

## Module 3: Handling Navigation with Routes

---

Demo Document 4

**edureka!**

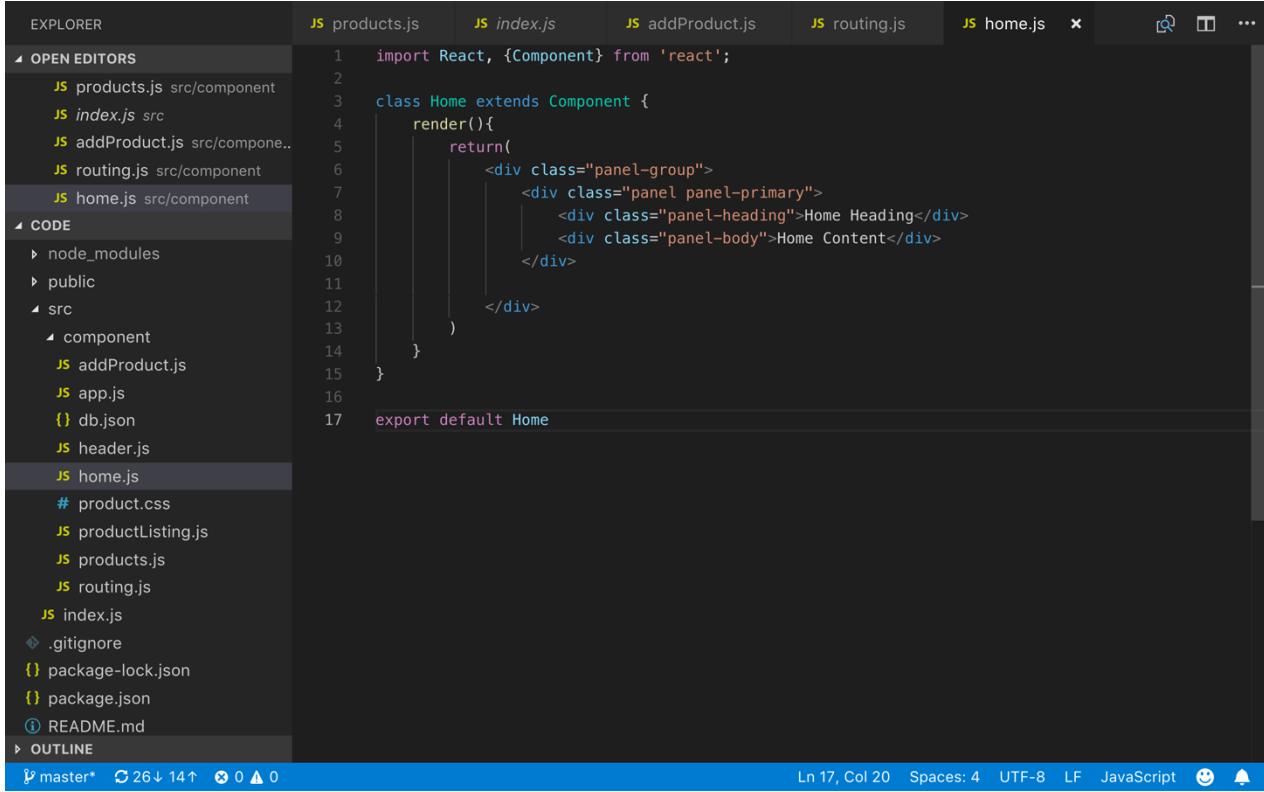
**edureka!**

© Brain4ce Education Solutions Pvt. Ltd.

## Dynamic Music Store Application with Routing and API connectivity

**Step 1:** To understand routing we need to create different components and navigate between these components. So, let us create the first component: ***Home***.

Open the component folder and create ***home.js*** as shown below.



```

EXPLORER JS products.js JS index.js JS addProduct.js JS routing.js JS home.js x ...
OPEN EDITORS JS products.js src/component JS index.js src JS addProduct.js src/component JS routing.js src/component JS home.js src/component
CODE node_modules public src component addProduct.js app.js db.json header.js home.js product.css productListing.js products.js routing.js index.js .gitignore package-lock.json package.json README.md
OUTLINE
products.js
index.js
addProduct.js
routing.js
home.js
product.css
productListing.js
products.js
routing.js
index.js
.gitignore
package-lock.json
package.json
README.md
Ln 17, Col 20 Spaces: 4 UTF-8 LF JavaScript 😊 📲

```

```

1 import React, {Component} from 'react';
2
3 class Home extends Component {
4     render(){
5         return(
6             <div class="panel-group">
7                 <div class="panel panel-primary">
8                     <div class="panel-heading">Home Heading</div>
9                     <div class="panel-body">Home Content</div>
10                </div>
11            )
12        }
13    }
14}
15
16
17 export default Home

```

**Step 2:** Create a folder: **routing.js**.

To implement routing install, **react-router-dom** using “**npm i react-router-dom**” and import **BrowserRouter** from **react-router-dom**.

The screenshot shows the VS Code interface with the routing.js file open in the editor. The code implements a Router component using the BrowserRouter and Route components from react-router-dom. It defines three routes: one exact path '/' component Home, one exact path '/product' component Products, and one path '/addProduct' component AddProduct.

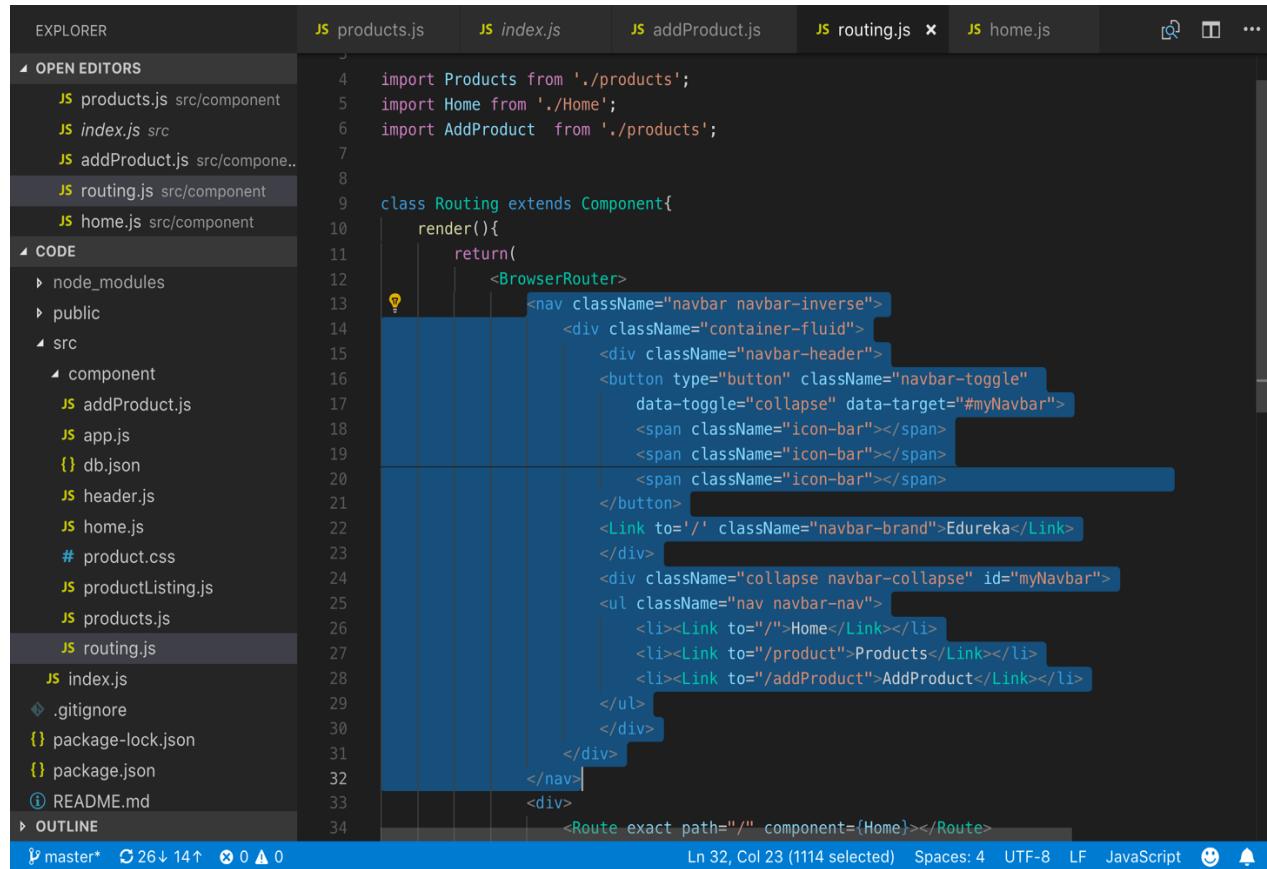
```
import React, {Component} from 'react';
import { BrowserRouter, Route, Link } from 'react-router-dom';

import Products from './products';
import Home from './Home';
import AddProduct from './products';

class Routing extends Component{
    render(){
        return(
            <BrowserRouter>
                <div>
                    <Route exact path="/" component={Home}></Route>
                    <Route exact path="/product" component={Products}></Route>
                    <Route path="/addProduct" component={AddProduct}></Route>
                </div>
            </BrowserRouter>
        )
    }
}

export default Routing;
```

**Step 3:** Now we need to use bootstrap navbar, to implement the routes we **will not use <a> tag**, but we will use **<link> tag** and map the path to the link.



