.

               <h1>bhai</h1>

               <h1>hello</h1>

        </Fragment>

    );

}

export default User;

Lifting State Up

This methods are used to child component data it’s use to parent component that’s called lifting state up.

App.js

import './App.css'

import React from 'react'

import User from './User';

function App(){

    function parentAlert(data){

console.log(data)

        alert(data.email)//for object

    }

    return(

        <div className="App">

            <h1>Lifting State Up</h1>

            <User data={parentAlert}/>

        </div>

    );

}

export default App;

User.js

import React, { Fragment } from "react";

function User(props){

    let items=

    {

        name:"bhaiya",email:"bhai@122", number:876238765

    }

    return(

        <Fragment>

               <h3>data : </h3>

               <button onClick={()=>props.data(items)}>Give Me</button>

        </Fragment>

    );

}

export default User;

PureComponent

PureComponent is used to user Calling the function in multi-time and render is call at that time. So the stoping re-rendering component it’s called PureComponent.

import './App.css'

import React, { PureComponent } from 'react'

import User from './User';

class App extends PureComponent{

    constructor(){

        super();

        this.state=

        {

            num:0

        }

    }

    render()

    {

        console.warn("re-rendering")

        return(

            <div className="App">

                <h1>num : {this.state.num}</h1>

                <button onClick={()=>this.setState({num:this.state.num})}>Update</button>

            </div>

        );

    }

}

export default App;

import User.js :

import React from 'react'

class User extends React.PureComponent {

    render()

    {

        return(

            <div className="App">

                <h1>data: {this.props.count}</h1>

            </div>

        );

    }

}

export default User;

Controlled Component

It’s used to any input field data access with states that’s called controlled component.

import './App.css'

import React,{useState} from 'react'

export default function App(){

    const[data,setData]=useState("null")

    return(

        <div className="App">

            <input type="text" value={data} onChange={(e)=>setData(e.target.value)}/>

            <h1>{data}</h1>

        </div>

    );

}

Uncontrolled Component

It’s used to any input field data on using JavaScript, DOM and Ref function are access.

And the input fields aren’t access of states that’s called Uncontrolled Component.

import './App.css'

import React, { useRef } from 'react'

export default function App() {

    let inputRef=useRef(null);

    let inputRef1=useRef(null);

    function submitForm(e) {

        e.preventDefault()

        console.warn(inputRef.current.value )

        console.warn(inputRef1.current.value )

        let input3=document.getElementById('input2').value

        console.warn(input3)

    }

    return (

        <div className="App">

            <form onSubmit={submitForm}>

                <input type="text" ref={inputRef}/><br /><br />

                <input type="text" ref={inputRef1}/><br /><br />

                <input type="text" id="input2" /><br /><br />

                <button>Submit</button>

            </form>

        </div>

    );

}

HOC

(High Order Component)

A component is any another component in using like props so that is high order component.

That component as an input used and output used.

import './App.css';

import React,{useState} from 'react';

function App(){

    return(

        <div className="App">

            <h1>HOC</h1>

            <HOCred hoc={Counter}/>

            <HOCgreen hoc={Counter}/>

            <HOCblue hoc={Counter}/>

        </div>

    );

}

function HOCred(props){

    return(

        <div style={{backgroundColor:"red", width:200}}>

            <props.hoc/>

        </div>

    );

}

function HOCgreen(props){

    return(

        <div style={{backgroundColor:"green", width:200}}>

            <props.hoc/>

        </div>

    );

}

function HOCblue(props){

    return(

        <div style={{backgroundColor:"blue", width:200}}>

            <props.hoc/>

        </div>

    );

}

function Counter(){

    const [count,setCount]=useState(0)

    return(

        <div>

            <h3>{count}</h3>

            <button onClick={()=>setCount(count+1)}>Update</button>

        </div>

    );

}

export default App;

Router

This is most popular library for rooting applications or components fetching to parent component.

