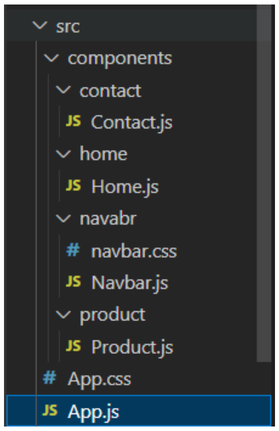
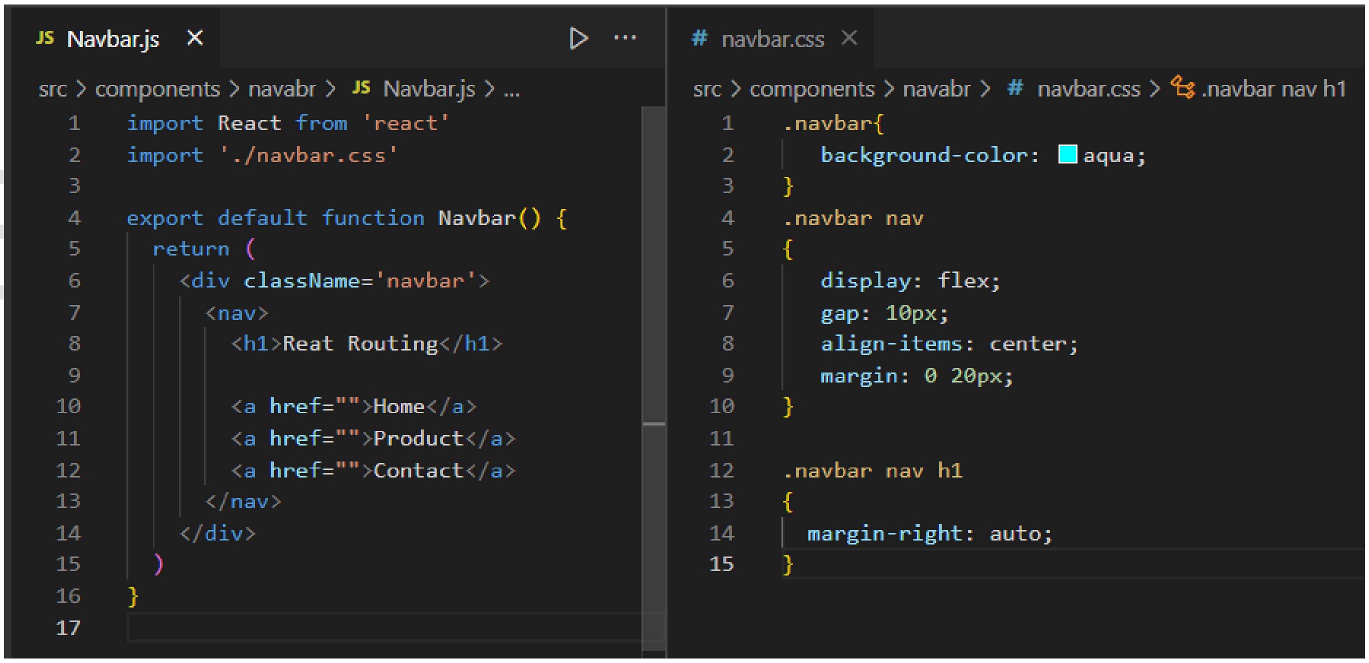
1. >npx create-react-app routing
2. Create components folder in src folder
    a. Create 4 folders in src – navbar, home, product, contact
    b. Create js file for each in respective folder



```
src
  components
    contact
      Contact.js
    home
      Home.js
    navabr
      navbar.css
      Navbar.js
    product
      Product.js
  App.css
  App.js
```

3. Code for navbar



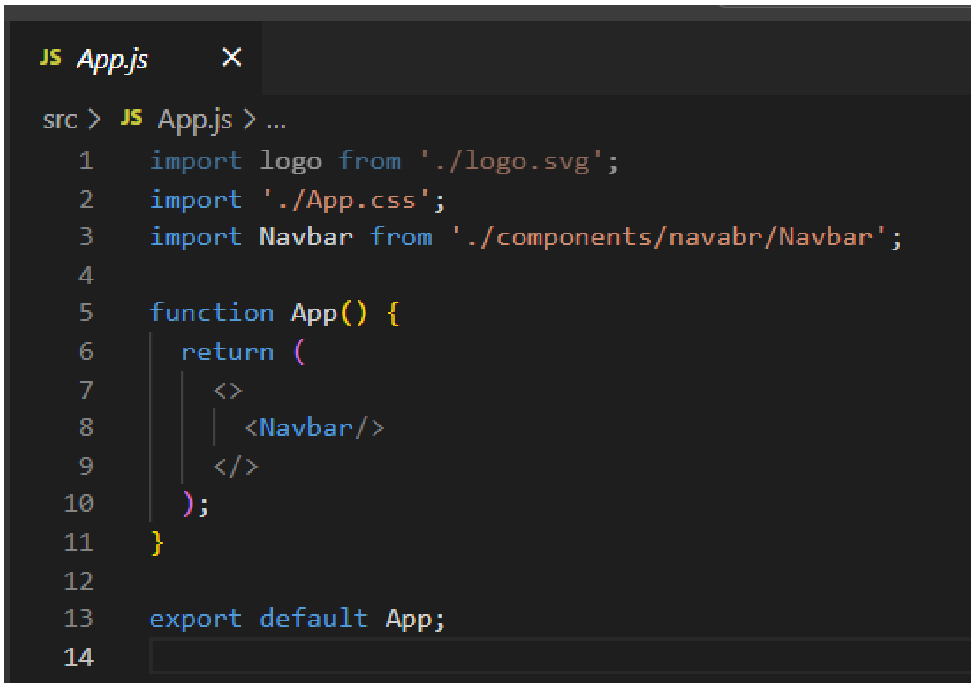
Navbar.js
src > components > navabr > Navbar.js > ...
navbar.css
src > components > navabr > navbar.css > .navbar nav h1

```js
import React from 'react'
import './navbar.css'

export default function Navbar() {
  return (
    <div className='navbar'>
      <nav>
        <h1>Reat Routing</h1>

        <a href="">Home</a>
        <a href="">Product</a>
        <a href="">Contact</a>
      </nav>
    </div>
  )
}
```

```css
.navbar{
   background-color: aqua;
}
.navbar nav
{
   display: flex;
   gap: 10px;
   align-items: center;
   margin: 0 20px;
}

.navbar nav h1
{
  margin-right: auto;
}
```

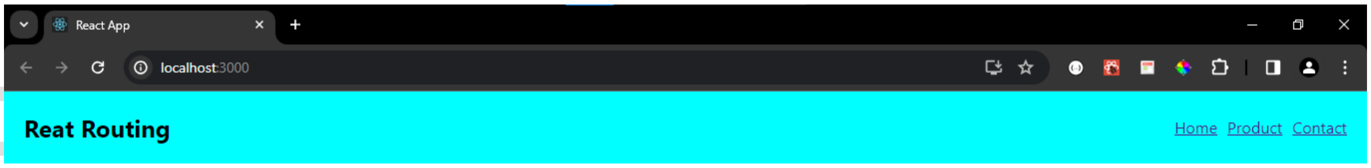
Tejas Kasare   +91 8459859415

4. Display navbar component in App compoenet



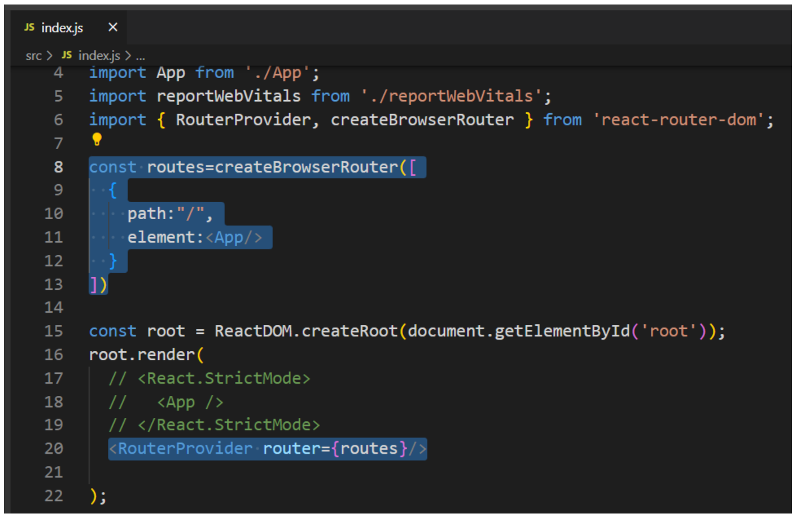
```
import logo from './logo.svg';
import './App.css';
import Navbar from './components/navabr/Navbar';

function App() {
  return (
    <>
      <Navbar/>
    </>
  );
}

export default App;
```

