ReactJS | Router

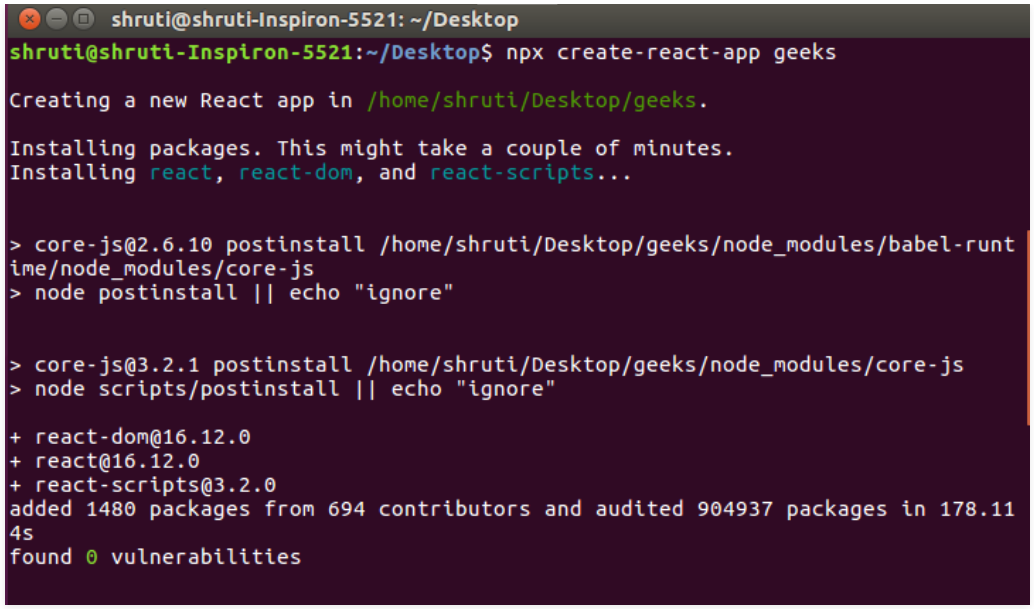
Difficulty Level : Medium ● Last Updated : 21 Oct, 2021

React Router is a standard library for routing in React. It enables the navigation among views of various components in a React Application, allows changing the browser URL, and keeps the UI in sync with the URL.
Let us create a simple application to React to understand how the React Router works. The application will contain three components: home component, about a component, and contact component. We will use React Router to navigate between these components.

Setting up the React Application: Create a React application using create-react-app and lets call it **geeks**.

Note: If you've previously installed create-react-app globally via npm, directly use the command below:

```
npx create-react-app geeks
```