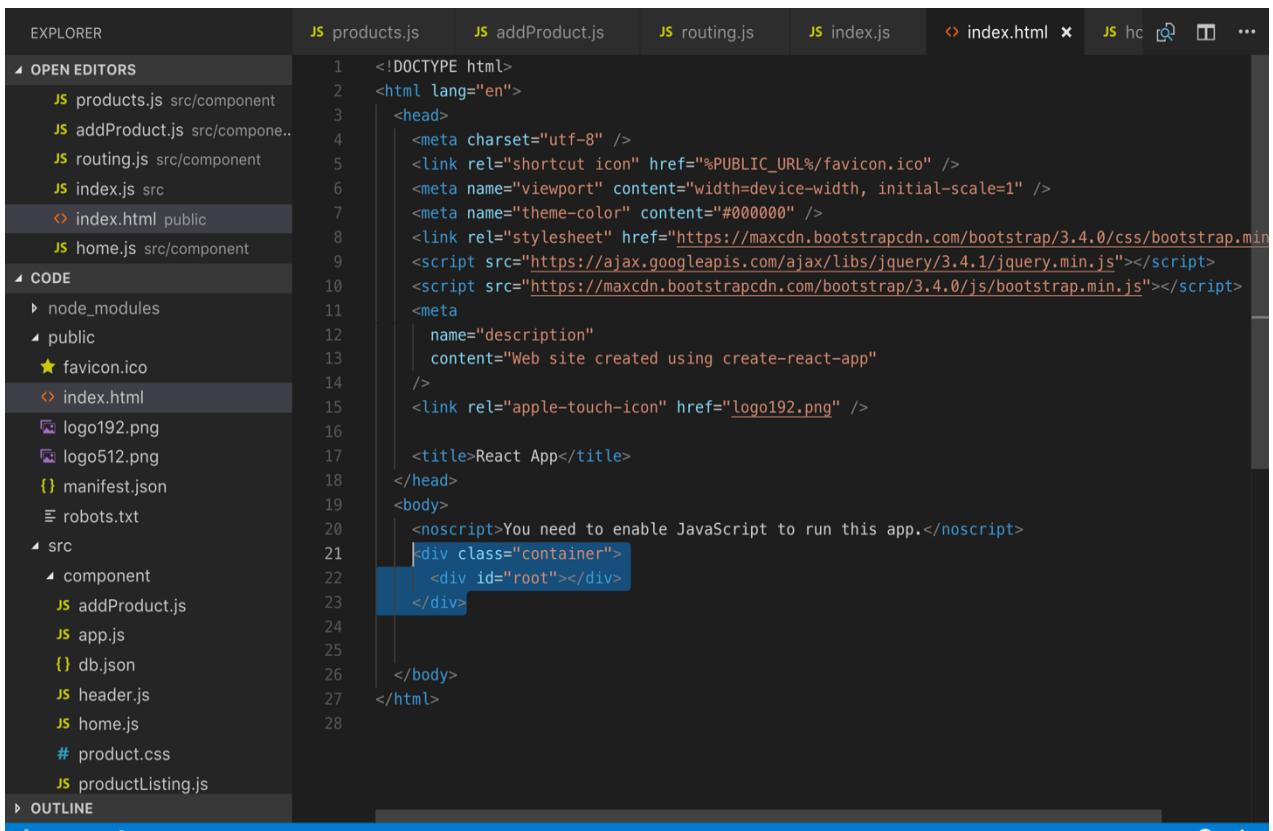
```

EXPLORER JS products.js JS index.js JS addProduct.js JS routing.js x JS home.js ...
OPEN EDITORS
  JS products.js src/component
  JS index.js src
  JS addProduct.js src/component..
  JS routing.js src/component
  JS home.js src/component
CODE
  node_modules
  public
  src
    component
      JS addProduct.js
      JS app.js
      db.json
      JS header.js
      JS home.js
      # product.css
      JS productListing.js
      JS products.js
      JS routing.js
      JS index.js
    .gitignore
    package-lock.json
    package.json
  README.md
OUTLINE
products.js
index.js
addProduct.js
routing.js
home.js
  4 import Products from './products';
  5 import Home from './Home';
  6 import AddProduct from './products';
  7
  8
  9 class Routing extends Component{
 10   render(){
 11     return(
 12       <BrowserRouter>
 13         <nav className="navbar navbar-inverse">
 14           <div className="container-fluid">
 15             <div className="navbar-header">
 16               <button type="button" className="navbar-toggle" data-toggle="collapse" data-target="#myNavbar">
 17                 <span className="icon-bar"></span>
 18                 <span className="icon-bar"></span>
 19                 <span className="icon-bar"></span>
 20               </button>
 21               <Link to="/" className="navbar-brand">Edureka</Link>
 22             </div>
 23             <div className="collapse navbar-collapse" id="myNavbar">
 24               <ul className="nav navbar-nav">
 25                 <li><Link to="/">Home</Link></li>
 26                 <li><Link to="/product">Products</Link></li>
 27                 <li><Link to="/addProduct">AddProduct</Link></li>
 28               </ul>
 29             </div>
 30           </div>
 31         <Route exact path="/" component={Home}></Route>
 32       </nav>
 33     )
 34   }

```

Ln 32, Col 23 (1114 selected) Spaces: 4 UTF-8 LF JavaScript 😊 🔔

**Step 4:** In the **index.html** we have to wrap the **root div** with container class of bootstrap to make it responsive.

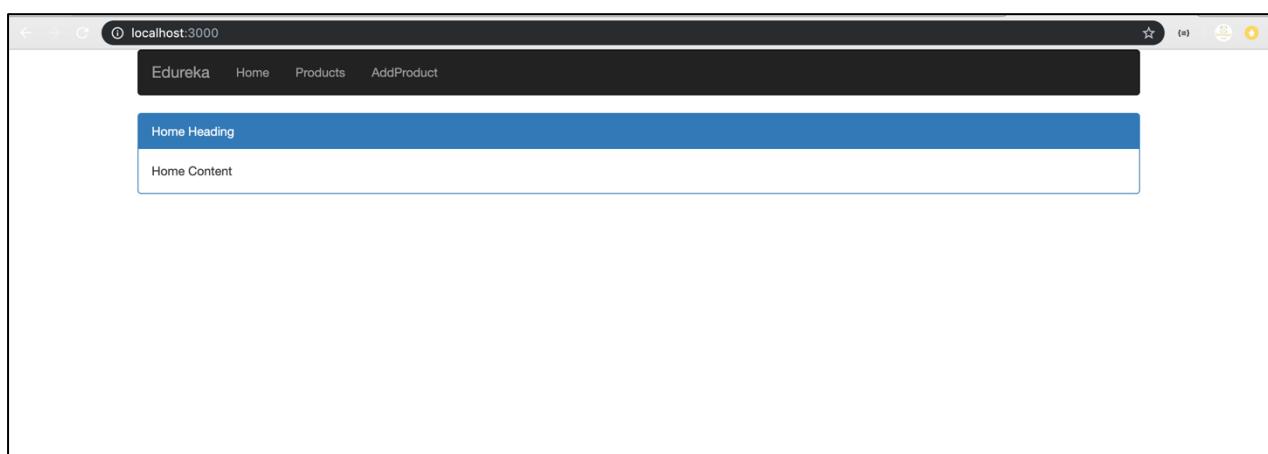


```

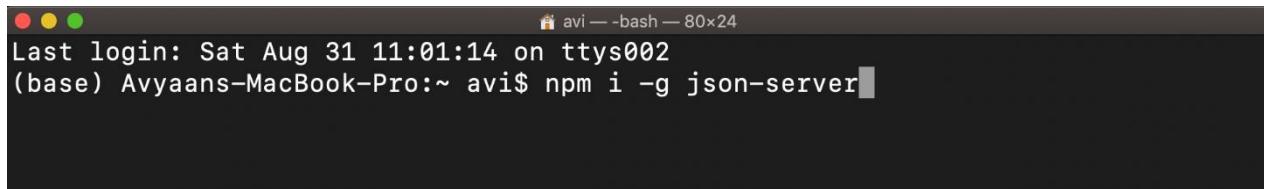
EXPLORER JS products.js JS addProduct.js JS routing.js JS index.js ⌂ index.html x JS hc ⌂ ...
OPEN EDITORS JS products.js src/component JS addProduct.js src/component...
CODE JS routing.js src/component JS index.js src ⌂ index.html public JS home.js src/component...
node_modules
public
★ favicon.ico
index.html
logo192.png
logo512.png
manifest.json
robots.txt
src
component
JS addProduct.js
JS app.js
{} db.json
JS header.js
JS home.js
# product.css
JS productListing.js
OUTLINE
Ln 21, Col 5 (62 selected) Spaces: 2 UTF-8 LF HTML 😊 🚶
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8" />
5     <link rel="shortcut icon" href="%PUBLIC_URL%/favicon.ico" />
6     <meta name="viewport" content="width=device-width, initial-scale=1" />
7     <meta name="theme-color" content="#000000" />
8     <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css" />
9     <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
10    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>
11    <meta
12      name="description"
13      content="Web site created using create-react-app"
14    />
15    <link rel="apple-touch-icon" href="logo192.png" />
16
17    <title>React App</title>
18  </head>
19  <body>
20    <noscript>You need to enable JavaScript to run this app.</noscript>
21    <div class="container">
22      <div id="root"></div>
23    </div>
24
25  </body>
26</html>
27
28

```

**Step 5:** If you check the browser, you will find out a new navbar. when you click on any menu item, you can see the page content changing in SPA manner without refreshing the page.