5. Npm start and check output



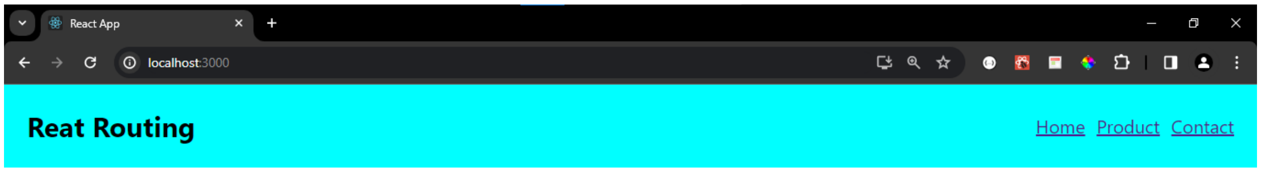
Reat Routing Home Product Contact

6. Implementing routing
   a. >npm i react-router-dom
   b. Create router array and attach it to the RouterProvider as property in index.js



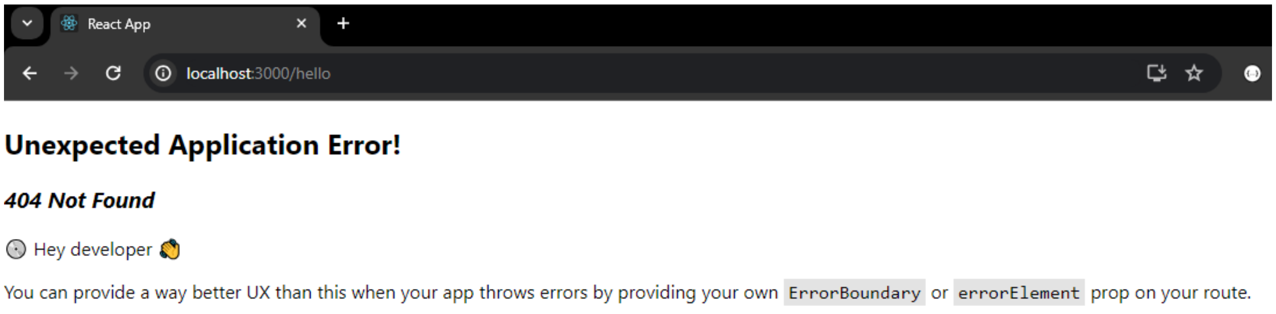
```
import App from './App';
import reportWebVitals from './reportWebVitals';
import { RouterProvider, createBrowserRouter } from 'react-router-dom';

const routes=createBrowserRouter([
  {
    path:"/",
    element:<App/>
  }
])

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  // <React.StrictMode>
  //   <App />
  // </React.StrictMode>
  <RouterProvider router={routes}/>

);
```

   c. Output



Reat Routing Home Product Contact

d. If we try to provide any other path than "/" we will get error



Unexpected Application Error!

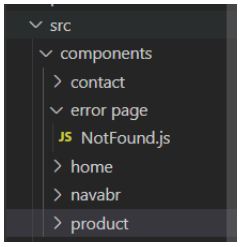
404 Not Found

Hey developer 👋

You can provide a way better UX than this when your app throws errors by providing your own `ErrorBoundary` or `errorElement` prop on your route.

7. Handling Error page

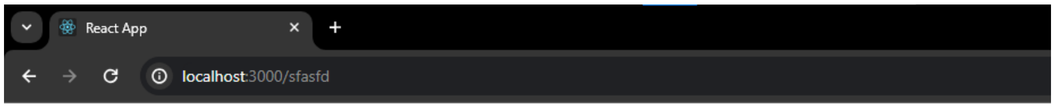
a. Create error page folder inside compoenets folder. In error page folder create NotFound.js

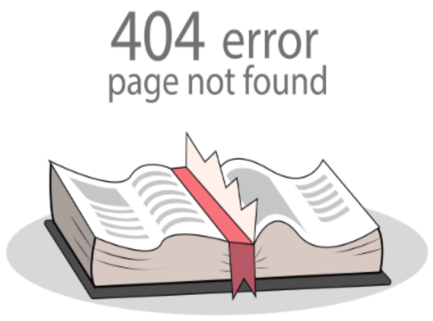


b. Code for NotFound.js

```
import React from 'react'

export default function NotFound() {
  return (
   <center>
          <img src="https://cdn2.hubspot.net/hubfs/242200/shutterstock_774749455.jpg" alt=""
height={"400px"} width={"500px"}/>
   </center>
  )
}
```

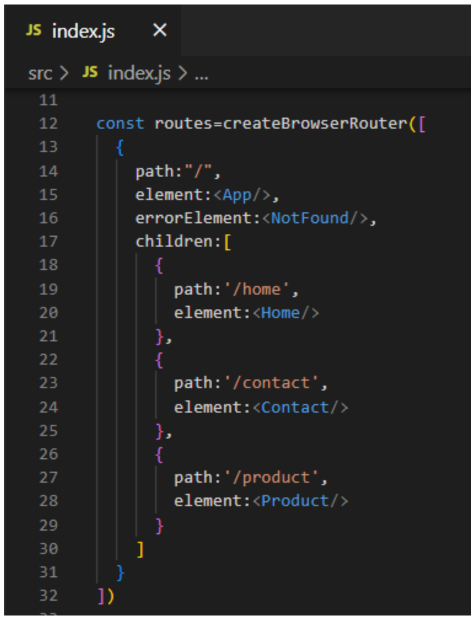
c. Check output





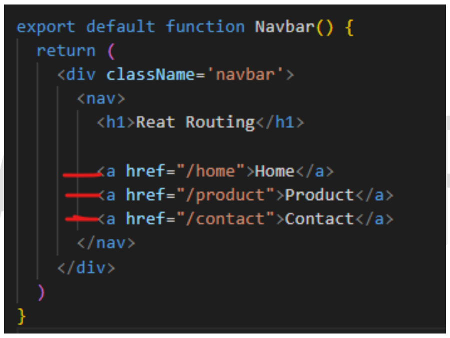
8. Handling routing from navbar

   a. In index.js provide children array



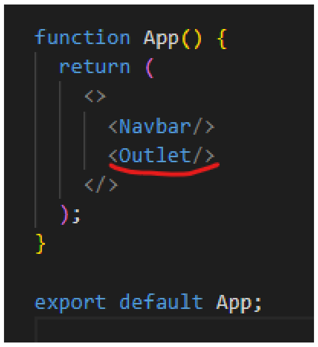
```
const routes=createBrowserRouter([
  {
    path:"/",
    element:<App/>,
    errorElement:<NotFound/>,
    children:[
      {
        path:'/home',
        element:<Home/>
      },
      {
        path:'/contact',
        element:<Contact/>
      },
      {
        path:'/product',
        element:<Product/>
      }
    ]
  }
])
```

   b. In Navbar.js provide links



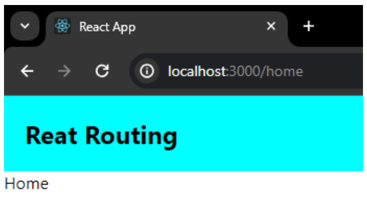
```
export default function Navbar() {
  return (
    <div className='navbar'>
      <nav>
        <h1>Reat Routing</h1>

        <a href="/home">Home</a>
        <a href="/product">Product</a>
        <a href="/contact">Contact</a>
      </nav>
    </div>
  )
}
```

   c. In App.js provide <Outlet/> for component switching



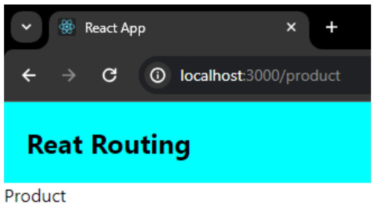
```
function App() {
  return (
    <>
      <Navbar/>
      <Outlet/>
    </>
  );
}

export default App;
```

   d. Output



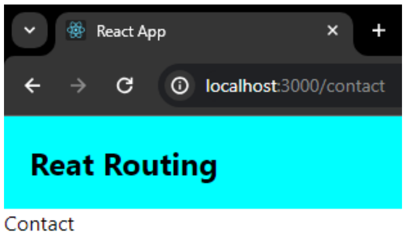
Reat Routing

Home



Reat Routing

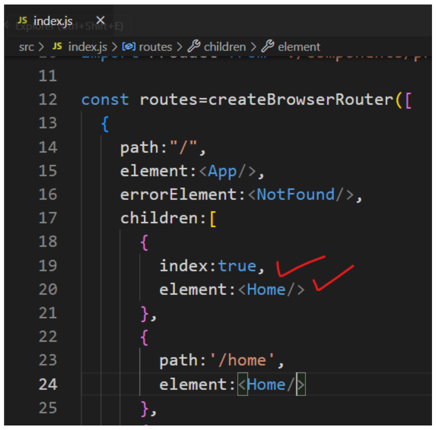
Product



Reat Routing

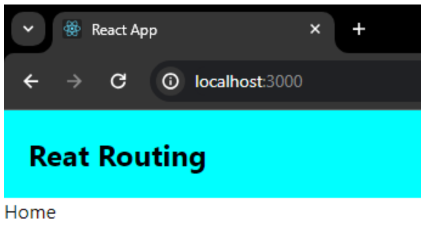
Contact

9. Handling index page (displaying home component as home/index page)
   a. In index.js provide index:true



```
src > index.js > routes > children > element

const routes=createBrowserRouter([
  {
    path:"/",
    element:<App/>,
    errorElement:<NotFound/>,
    children:[
      {
        index:true,
        element:<Home/>
      },
      {
        path:'/home',
        element:<Home/>
      },
```

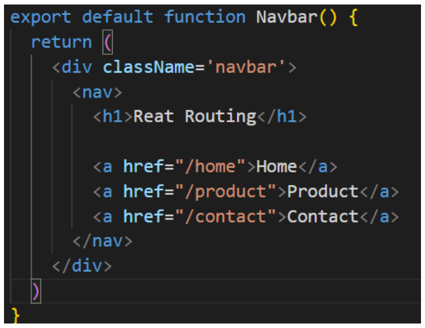
   b. Output



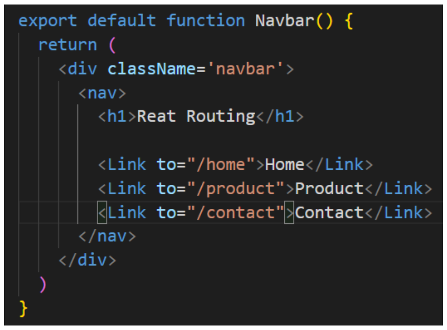
```
Reat Routing
Home
```

   c.