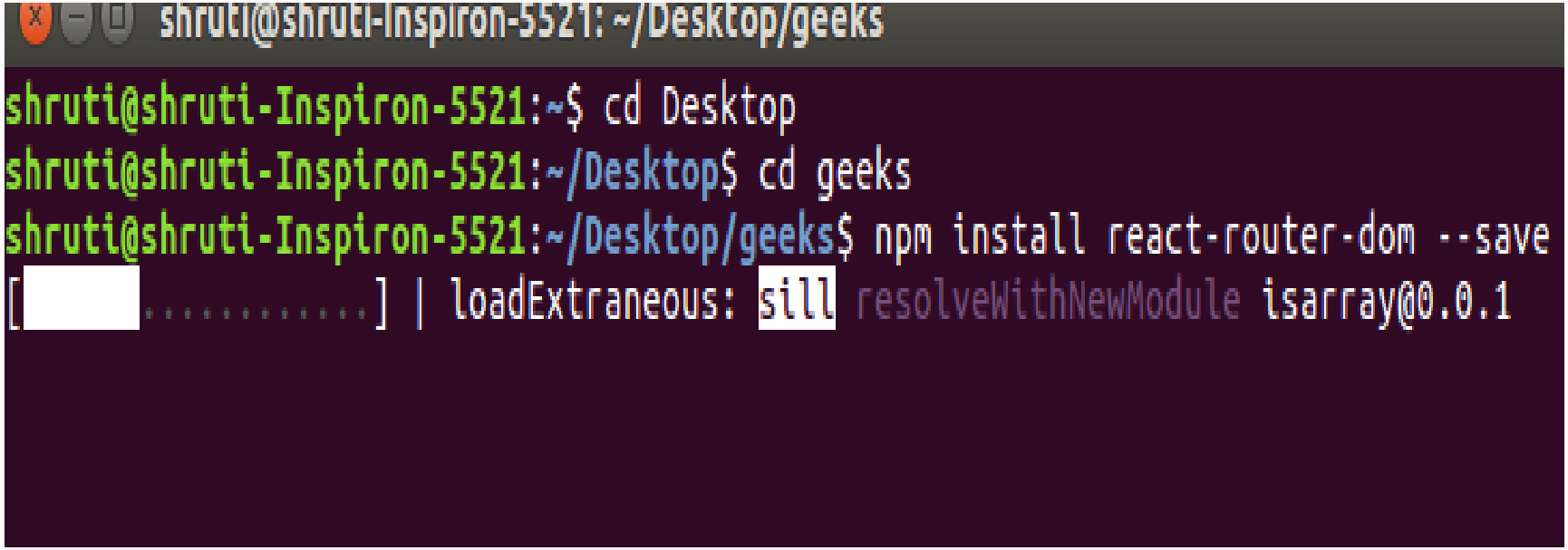


Your development environment is ready. Let us now install React Router in our Application.

Installing React Router: React Router can be installed via npm in your React application. Follow the steps given below to install Router in your React application:

- **Step 1:** cd into your project directory i.e **geeks**.
- **Step 2:** To install the React Router use the following command:

npm install –save react-router-dom



After installing react-router-dom, add its components to your React application.

Adding React Router Components: The main Components of React Router are:

- **BrowserRouter:** BrowserRouter is a router implementation that uses the HTML5 history

API(pushState, replaceState and the popstate event) to keep your UI in sync with the URL. It is the parent component that is used to store all of the other components.

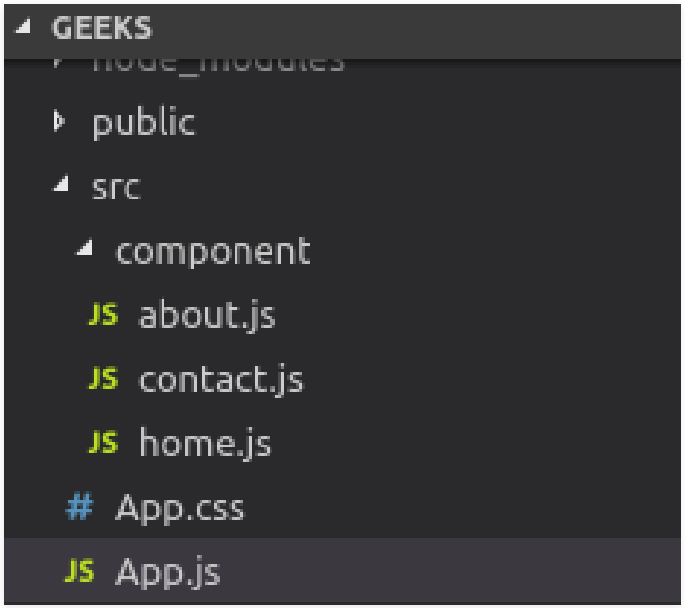
- **Route:** Route is the conditionally shown component that renders some UI when its path matches the current URL.
- **Link:** Link component is used to create links to different routes and implement navigation around the application. It works like HTML anchor tag.
- **Switch:** Switch component is used to render only the first route that matches the location rather than rendering all matching routes. Although there is no defying functionality of SWITCH tag in our application because none of the LINK paths are ever going to coincide. But let's say we have a route (Note that there is no EXACT in here), then all the Route tags are going to be processed which start with '/' (all Routes start with /). This is where we need SWITCH statement to process only one of the statements.