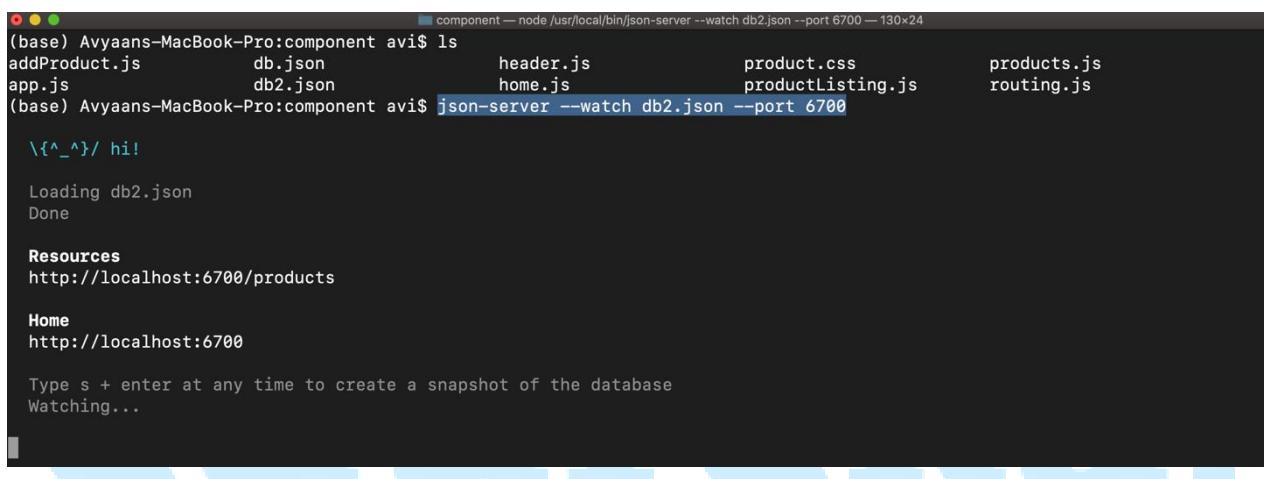


**Step 6:** For creating static API we have a package: **json-server**. Install json-server as a global package.



```
avi ~ -bash — 80x24
Last login: Sat Aug 31 11:01:14 on ttys002
(base) Avyaans-MacBook-Pro:~ avi$ npm i -g json-server
```

**Step 7:** Run the json-server with **db.json** file and provide a port number. By this you will be creating an API.



```
component — node /usr/local/bin/json-server --watch db2.json --port 6700 — 130x24
(base) Avyaans-MacBook-Pro:component avi$ ls
addProduct.js      db.json          header.js        product.css    products.js
app.js             db2.json         home.js         productListing.js  routing.js
(base) Avyaans-MacBook-Pro:component avi$ json-server --watch db2.json --port 6700

\{^_^\}/ hi!

Loading db2.json
Done

Resources
http://localhost:6700/products

Home
http://localhost:6700

Type s + enter at any time to create a snapshot of the database
Watching...
```

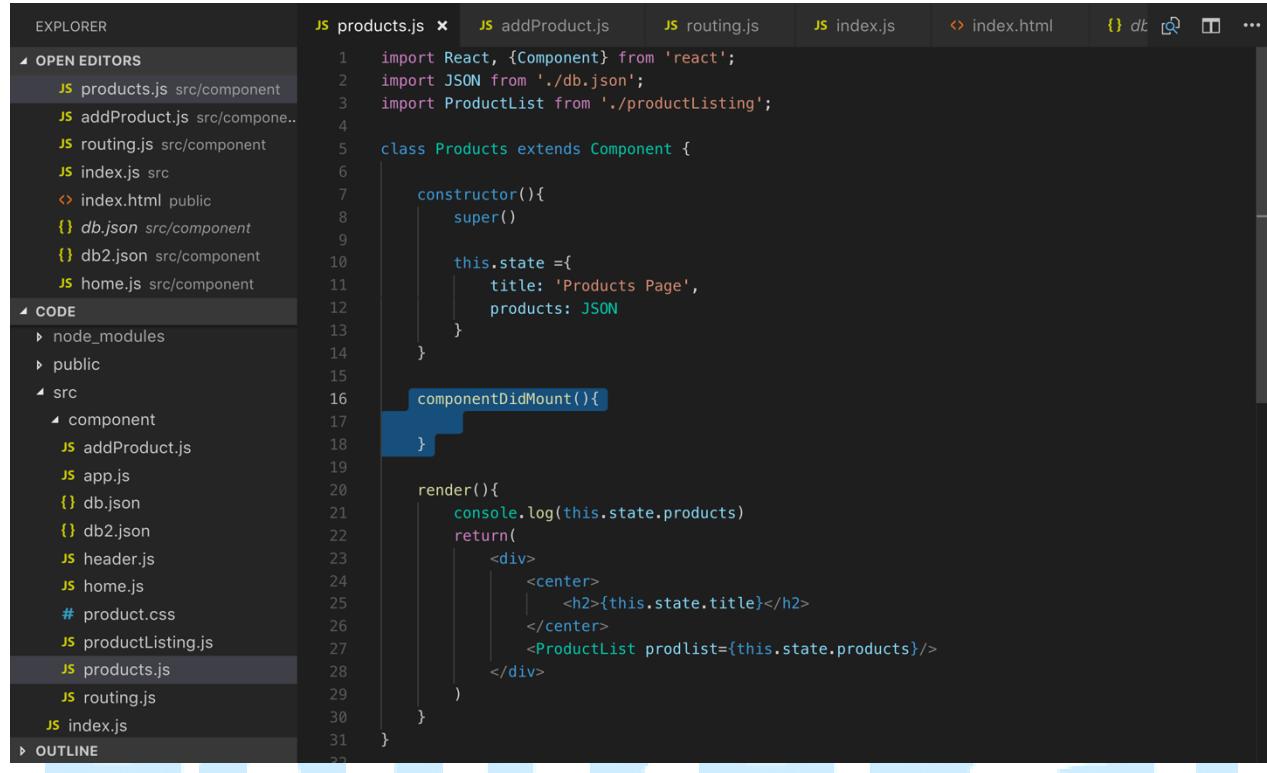
**Step 8:** Enter the endpoint in browser, you will receive json data present at the API.



```
// 20190831110704
// http://localhost:6700/products

[
  {
    "id": 1,
    "name": "Grand Piano",
    "price": 44500,
    "type": "manual",
    "description": "This is a larger baby-grand piano with fuller sound due to its bigger size and longer strings",
    "img": "https://i.ibb.co/wc6qzwW/piano.png"
  },
  {
    "id": 2,
    "name": "Electric Guitar",
    "price": 11337,
    "type": "Electric",
    "description": "The Bullet Strat With Tremolo HSS is a simple, affordable and practical guitar designed for beginning players and students.",
    "img": "https://i.ibb.co/JsbJrBB/electricguitar.png"
  },
  {
    "id": 3,
    "name": "Acoustic Drum Set",
    "price": 22599,
    "type": "Acoustic",
    "description": "Set the stage on fire with this Acoustic Drum set from the house of Havana. It is an excellent drum kit delivering great specifications",
    "img": "https://i.ibb.co/9gTfdfX/drum.png"
  },
  {
    "id": 4,
```