**Install process: version: 5.3.0**

npm i react-router-dom@5.3.0

And there hooks:

**BrowerRouter:** Its hook using the main parent tag for rooting in a single page.

**Link:** Its hook use to rooting link of redirect in a single page (just like navigator).

**Route:** It’s hook using the root path of Route tag and there statement pass on child component for Router.

**Switch:** That is wrapping tag for multi- Route tag in single page. It’s reduce complexity codding and easy to debugging.

import './App.css';

import React from 'react';

import { BrowserRouter as Router, Link, Switch, Route} from 'react-router-dom';

function App() {

    return (

        <div className="App">

            <Router>

                <Link to="/home">Home</Link> &nbsp;&nbsp;

                <Link to="/about">About</Link> &nbsp;&nbsp;

                <Link to="/blog">Blog</Link>

                <Route path="/home"><Home/></Route>

                <Route path="/about"><About/></Route>

                <Route path="/blog"><Blog/></Route>

            </Router>

        </div>

    );

}

function Home() {

    return (

        <div>

            <h4>HOME</h4>

            <p>Lorem Ipsum placeholder text for use in your graphic</p>

        </div>

    );

}

function About() {

    return (

        <div>

            <h4>ABOUT</h4>

            <p>Lorem Ipsum placeholder text for use in your graphic</p>

        </div>

    );

}

function Blog() {

    return (

        <div>

            <h4>BLOG</h4>

            <p>Lorem Ipsum placeholder text for use in your graphic</p>

        </div>

    );

}

export default App;



Perfect Format in Router

**Index.js :-**

import React from 'react';

import ReactDOM from 'react-dom';

import './index.css';

import App from './App';

import 'bootstrap/dist/css/bootstrap.min.css';

import reportWebVitals from './reportWebVitals';

import {BrowserRouter as Router} from 'react-router-dom'

ReactDOM.render(

  <React.StrictMode>

    <Router> {/\*using BrowserRouter as a Router \*/}

<App />

    </Router>

  </React.StrictMode>,

  document.getElementById('root')

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

**App.js:-**

import React from "react";

import {Link,Switch,Route} from 'react-router-dom'

import './App.css'

export default function App(){

    return(

        <div className="App">

                <Link to="/home">Home</Link>

                <Link to="/about">About</Link>

                <Switch>

                <Route path="/home" exact={true}><Home/></Route>

                <Route path="/about"><About/></Route>

                <Route path="\*"><Page404/></Route>

                </Switch>

        </div>

    );

}

404 Page Error

This page is undefined or page not existing as default page is using 404 page error.

import React from "react";

import {Link,Switch,Route} from 'react-router-dom'

import './App.css'

export default function App(){

    return(

        <div className="App">

                <Link to="/">Home</Link>

                <Link to="/about">About</Link>

                <Switch>

                <Route path="/" exact={true}><Home/></Route>

                <Route path="/about"><About/></Route>

                <Route path="\*"><Page404/></Route>

                </Switch>

        </div>

    );

}

function Home(){

    return(

        <>

        <h1>welcome Home</h1>

        <p>welcome Home</p>

        </>

    );

}

function About(){

    return(

        <>

        <h1>welcome About</h1>

        <p>welcome About</p>

        </>

    );

}

function Page404(){

    return(

        <>

        <h1>404</h1>

        <p>This Page Not Found</p>

        </>

    );

}

Dynamic Routing

In case user not defined or not identify in router for page so it can use the dynamic routing.

Dynamic routing is basically, it’s create in loop for router and that’s based on id’s, key’s and unique code.

**App.js:-**

import React from 'react'

import './App.css'

import { Link,Route} from 'react-router-dom'

import User from './User'

export default function App() {

    let users = [

        { id: 1, name: "lucky", email: "lucky#$%.com" },

        { id: 2, name: "harsh", email: "harsh#$%.com" },

        { id: 3, name: "nikhil", email: "nikhil#$%.com" },

        { id: 4, name: "buggy", email: "buggy#$%.com" }

    ]

    return (

        <div className="App">

            <h1>Dynamic Routing</h1>

            {

                users.map((item) =>

                    <div>

                       <Link to={"/user/" + item.id+"/"+item.name}><h3>{item.name}</h3></Link>

                    </div>

                )

            }

            <Route path="/user/:id/:name"><User/></Route>

        </div>

    )

}

**User.js:-**

**withRouter:-** However, you’ll often have the need for deeply nested components, so React Router includes the withRouter method to give you access to the match, location, and history properties of the browser’s native API necessary to make your Link and Route components work correctly.

import {withRouter} from 'react-router-dom';

 function User(prop){

    console.log(prop)

    return(

        <div>

            <h2>{prop.match.params.id}</h2>

            <h2>{prop.match.params.name}</h2>

        </div>

    )

}

export default withRouter(User);