10. Preventing page from getting refersh
   a. If we click on home/contact/product from navbar then ypu will see, our page is getting refersh. As REACT is used to create SPA, we have to prevent our page from getting resfesh. We can achieve this bye converting anchor tag to Link tag provided by react-router-dom

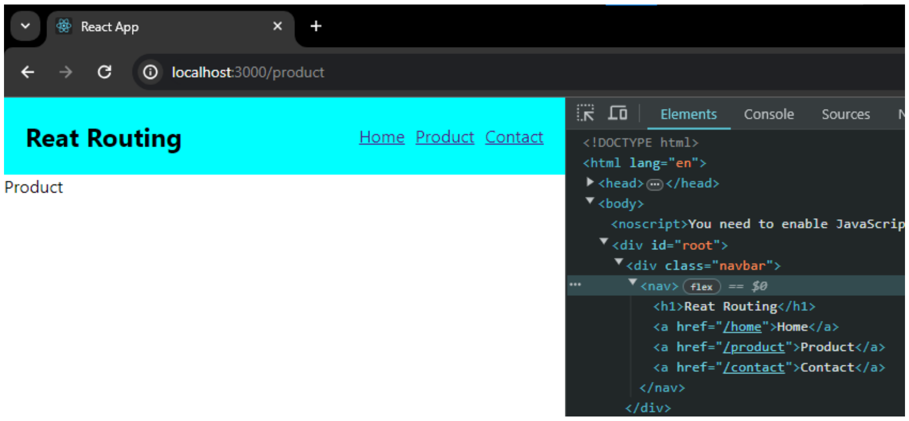


```
export default function Navbar() {
  return (
    <div className='navbar'>
      <nav>
        <h1>Reat Routing</h1>

        <a href="/home">Home</a>
        <a href="/product">Product</a>
        <a href="/contact">Contact</a>
      </nav>
    </div>
  )
}
```



```
export default function Navbar() {
  return (
    <div className='navbar'>
      <nav>
        <h1>Reat Routing</h1>

        <Link to="/home">Home</Link>
        <Link to="/product">Product</Link>
        <Link to="/contact">Contact</Link>
      </nav>
    </div>
  )
}
```

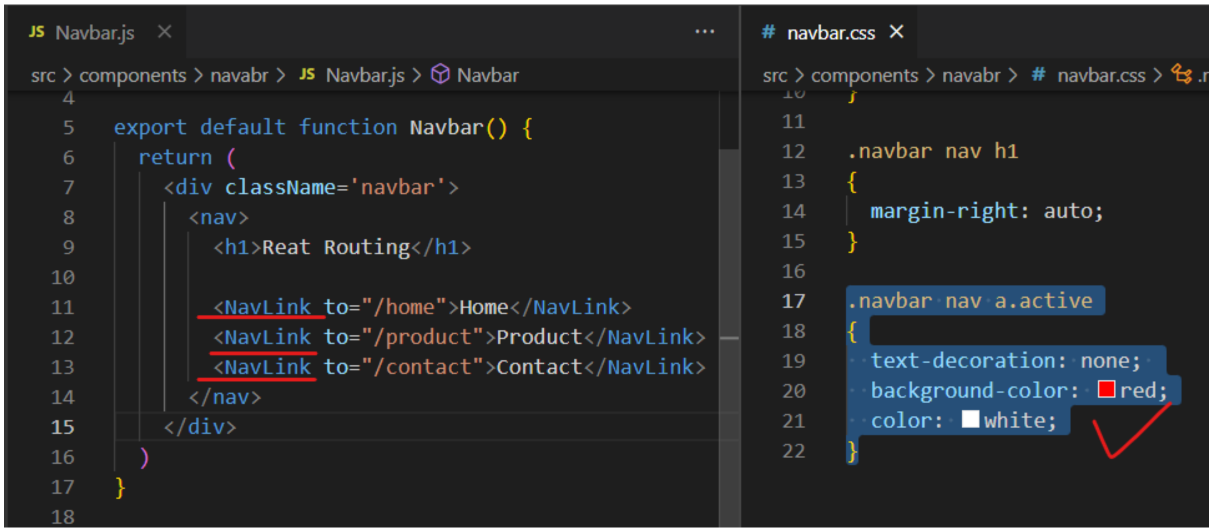
But if you inspect, you will see anchor tag only



```
Reat Routing        Home  Product  Contact
Product
```

```
<!DOCTYPE html>
<html lang="en">
▶<head>⋯</head>
▼<body>
   <noscript>You need to enable JavaScrip
  ▼<div id="root">
    ▼<div class="navbar">
      ▼<nav> flex == $0
         <h1>Reat Routing</h1>
         <a href="/home">Home</a>
         <a href="/product">Product</a>
         <a href="/contact">Contact</a>
       </nav>
     </div>
```

11. Providing CSS on navbar
    a. Convert Link to NavLink – this will automatically add active class on anchor tag (inspect and check)

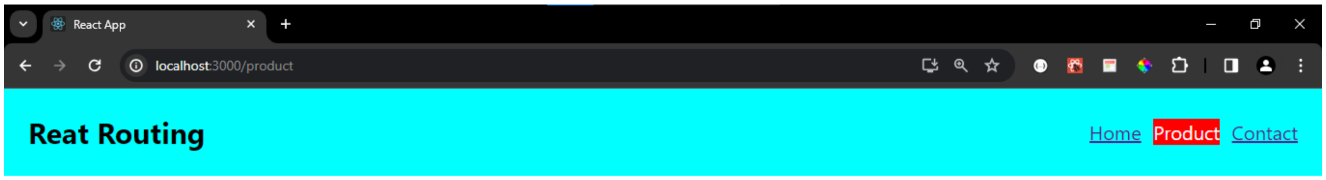


Navbar.js
src > components > navabr > Navbar.js > Navbar
navbar.css
src > components > navabr > navbar.css

```
export default function Navbar() {
  return (
    <div className='navbar'>
      <nav>
        <h1>Reat Routing</h1>

        <NavLink to="/home">Home</NavLink>
        <NavLink to="/product">Product</NavLink>
        <NavLink to="/contact">Contact</NavLink>
      </nav>
    </div>
  )
}
```

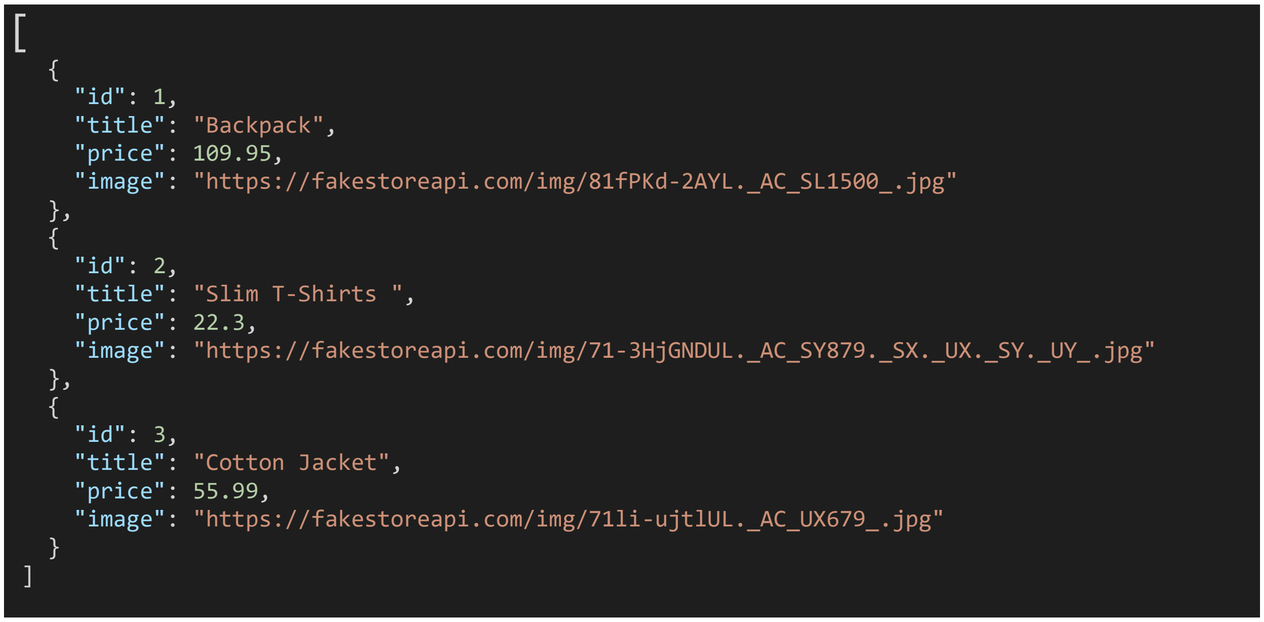
```
.navbar nav h1
{
  margin-right: auto;
}

.navbar nav a.active
{
  text-decoration: none;
  background-color: red;
  color: white;
}
```