**Step 9:** To call API, make use of lifecycle component: ***componentDidMount()***, which will be executed after render part.



```
EXPLORER JS products.js x JS addProduct.js JS routing.js JS index.js < index.html { dk ⌂ ...  
OPEN EDITORS JS products.js src/component JS addProduct.js src/compon.. JS routing.js src/component JS index.js src < index.html public JS db.json src/component JS db2.json src/component JS home.js src/component  
CODE node_modules public src component JS addProduct.js JS app.js JS db.json JS db2.json JS header.js JS home.js # product.css JS productListing.js JS products.js JS routing.js JS index.js OUTLINE  
1 import React, {Component} from 'react';  
2 import JSON from './db.json';  
3 import ProductList from './productListing';  
4  
5 class Products extends Component {  
6  
7     constructor(){  
8         super()  
9  
10        this.state ={  
11            title: 'Products Page',  
12            products: JSON  
13        }  
14    }  
15  
16    componentDidMount(){  
17    }  
18  
19    render(){  
20        console.log(this.state.products)  
21        return(  
22            <div>  
23                <center>  
24                    <h2>{this.state.title}</h2>  
25                </center>  
26                <ProductList prolist={this.state.products}/>  
27            </div>  
28        )  
29    }  
30}  
31 }  
32  
33
```

**Step 10:** Make a call to the API using ***fetch()***.

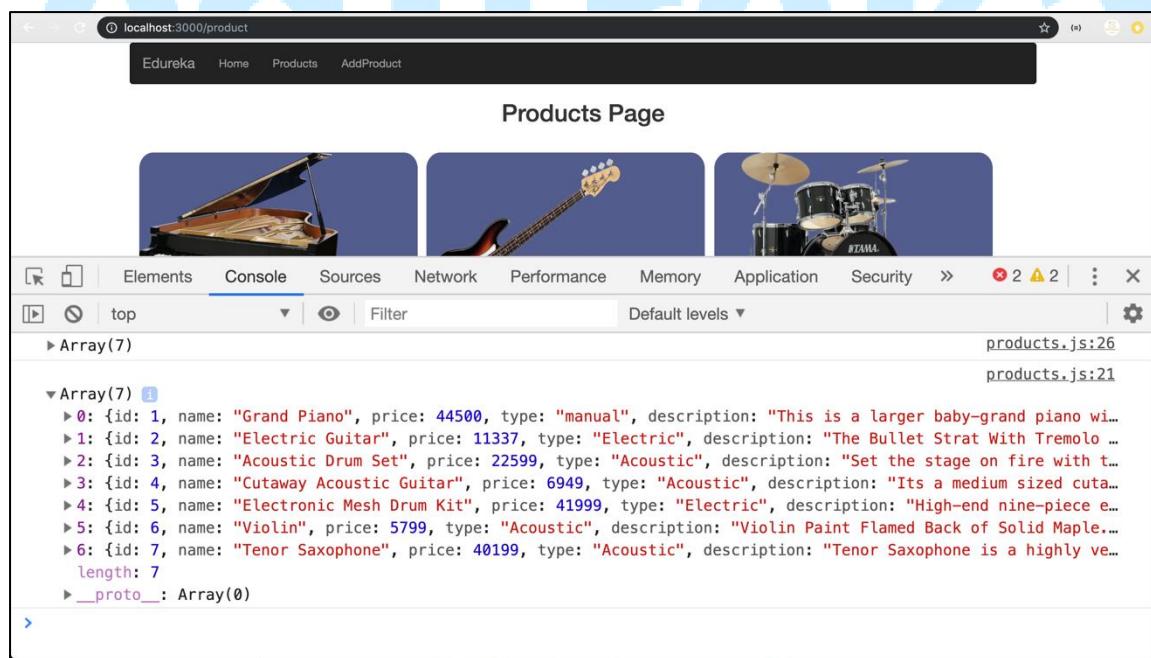
Using `fetch()` method we resolve a promise to get the data and the data is displayed in the `console.log()`.

## Module 3: Handling Navigation with Routes

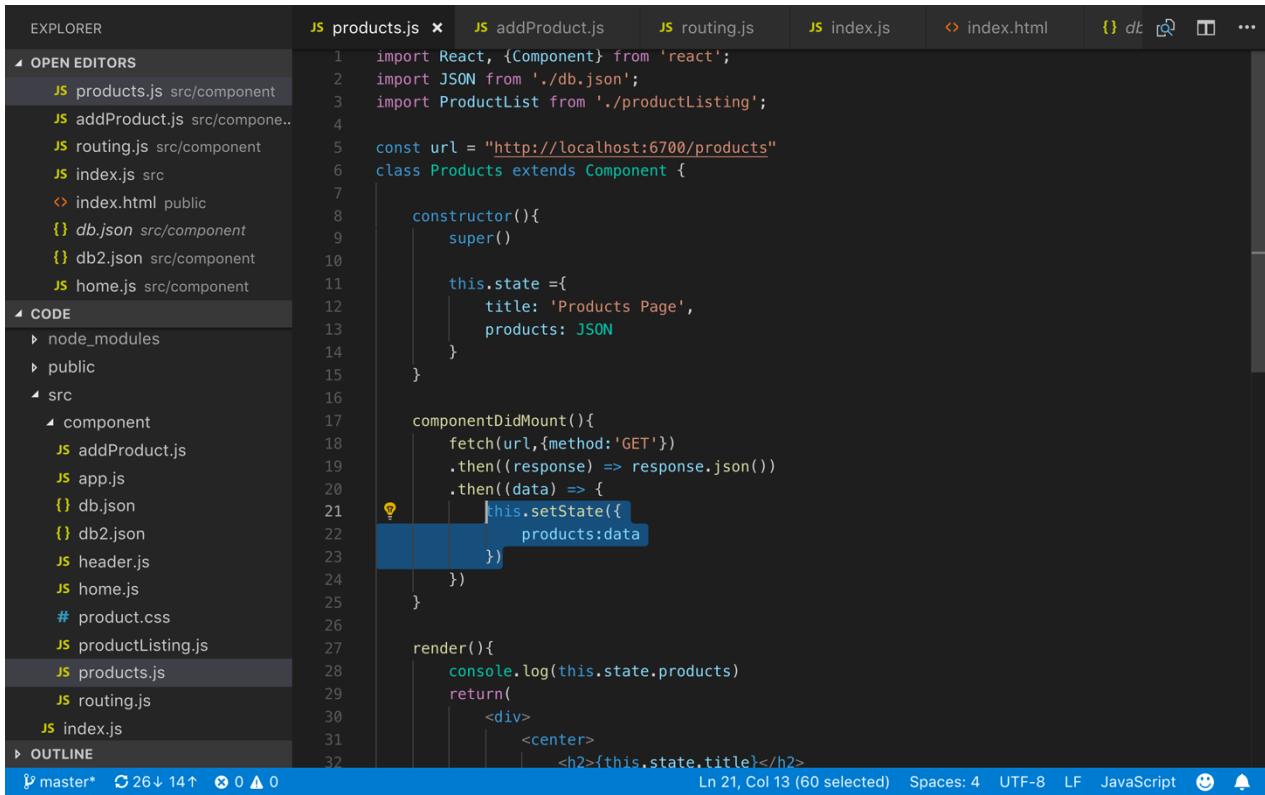
The screenshot shows the VS Code interface with the 'products.js' file open in the editor. The code is a class component that fetches data from a local API and logs it to the console. The 'CODE' sidebar on the left shows the project structure with files like 'addProduct.js', 'index.js', and 'routing.js'. The status bar at the bottom indicates the file is 'master\*' and shows other details like line count and encoding.

```
1 import React, {Component} from 'react';
2 import JSON from './db.json';
3 import ProductList from './productListing';
4
5 const url = "http://localhost:6700/products"
6 class Products extends Component {
7
8     constructor(){
9         super()
10
11         this.state ={
12             title: 'Products Page',
13             products: JSON
14         }
15     }
16
17     componentDidMount(){
18         fetch(url,{method:'GET'})
19             .then((response) => response.json())
20             .then((data) => {
21                 console.log(data)
22             })
23     }
24
25     render(){
26         console.log(this.state.products)
27         return(
28             <div>
29                 <center>
30                     <h2>{this.state.title}</h2>
31                 </center>
32                 <ProductList prodlist={this.state.products}/>
33         )
34     }
35 }
36
37 export default Products;
```