To add React Router components in your application, open your project directory in the editor you use and go to **app.js** file. Now, add the below given code in app.js.

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

Note: BrowserRouter is aliased as Router.

Using React Router: To use React Router, let us first create few components in the react application. In your project directory, create a folder named **component** inside the src folder and now add 3 files named **home.js**, **about.js** and **contact.js** to the component folder.



Let us add some code to our 3 components:

- **Home.js:**

```
import React from 'react';

function Home (){
    return <h1>Welcome to the world of Geeks!</h1>
}

export default Home;
```

- **About.js:**

```
import React from 'react';

function About () {
    return <div>
        <h2>GeeksforGeeks is a computer science portal for geeks!</h2>

        Read more about us at :
        <a href="https://www.geeksforgeeks.org/about/">
            https://www.geeksforgeeks.org/about/
        </a>
    </div>
}
export default About;
```

- **Contact.js:**

```
import React from 'react';

function Contact (){
 return <address>
            You can find us here:<br />
            GeeksforGeeks<br />
            5th & 6th Floor, Royal Kapsons, A- 118, <br />
            Sector- 136, Noida, Uttar Pradesh (201305)
        </address>
}

export default Contact;
```

Now, let us include React Router components to the application:

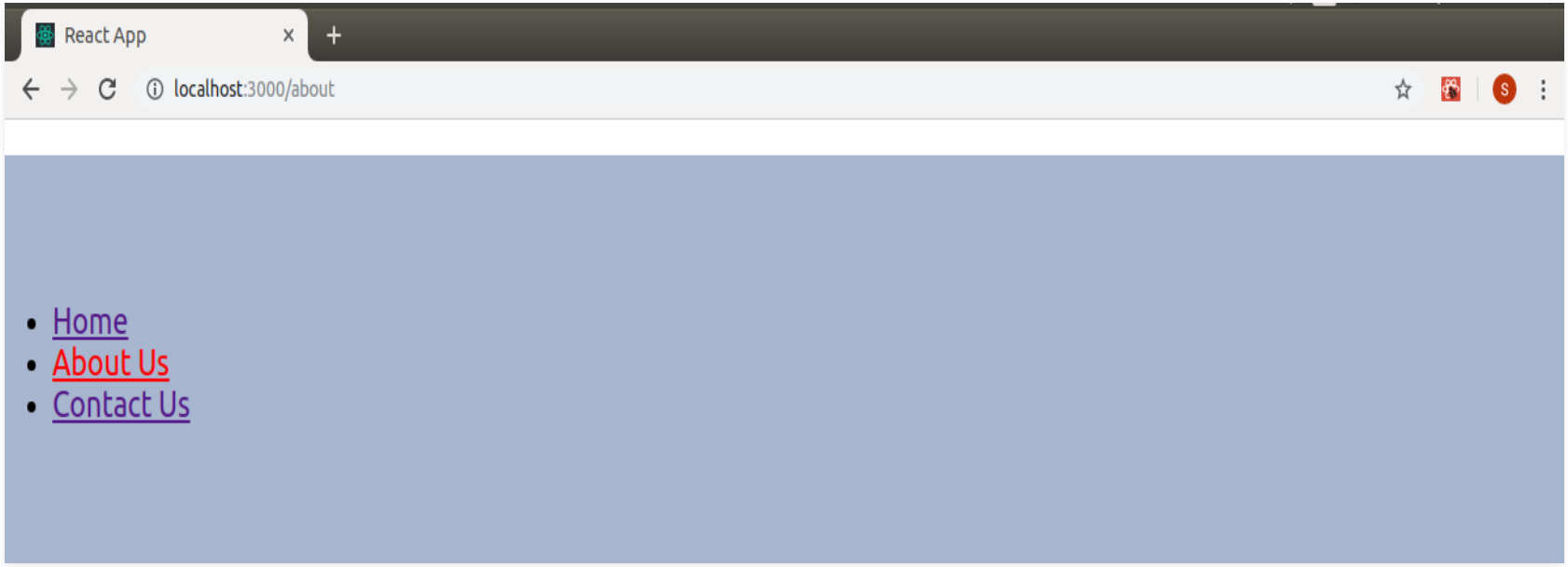
- **BrowserRouter:** Add BrowserRouter aliased as Router to your app.js file in order to wrap all the other components. BrowserRouter is a parent component and can have only single child.

```
class App extends Component {
  render() {
    return (
       <Router>
           <div className="App">
           </div>
       </Router>
   );
  }
}
```