    b. Output



Product

12. Implementing dynamic routing
    a. Create products.json inside src folder
    b. Code for products.json



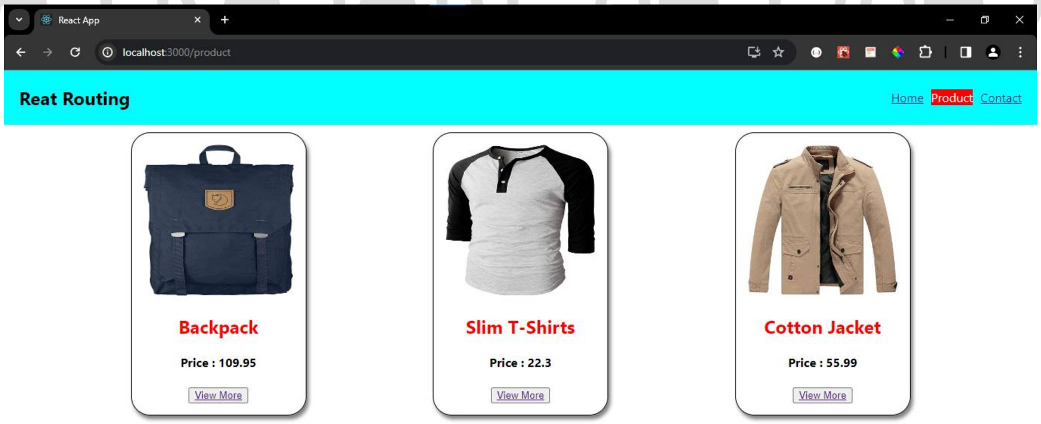
```
[
  {
    "id": 1,
    "title": "Backpack",
    "price": 109.95,
    "image": "https://fakestoreapi.com/img/81fPKd-2AYL._AC_SL1500_.jpg"
  },
  {
    "id": 2,
    "title": "Slim T-Shirts ",
    "price": 22.3,
    "image": "https://fakestoreapi.com/img/71-3HjGNDUL._AC_SY879._SX._UX._SY._UY_.jpg"
  },
  {
    "id": 3,
    "title": "Cotton Jacket",
    "price": 55.99,
    "image": "https://fakestoreapi.com/img/71li-ujtlUL._AC_UX679_.jpg"
  }
]
```

c. Fetch data from products.json into Product.js and display into a card

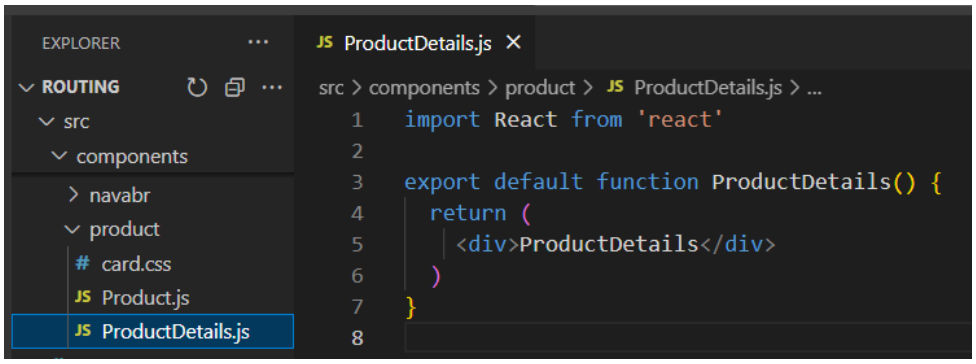
```
import React from 'react'
import products from '../../products.json'
import './card.css'
import { Link } from 'react-router-dom'

export default function Product() {
  return (
  <div className='card'>
    {
        products.map(product =>
          {
            return (
                <div className='card-items'>
                    <img src={product.image} width="200px"
height="200px"/>
                    <h2>{product.title}</h2>
                    <h4>Price : {product.price}</h4>
                    <button><Link
to={`/product/${product.id}`}>View More</Link></button>
                  </div>

              )
          })
      }
  </div>
  )
}
```

Create card.css in product folder. Code –

```
.card
{
   display: flex;
   justify-content: space-evenly;
   margin-top: 10px;
}
.card-items
{
   text-align: center;
   border: 1px solid black;
   padding: 15px;
   border-radius: 25px;
   box-shadow: 3px 5px 5px grey;
}
.card-items h2
{
   color: red;
}
```

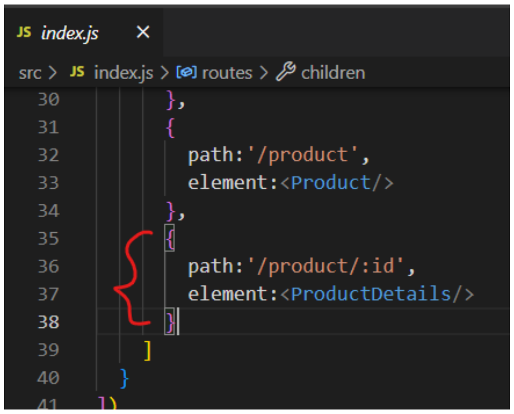


d. Create ProductDetails.js component in product folder



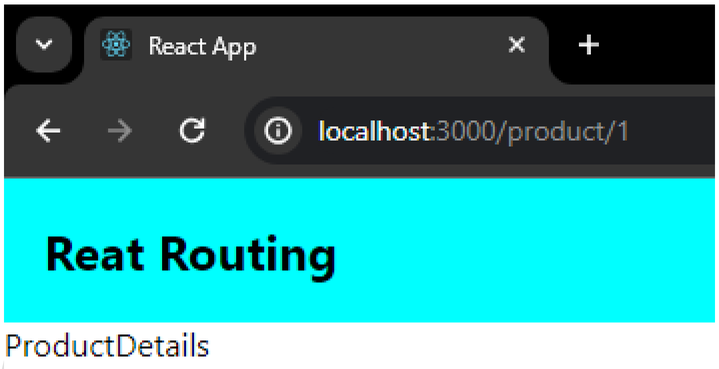
```
import React from 'react'

export default function ProductDetails() {
  return (
    <div>ProductDetails</div>
  )
}
```

e. Provide path for ProductDetails from Index.js



```
index.js
src > index.js > routes > children
        },
        {
          path:'/product',
          element:<Product/>
        },
        {
          path:'/product/:id',
          element:<ProductDetails/>
        }
      ]
    }
])
```

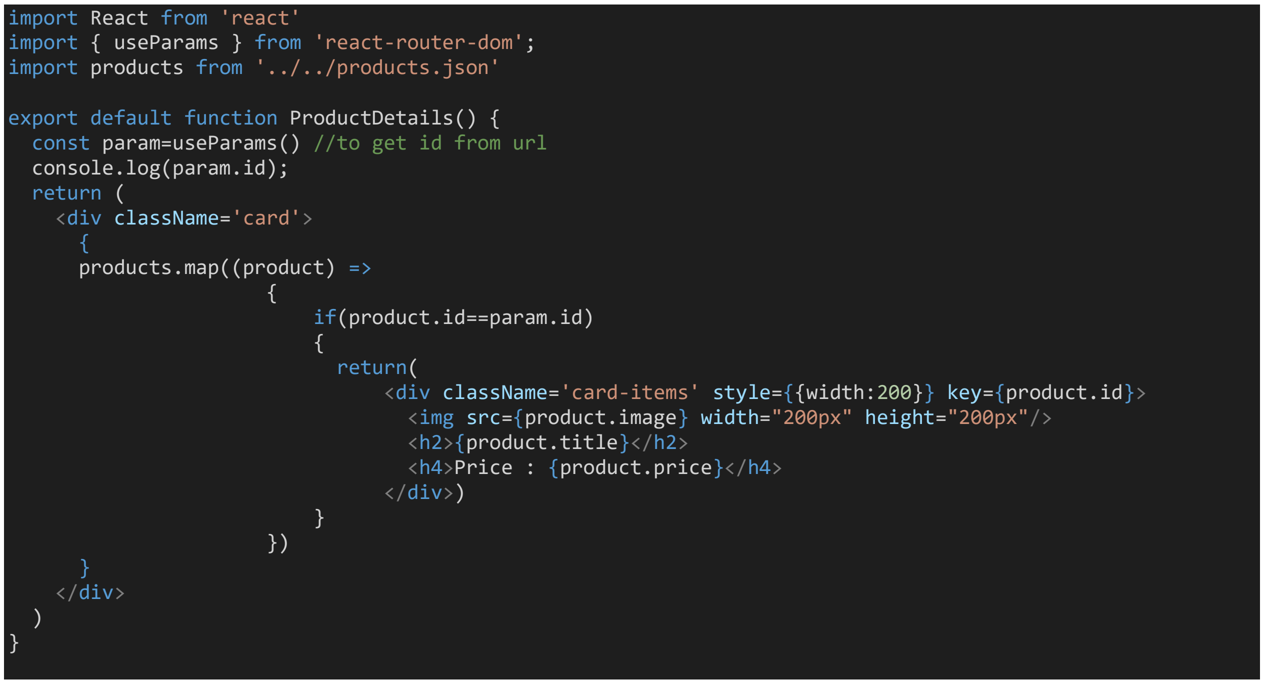
Output :