**Step 11:** Check the console, you can see the array of data returned from the API (as output).



**Step 12:** Now save the received data to a state using ***this.setState***.

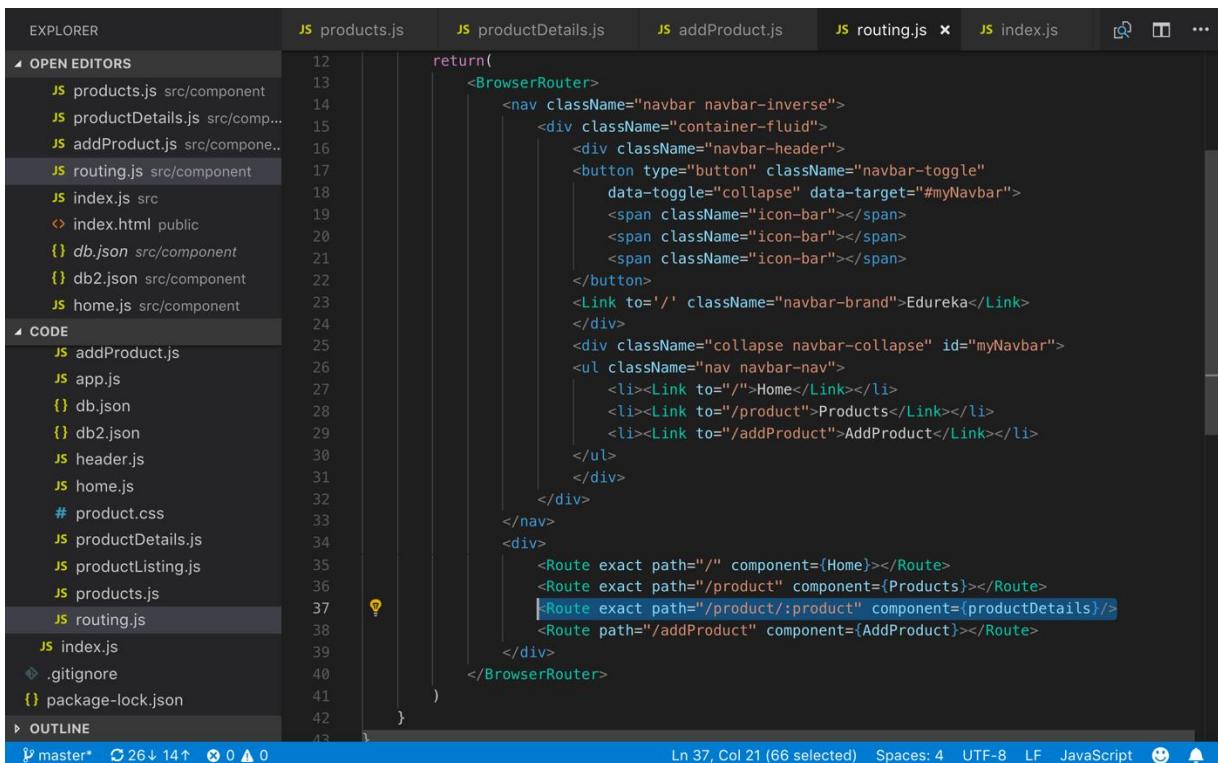


```

EXPLORER JS products.js x JS addProduct.js JS routing.js JS index.js < index.html ⌂ dt ⌂ ...
OPEN EDITORS
  JS products.js src/component
  JS addProduct.js src/component
  JS routing.js src/component
  JS index.js src
  < index.html public
  {} db.json src/component
  {} db2.json src/component
  JS home.js src/component
CODE
  node_modules
  public
  src
    component
      JS addProduct.js
      JS app.js
      {} db.json
      {} db2.json
      JS header.js
      JS home.js
      # product.css
      JS productListing.js
      JS products.js
      JS routing.js
      JS index.js
OUTLINE
  master* 26↓ 14↑ 0 ▲ 0
JS products.js
1 import React, {Component} from 'react';
2 import JSON from './db.json';
3 import ProductList from './productListing';
4
5 const url = "http://localhost:6700/products"
6 class Products extends Component {
7
8   constructor(){
9     super()
10
11     this.state ={
12       title: 'Products Page',
13       products: JSON
14     }
15   }
16
17   componentDidMount(){
18     fetch(url,{method:'GET'})
19     .then((response) => response.json())
20     .then((data) => {
21       this.setState({
22         products: data
23       })
24     })
25   }
26
27   render(){
28     console.log(this.state.products)
29     return(
30       <div>
31         <center>
32           <h2>{this.state.title}</h2>
33         </center>
34       </div>
35     )
36   }
37 }
38
39
40
41
42
43
  
```

Ln 21, Col 13 (60 selected) Spaces: 4 UTF-8 LF JavaScript

**Step 13:** To create a details page you have to create a **route**, here we will pass the param as id which will later act as a unique parameter.

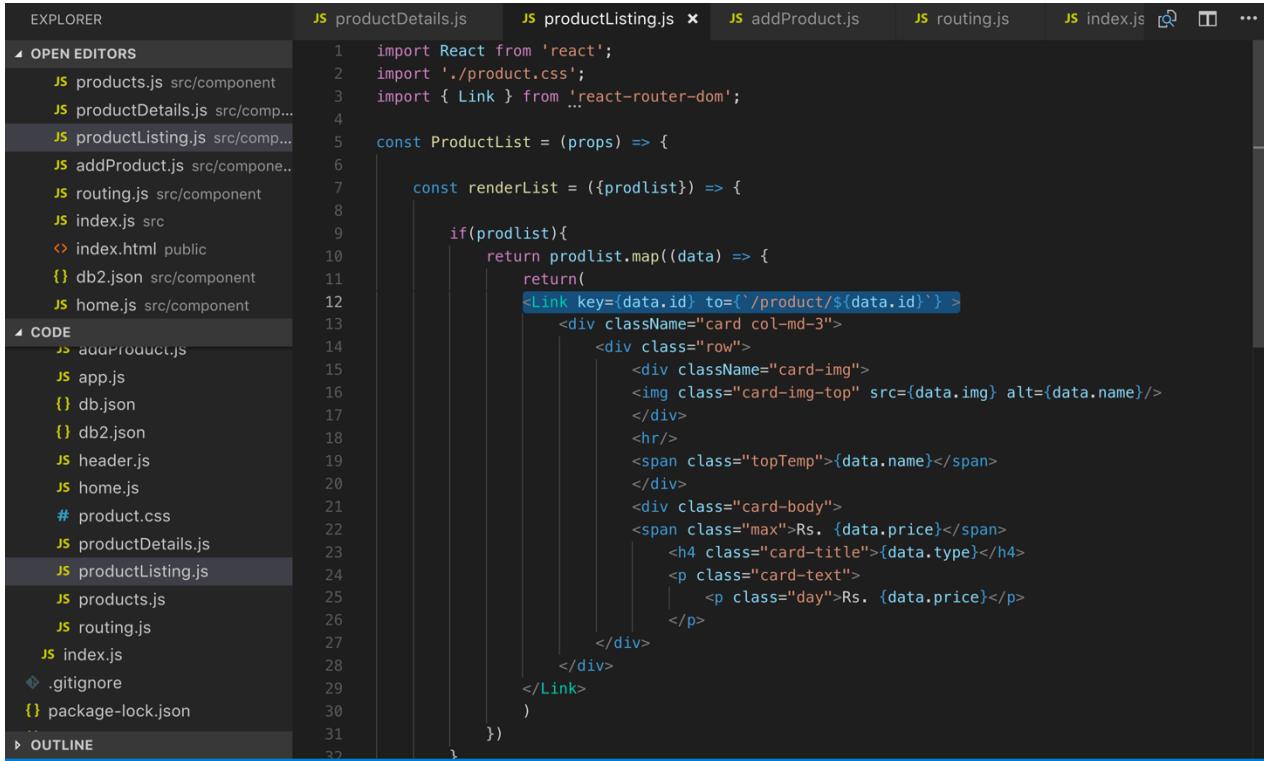


```

EXPLORER JS products.js JS productDetails.js JS addProduct.js JS routing.js x JS index.js ⌂ dt ⌂ ...
OPEN EDITORS
  JS products.js src/component
  JS productDetails.js src/component...
  JS addProduct.js src/component...
  JS routing.js src/component...
  JS index.js src
  < index.html public
  {} db.json src/component
  {} db2.json src/component
  JS home.js src/component
CODE
  addProduct.js
  JS app.js
  {} db.json
  {} db2.json
  JS header.js
  JS home.js
  # product.css
  JS productListing.js
  JS products.js
  JS productDetails.js
  JS routing.js
  JS index.js
  .gitignore
  {} package-lock.json
OUTLINE
  master* 26↓ 14↑ 0 ▲ 0
JS products.js
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
  
```

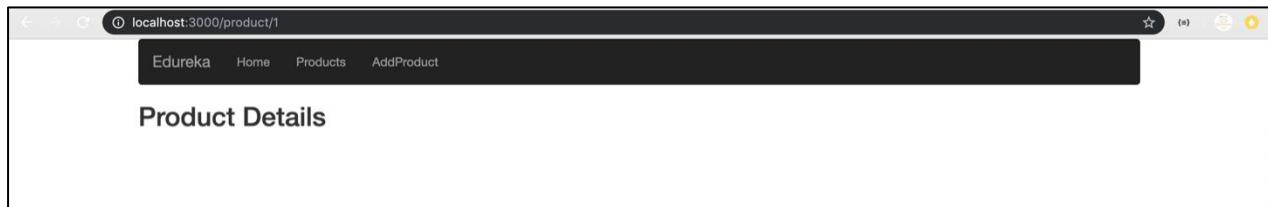
Ln 37, Col 21 (66 selected) Spaces: 4 UTF-8 LF JavaScript