- **Link:** Let us now create links to our components. Link component uses the **to** prop to describe the location where the links should navigate to.

```
<div className="App">
    <ul>
      <li>
        <Link to="/">Home</Link>
      </li>
      <li>
        <Link to="/about">About Us</Link>
      </li>
      <li>
        <Link to="/contact">Contact Us</Link>
      </li>
    </ul>
</div>
```

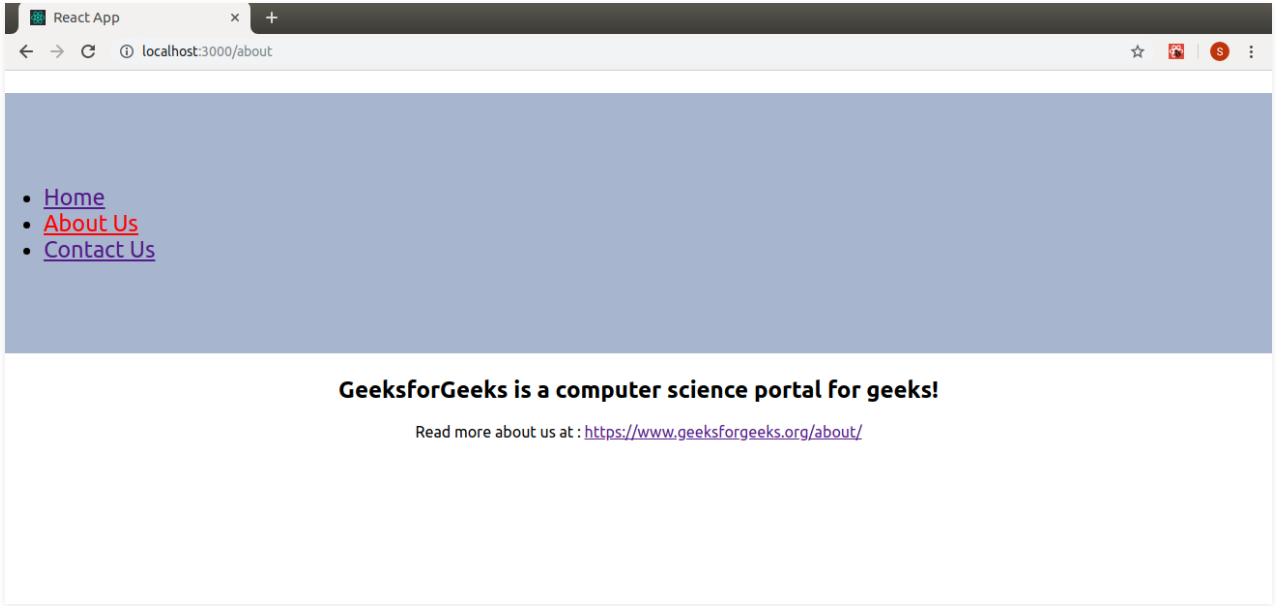
Now, run your application on the local host and click on the links you created. You will notice the url changing according the value in **to** props of the Link component.