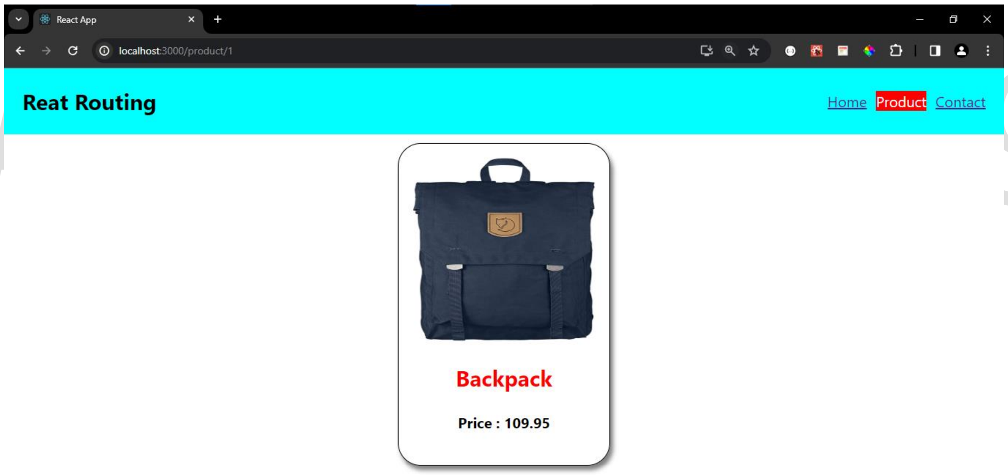
Reat Routing

ProductDetails

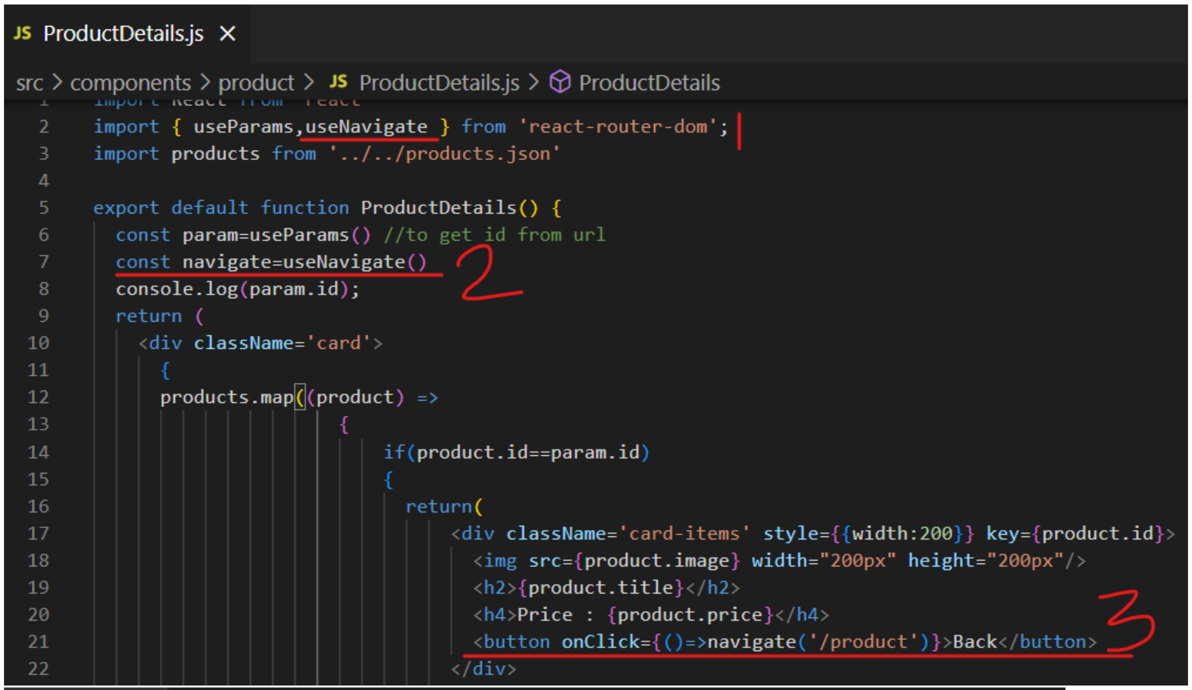
13. Displaying product in ProductDetails.js



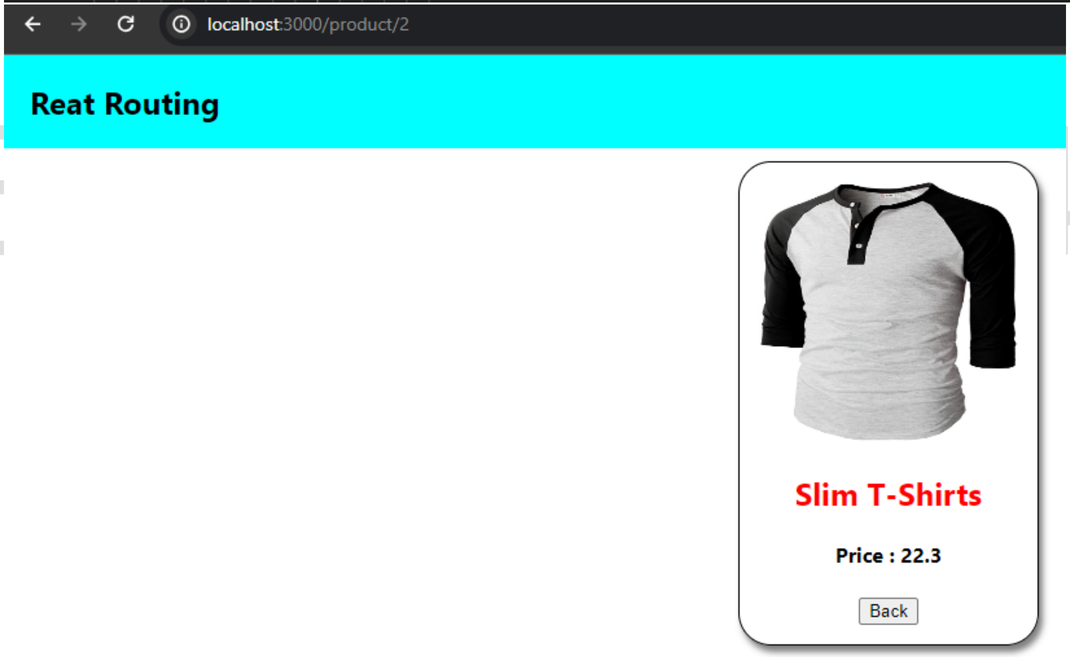
```
import React from 'react'
import { useParams } from 'react-router-dom';
import products from '../../products.json'

export default function ProductDetails() {
  const param=useParams() //to get id from url
  console.log(param.id);
  return (
    <div className='card'>
      {
      products.map((product) =>
                      {
                          if(product.id==param.id)
                          {
                            return(
                                <div className='card-items' style={{width:200}} key={product.id}>
                                  <img src={product.image} width="200px" height="200px"/>
                                  <h2>{product.title}</h2>
                                  <h4>Price : {product.price}</h4>
                                </div>)
                          }
                      })
      }
    </div>
  )
}
```



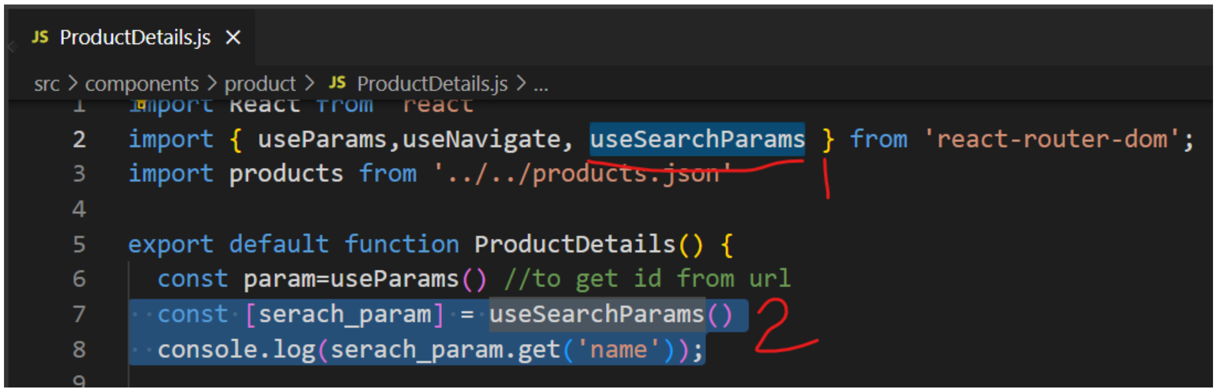
14. Back button functionality


ProductDetails.js
src > components > product > ProductDetails.js > ProductDetails
2
3