**Step 14:** Wrap all the element in the **link tag** and pass param as a unique id.

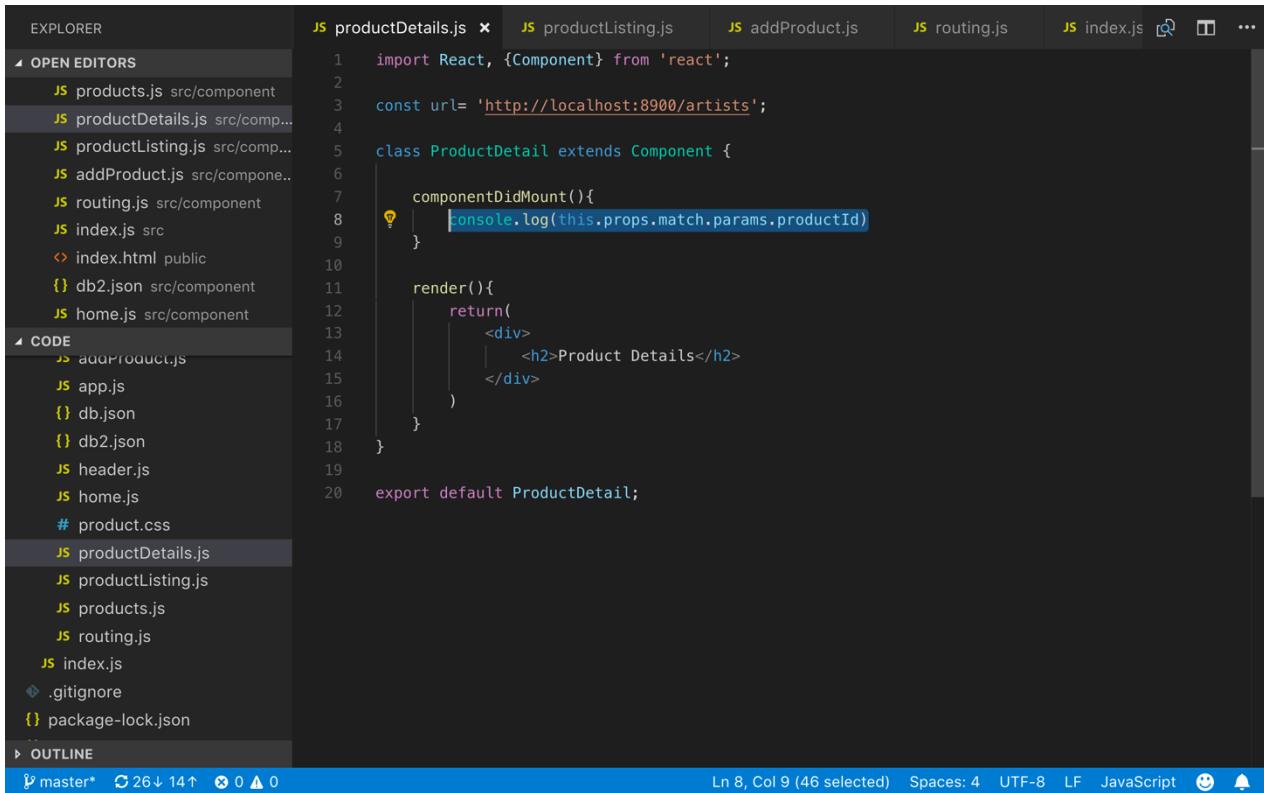


```
EXPLORER JS productDetails.js JS productListing.js x JS addProduct.js JS routing.js JS index.js 🔍 ...  
OPEN EDITORS JS products.js src/component JS productDetails.js src/comp... JS productListing.js src/comp... JS addProduct.js src/compone... JS routing.js src/component JS index.js src index.html public {} db2.json src/component JS home.js src/component  
CODE JS addProduct.js JS app.js {} db.json {} db2.json JS header.js JS home.js # product.css JS productDetails.js JS productListing.js JS products.js JS routing.js JS index.js .gitignore {} package-lock.json  
OUTLINE  
productListing.js  
1 import React from 'react';
2 import './product.css';
3 import { Link } from 'react-router-dom';
4
5 const ProductList = (props) => {
6
7     const renderList = ({prolist}) => {
8
9         if(prolist){
10             return prolist.map((data) => {
11                 return(
12                     <Link key={data.id} to={`/product/${data.id}`}>
13                         <div className="card col-md-3">
14                             <div className="row">
15                                 <div className="card-img">
16                                     <img className="card-img-top" src={data.img} alt={data.name}/>
17                                 </div>
18                                 <hr/>
19                                 <span className="topTemp">{data.name}</span>
20                                 </div>
21                                 <div className="card-body">
22                                     <span className="max">Rs. {data.price}</span>
23                                         <h4 className="card-title">{data.type}</h4>
24                                         <p className="card-text">
25                                             <p className="day">Rs. {data.price}</p>
26                                         </p>
27                                     </div>
28                                 </div>
29                             </div>
30                         )
31                     </Link>
32                 )
33             })
34         }
35     }
36 }
```

**Step 15:** Now on clicking any product user will be navigated to the details page of the respective product. Here you can see the params in the URL.



**Step 16:** This params value can be received via props, we will pass it to the API to get the data of respective item.



```

EXPLORER
OPEN EDITORS
productDetails.js x productListing.js addProduct.js routing.js index.js ...
CODE
products.js src/component productDetails.js src/com...
productListing.js src/com...
addProduct.js src/compon...
routing.js src/component
index.js src
index.html public
db2.json src/component
home.js src/component
product.css
productDetails.js
productListing.js
products.js
routing.js
index.js
.gitignore
package-lock.json ...
OUTLINE
master* 26 14 0 0 ▲ 0
Ln 8, Col 9 (46 selected) Spaces: 4 UTF-8 LF JavaScript 😊 🚨

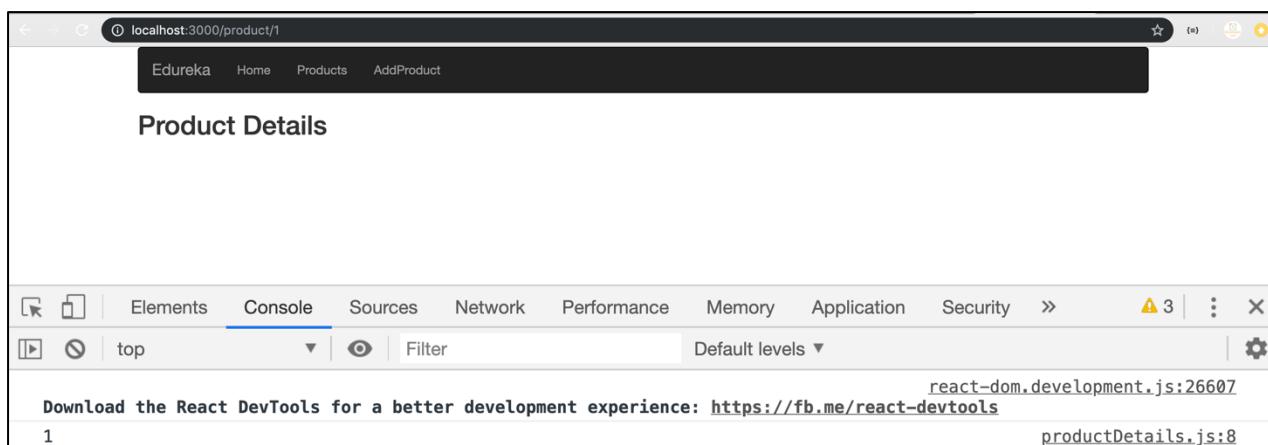
```

```

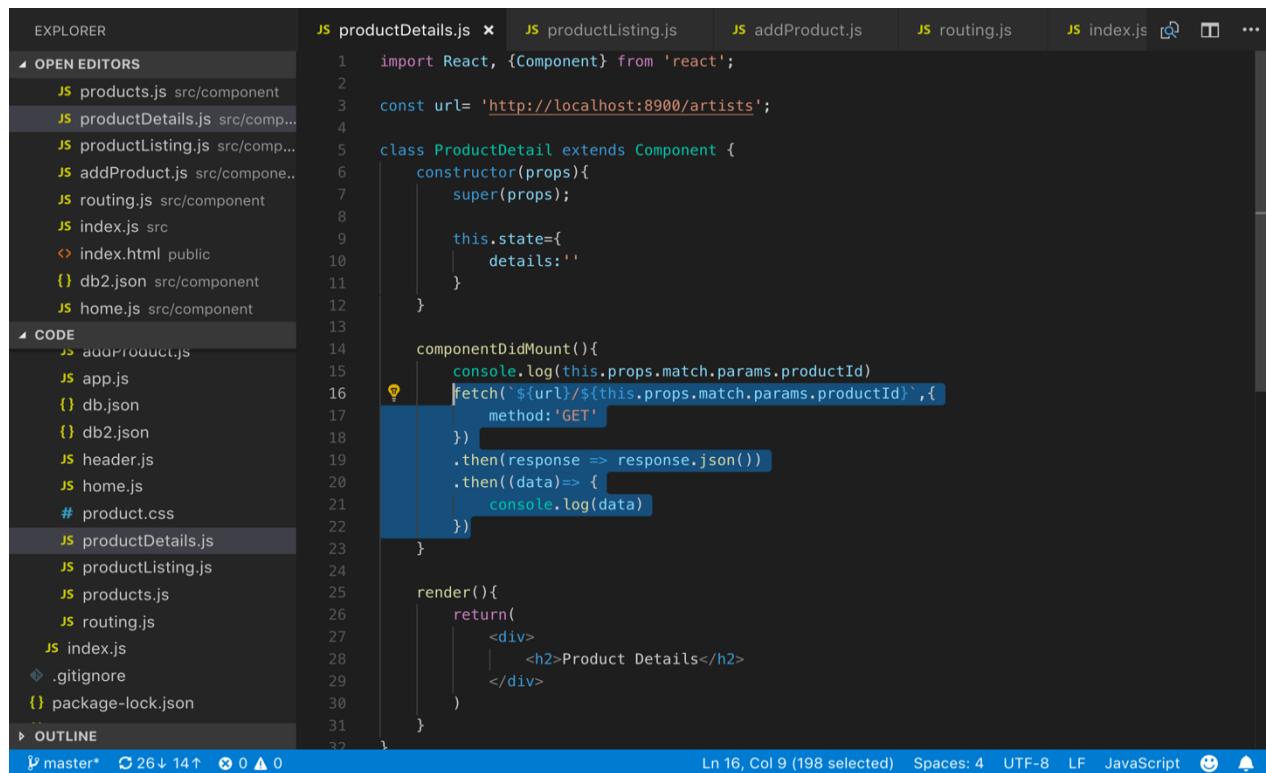
JS productDetails.js
1 import React, {Component} from 'react';
2
3 const url= 'http://localhost:8900/artists';
4
5 class ProductDetail extends Component {
6
7     componentDidMount(){
8         console.log(this.props.match.params.productId)
9     }
10
11     render(){
12         return(
13             <div>
14                 <h2>Product Details</h2>
15             </div>
16         )
17     }
18 }
19
20 export default ProductDetail;

```

**Step 17:** If you check the console of the details page, you can see the param in the console.



**Step 18:** Move to `fetch()` and this time pass the param in the URL, to retrieve particular data (later, set it to a state).



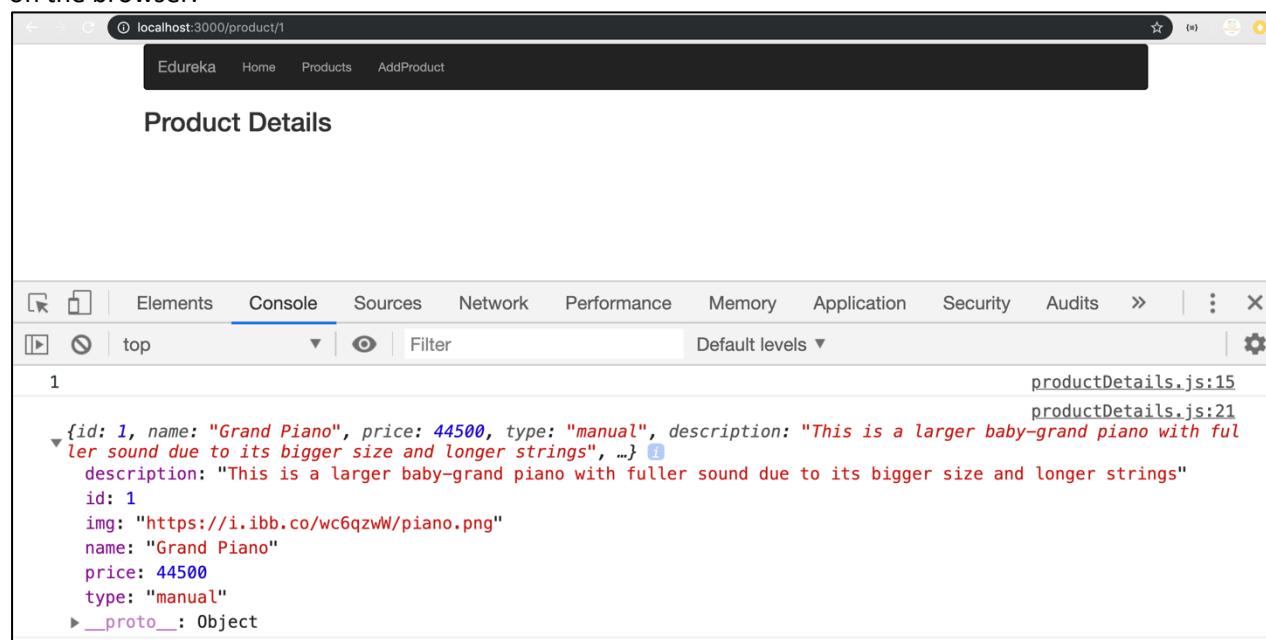
```

productDetails.js
1 import React, {Component} from 'react';
2
3 const url= 'http://localhost:8900/artists';
4
5 class ProductDetail extends Component {
6     constructor(props){
7         super(props);
8
9         this.state={
10             details:''
11         }
12     }
13
14     componentDidMount(){
15         console.log(this.props.match.params.productId)
16         fetch(`${url}/${this.props.match.params.productId}`,{
17             method:'GET'
18         })
19         .then(response => response.json())
20         .then((data)=> {
21             console.log(data)
22         })
23     }
24
25     render(){
26         return(
27             <div>
28                 <h2>Product Details</h2>
29             </div>
30         )
31     }
32 }

```

# eureka!

**Step 19:** Now, when you check the console of details page, you can see the data which we have to display on the browser.



The browser address bar shows `localhost:3000/product/1`. The page title is "Product Details". The developer tools console tab is selected, showing the following output:

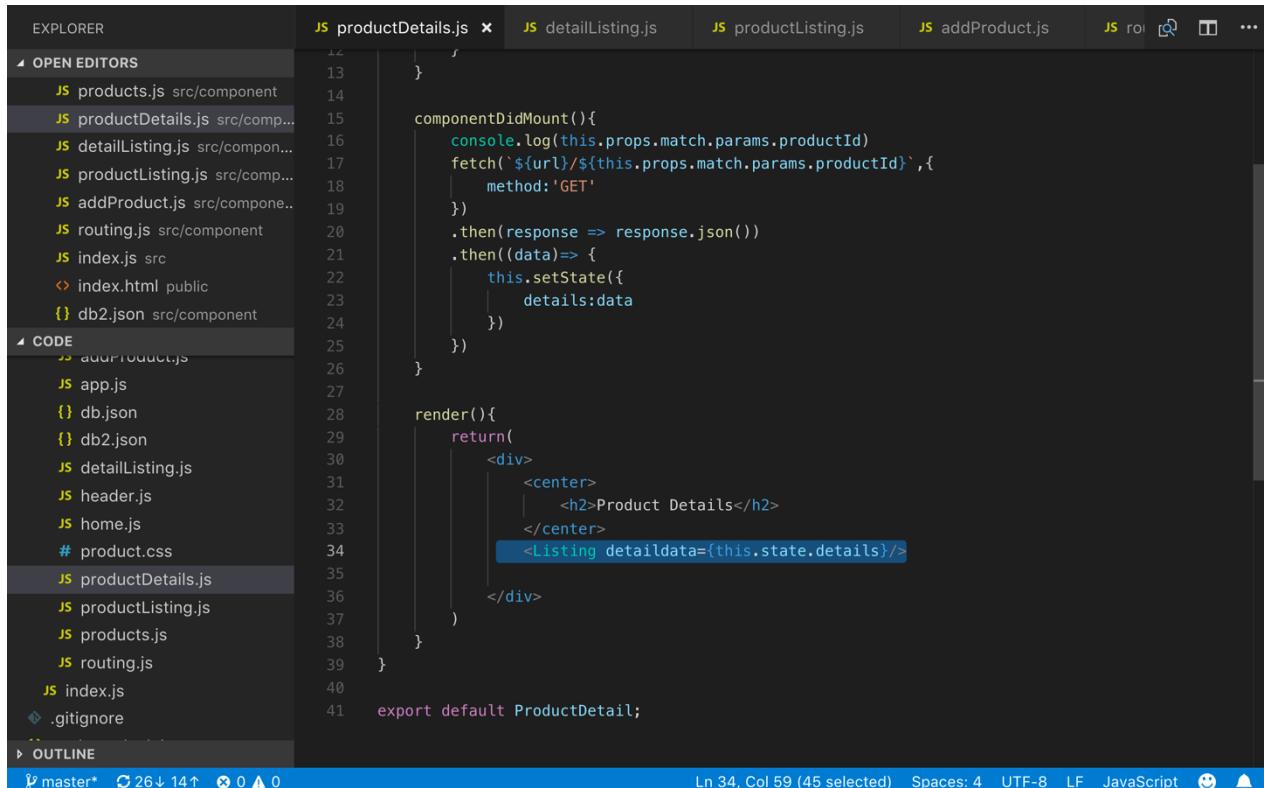
```

productDetails.js:1
productDetails.js:21
{
  id: 1,
  name: "Grand Piano",
  price: 44500,
  type: "manual",
  description: "This is a larger baby-grand piano with fuller sound due to its bigger size and longer strings",
  ...
}
description: "This is a larger baby-grand piano with fuller sound due to its bigger size and longer strings"
id: 1
img: "https://i.ibb.co/wc6qzwW/piano.png"
name: "Grand Piano"
price: 44500
type: "manual"
__proto__: Object