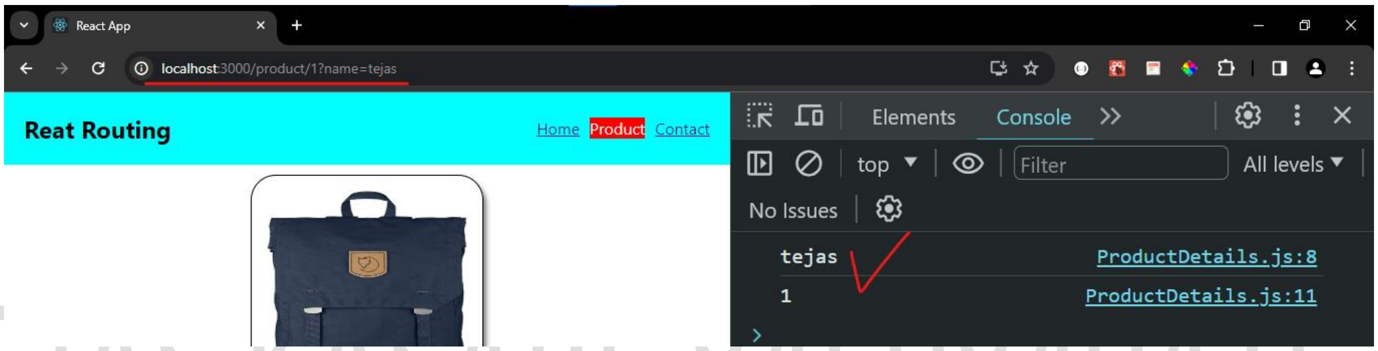
```jsx
import React from 'react'
import { useParams,useNavigate } from 'react-router-dom';
import products from '../../products.json'

export default function ProductDetails() {
  const param=useParams() //to get id from url
  const navigate=useNavigate()
  console.log(param.id);
  return (
    <div className='card'>
      {
      products.map((product) =>
                {
                  if(product.id==param.id)
                  {
                    return(
                      <div className='card-items' style={{width:200}} key={product.id}>
                        <img src={product.image} width="200px" height="200px"/>
                        <h2>{product.title}</h2>
                        <h4>Price : {product.price}</h4>
                        <button onClick={()=>navigate('/product')}>Back</button>
                      </div>
```



15. Search parameter/query parameter
    product/1 => here 1 is called as dynamic parameter => useParams()
    product?name=backpack => here name is called as Search parameter/query parameter => useSearchParams()



```
ProductDetails.js
src > components > product > ProductDetails.js > ...
import React from 'react'
import { useParams,useNavigate, useSearchParams } from 'react-router-dom';
import products from '../../products.json'

export default function ProductDetails() {
  const param=useParams() //to get id from url
  const [serach_param] = useSearchParams()
  console.log(serach_param.get('name'));
```



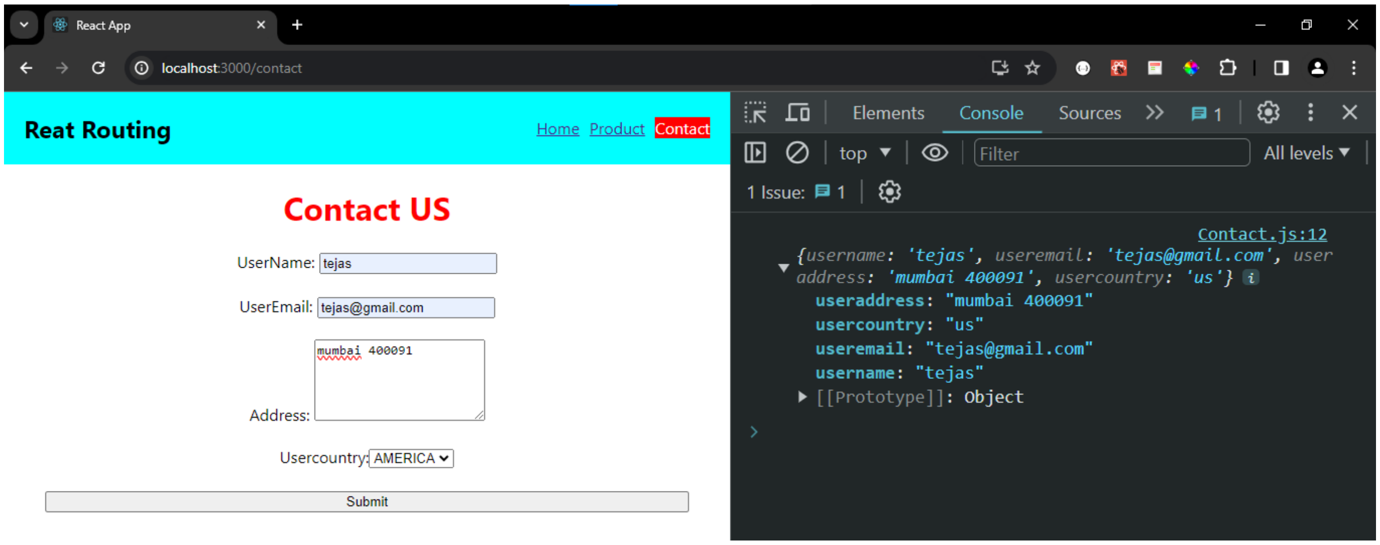
```
localhost:3000/product/1?name=tejas
```

```
tejas        ProductDetails.js:8
1            ProductDetails.js:11
```

16. Working with forms – in Contact.js

```
import React, { useState } from 'react'

export default function Contact() {
  const [contactForm,setContactForm]=useState({
    username:'',
    useremail:'',
    useraddress:'',
    usercountry:''
  })
  const handleSubmit = (event)=>{
      event.preventDefault()
      console.log(contactForm);
  }

  const handleChange = (event)=>{
    //console.log(event.target.value);
    // console.log(event.target.name);
    setContactForm(
      {
        ...contactForm,
        [event.target.name]:event.target.value
      }
    )
}
  return (
    <div>
      <h1 style={{color:"red",textAlign:"center"}}>Contact US</h1>
      <form onSubmit={handleSubmit} style={{textAlign:'center'}}>
          <label htmlFor="username">UserName: </label>
          <input type="text" id='username'
                              name='username'
                              value={contactForm.username}
                              onChange={handleChange}
                                  />
          <br/><br/>
          <label htmlFor="useremail">UserEmail: </label>
          <input type="email" id='useremail'
                              name='useremail'
                              value={contactForm.useremail}
                              onChange={handleChange}/>
          <br/><br/>
          <label htmlFor="useraddress">Address: </label>
          <textarea id="useraddress" cols="20" rows="5"
                              name='useraddress'
                              value={contactForm.useraddress}
                              onChange={handleChange}></textarea>
          <br/><br/>
          <label htmlFor="usercountry">Usercountry:</label>
          <select name="usercountry" id="usercountry"
                      value={contactForm.usercountry}
                      onChange={handleChange}>
            <option value="india">INDIA</option>
            <option value="us">AMERICA</option>
            <option value="china">CHINA</option>
          </select>
          <br/><br/>
          <input type="submit" style={{width:"100vh"}}/>
      </form>
    </div>
  )
}
```

Output :


React App
localhost:3000/contact

Reat Routing Home Product Contact

Contact US

UserName: tejas

UserEmail: tejas@gmail.com

Address: mumbai 400091

Usercountry: AMERICA

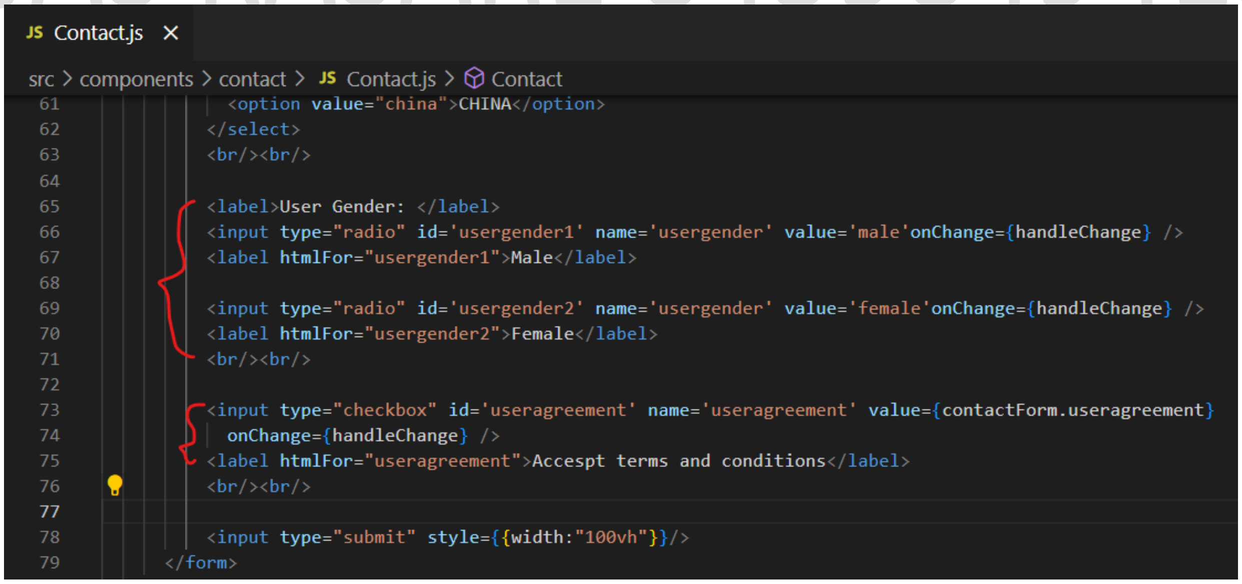
Submit

```
                                                              Contact.js:12
 ▼ {username: 'tejas', useremail: 'tejas@gmail.com', user
   address: 'mumbai 400091', usercountry: 'us'} ℹ
     useraddress: "mumbai 400091"
     usercountry: "us"
     useremail: "tejas@gmail.com"
     username: "tejas"
   ▶ [[Prototype]]: Object
>
```