```

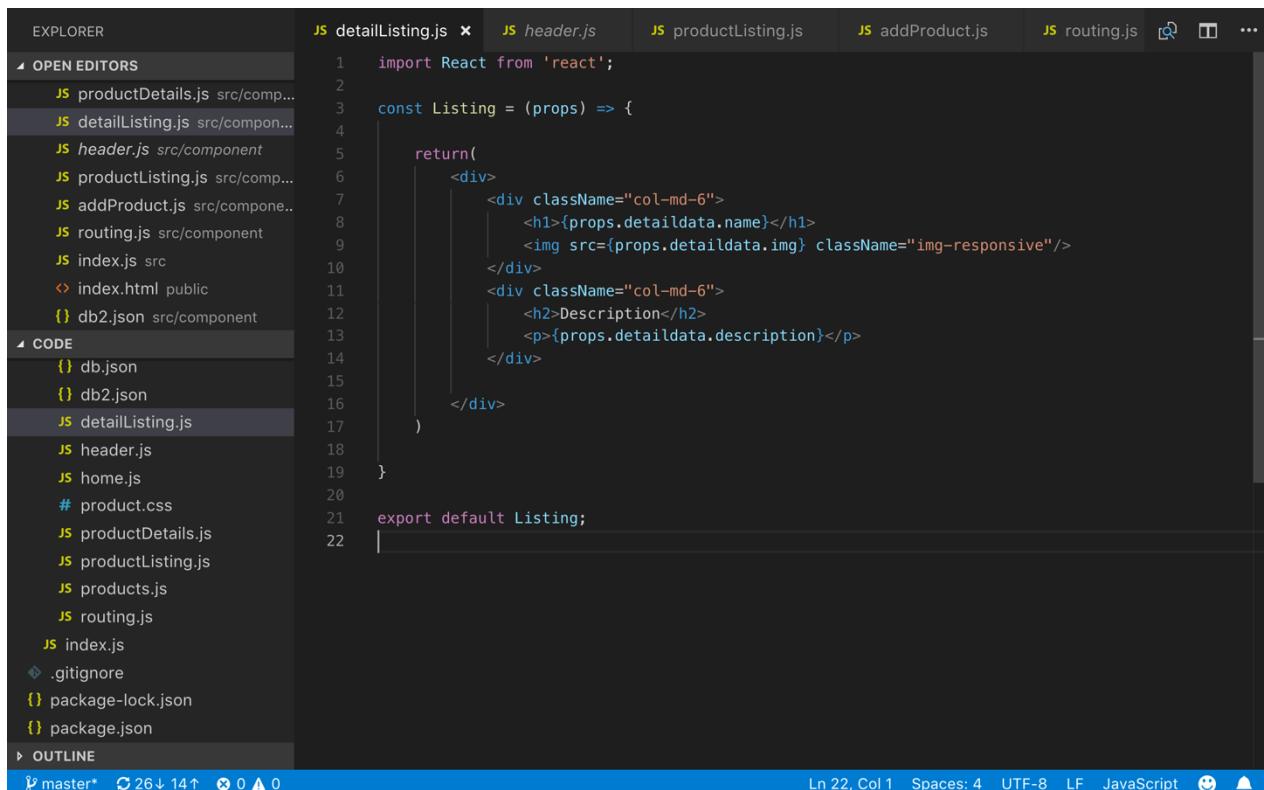
## Module 3: Handling Navigation with Routes

Step 20: Create a file called **detail\_listing.js**, to send the output of the details page.



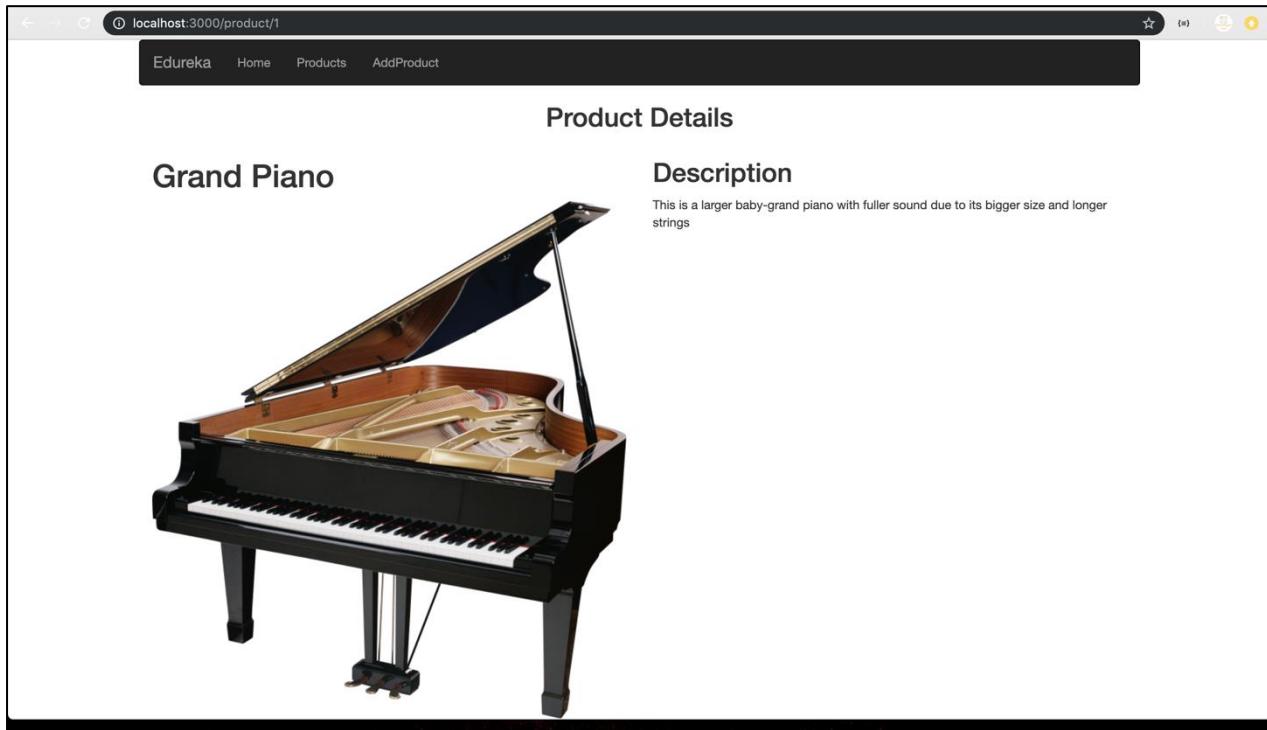
```
productDetails.js
12
13 }
14
15 componentDidMount(){
16     console.log(this.props.match.params.productId)
17     fetch(`${url}/${this.props.match.params.productId}`,{
18         method:'GET'
19     })
20     .then(response => response.json())
21     .then((data)=> {
22         this.setState({
23             details:data
24         })
25     })
26 }
27
28 render(){
29     return(
30         <div>
31             <center>
32                 <h2>Product Details</h2>
33             </center>
34             <Listing detaildata={this.state.details}/>
35         </div>
36     )
37 }
38
39 }
40
41 export default ProductDetail;
```

Step 21: Perform the **data binding** to display the data of respective product in the details page.



```
detailListing.js
1 import React from 'react';
2
3 const Listing = (props) => {
4
5     return(
6         <div>
7             <div className="col-md-6">
8                 <h1>{props.detaildata.name}</h1>
9                 <img src={props.detaildata.img} className="img-responsive"/>
10            </div>
11            <div className="col-md-6">
12                <h2>Description</h2>
13                <p>{props.detaildata.description}</p>
14            </div>
15        </div>
16    )
17
18 }
19
20
21 export default Listing;
```

**Step 22:** Finally, by clicking on any particular product, the details of the respective product is displayed on the screen.



## Conclusion:

We have successfully displayed the details of the respective product from an API with the help of routing methods.