17. Handling checkbox and radio buttons
    a. Create variables in useState()

```
export default function Contact() {
  const [contactForm,setContactForm]=useState({
    username:'',
    useremail:'',
    useraddress:'',
    usercountry:'',
    usergender:'',
    useragreement:false,
  })
```

    b. Create UI for radio button and check box


Contact.js

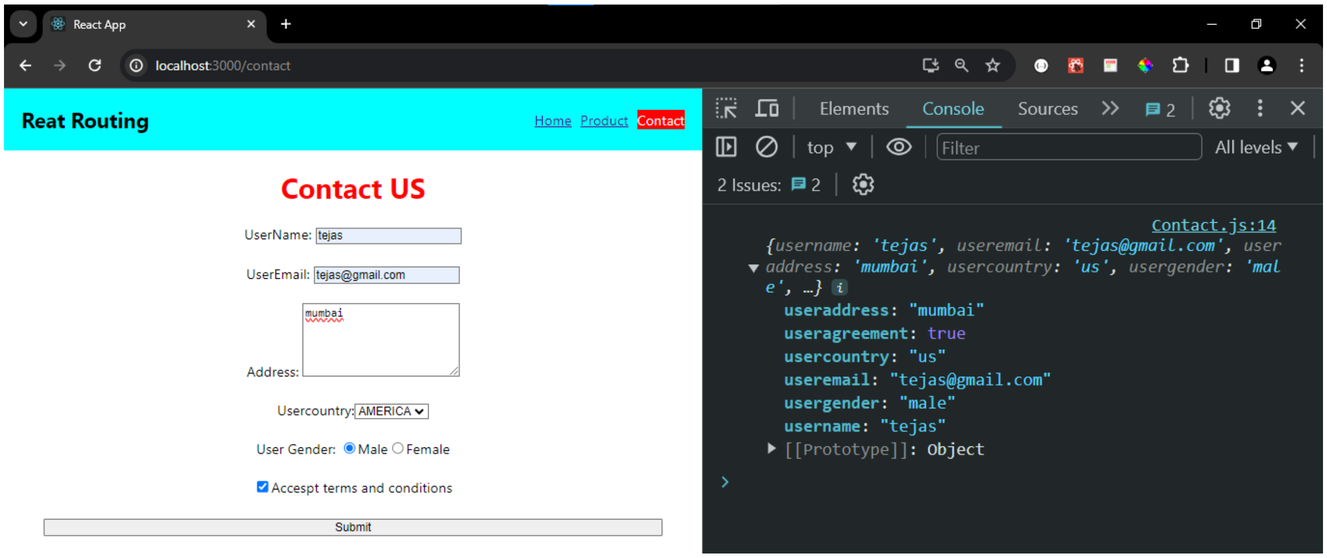
src > components > contact > Contact.js > Contact

```
                <option value="china">CHINA</option>
              </select>
              <br/><br/>

              <label>User Gender: </label>
              <input type="radio" id='usergender1' name='usergender' value='male'onChange={handleChange} />
              <label htmlFor="usergender1">Male</label>

              <input type="radio" id='usergender2' name='usergender' value='female'onChange={handleChange} />
              <label htmlFor="usergender2">Female</label>
              <br/><br/>

              <input type="checkbox" id='useragreement' name='useragreement' value={contactForm.useragreement}
                onChange={handleChange} />
              <label htmlFor="useragreement">Accespt terms and conditions</label>
              <br/><br/>

              <input type="submit" style={{width:"100vh"}}/>
            </form>
```

    c. Checking type and collecting check box input

```
setContactForm(
  {
    ...contactForm,
    // [event.target.name]:event.target.value
    [event.target.name]:event.target.type=='checkbox'?event.target.checked : event.target.value
  }
)
```

d. Output



Contact US

UserName: tejas

UserEmail: tejas@gmail.com

Address: mumbai

Usercountry: AMERICA

User Gender: [✓ Male] Female

[✓] Accept terms and conditions

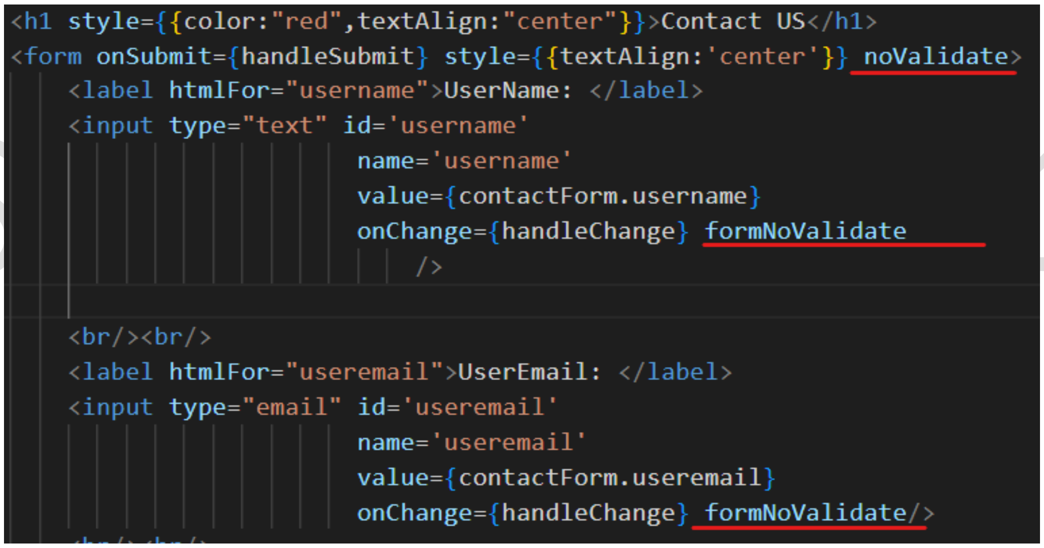
Submit

```
Contact.js:14
{username: 'tejas', useremail: 'tejas@gmail.com', user
address: 'mumbai', usercountry: 'us', usergender: 'mal
e', …}
  useraddress: "mumbai"
  useragreement: true
  usercountry: "us"
  useremail: "tejas@gmail.com"
  usergender: "male"
  username: "tejas"
▶ [[Prototype]]: Object
```

18. Handling Form Errors

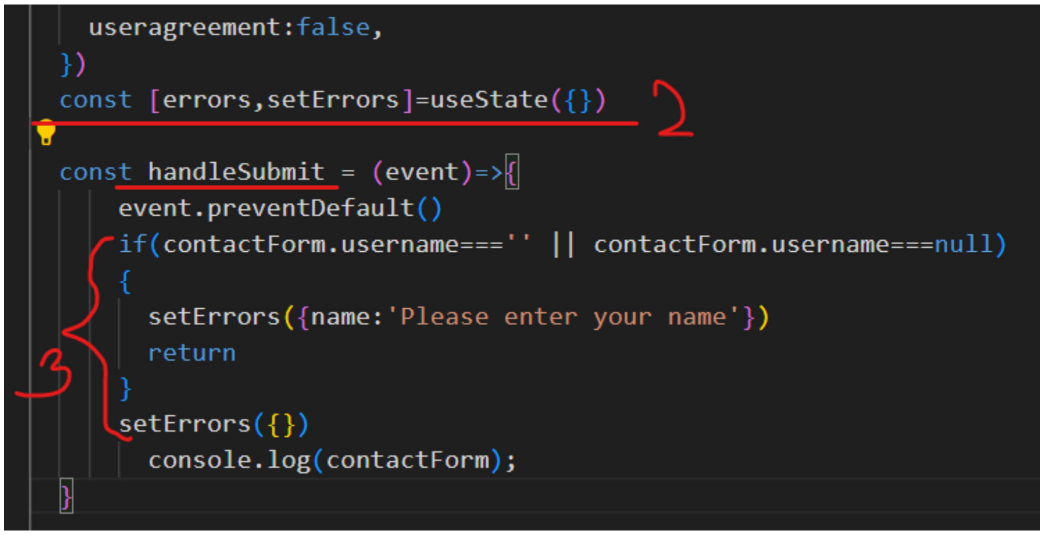
a. Handling form errors by traditional way

i. Add noValidate on form tag and formNoValidate on input tag to enable react from validation



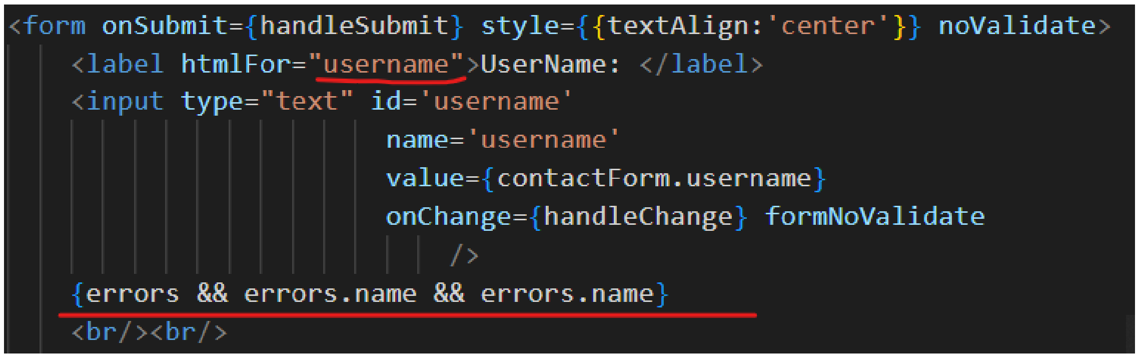
```
<h1 style={{color:"red",textAlign:"center"}}>Contact US</h1>
<form onSubmit={handleSubmit} style={{textAlign:'center'}} noValidate>
    <label htmlFor="username">UserName: </label>
    <input type="text" id='username'
                        name='username'
                        value={contactForm.username}
                        onChange={handleChange} formNoValidate
                            />

    <br/><br/>
    <label htmlFor="useremail">UserEmail: </label>
    <input type="email" id='useremail'
                        name='useremail'
                        value={contactForm.useremail}
                        onChange={handleChange} formNoValidate/>
```

ii. Manage error state



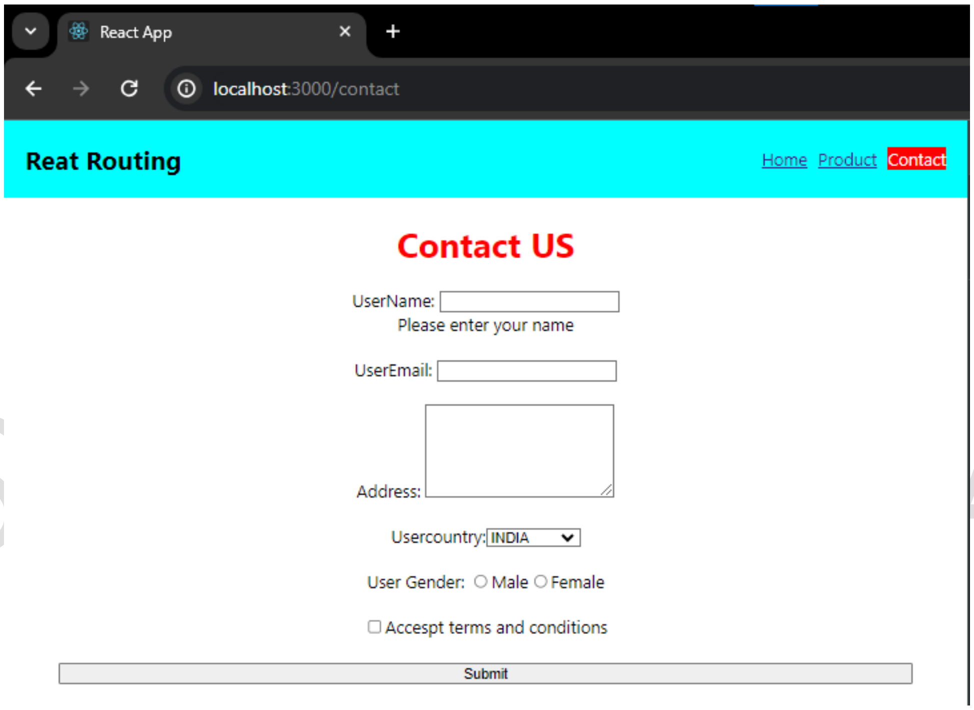
```
        useragreement:false,
    })
    const [errors,setErrors]=useState({})   2

    const handleSubmit = (event)=>{
        event.preventDefault()
        if(contactForm.username==='' || contactForm.username===null)
        {
            setErrors({name:'Please enter your name'})
            return
        }
   3    setErrors({})
            console.log(contactForm);
    }
```

iii. Display error message on respective input



```
<form onSubmit={handleSubmit} style={{textAlign:'center'}} noValidate>
    <label htmlFor="username">UserName: </label>
    <input type="text" id='username'
                    name='username'
                    value={contactForm.username}
                    onChange={handleChange} formNoValidate
                        />
    {errors && errors.name && errors.name}
    <br/><br/>
```

iv. Output : click on sumbit button without providing username



Contact US

UserName:
Please enter your name

UserEmail:

Address:

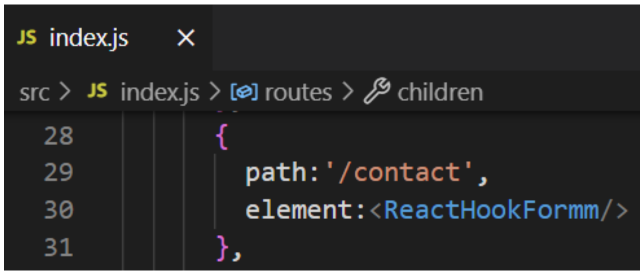
Usercountry: INDIA

User Gender: ○ Male ○ Female

☐ Accespt terms and conditions

v. DONE !!

b. Handling form errors by using external libraries : React Hook Form
   i. Install npm install react-hook-form
   ii. Create ReactHookFormm.js in components folder > contact folder
   iii. In index.js change reference of contact path to above ReactHookFormm.js



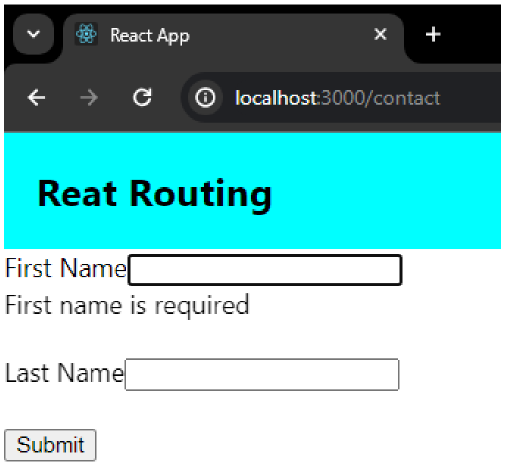
```
{
  path:'/contact',
  element:<ReactHookFormm/>
},
```

   iv. Code ReactHookFormm.js

```
import React from 'react'
import { useForm } from 'react-hook-form'

export default function ReactHookFormm() {
   const {register,handleSubmit,formState}=useForm()
   //to register out input tag use register
   //handlesubmit check the validation before submitting data
   //formState manage the errors
   const submitData =(data) =>{
      console.log(data);
   }

  return (
    <div>
      <form onSubmit={handleSubmit(submitData)}>
      <label>First Name</label>
      <input type='text' {...register("firstname",{required:true,minLength:3})} />
      <br />
      {formState.errors && formState.errors.firstname && formState.errors.firstname.type=="required" && "First name is required"}

      {formState.errors && formState.errors.firstname && formState.errors.firstname.type=="minLength" && "min 3 characters required"}

         <br /><br />
      <label>Last Name</label>
      <input type='text' {...register("lastname")} />
         <br /><br />
      <input type="submit" />
    </form>
    </div>
  )
}
```

v. DONE !!

vi. Output :


React App

localhost:3000/contact

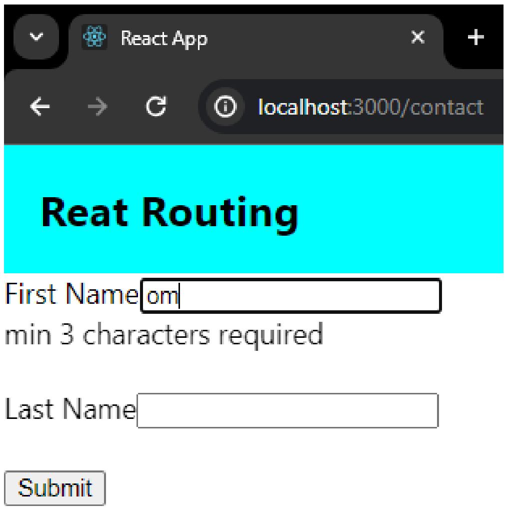
Reat Routing

First Name

First name is required

Last Name

Submit


React App

localhost:3000/contact

Reat Routing

First Name om

min 3 characters required

Last Name

Submit

19. COMPLETED !!!!! see you in context, reducer and redux