- **Route:** Route component will now help us to establish the link between component's UI and the URL. To include routes to the application, add the code give below to your app.js.

```
<Route exact path='/' component={Home}></Route>
<Route exact path='/about' component={About}></Route>
<Route exact path='/contact' component={Contact}></Route>
```

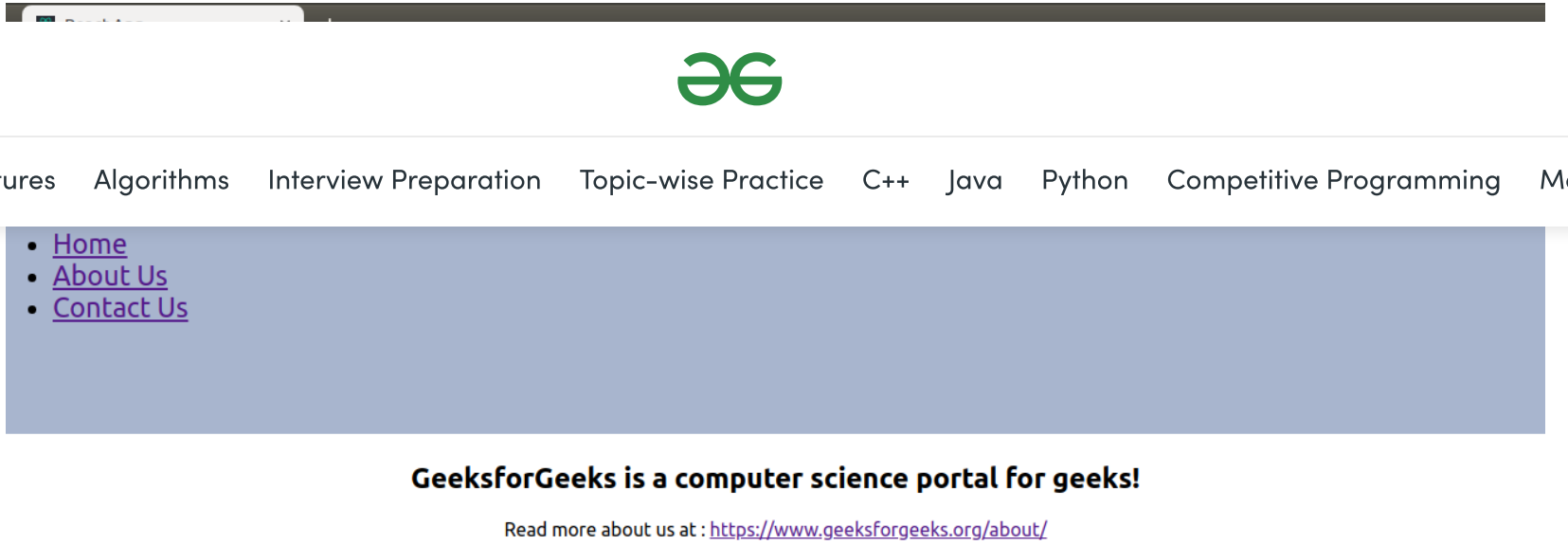
Components are linked now and clicking on any link will render the component associated with it.



Let us now try to understand the props associated with the Route component.

1.exact: It is used to match the exact value with the URL. For Eg., exact path='/about' will only render the component if it exactly matches the path but if we remove exact from the syntax, then

UI will still be rendered even if the strucute is like /about/10.



2. path: Path specifies a pathname we assign to our component.

- **Switch:** To render a single component, wrap all the routes inside the Switch Component.
- **3. component:** It refers to the component which will render on matching the path.

```
<Switch>
    <Route exact path='/' component={Home}></Route>
    <Route exact path='/about' component={About}></Route>
    <Route exact path='/contact' component={Contact}></Route>
</Switch>
```

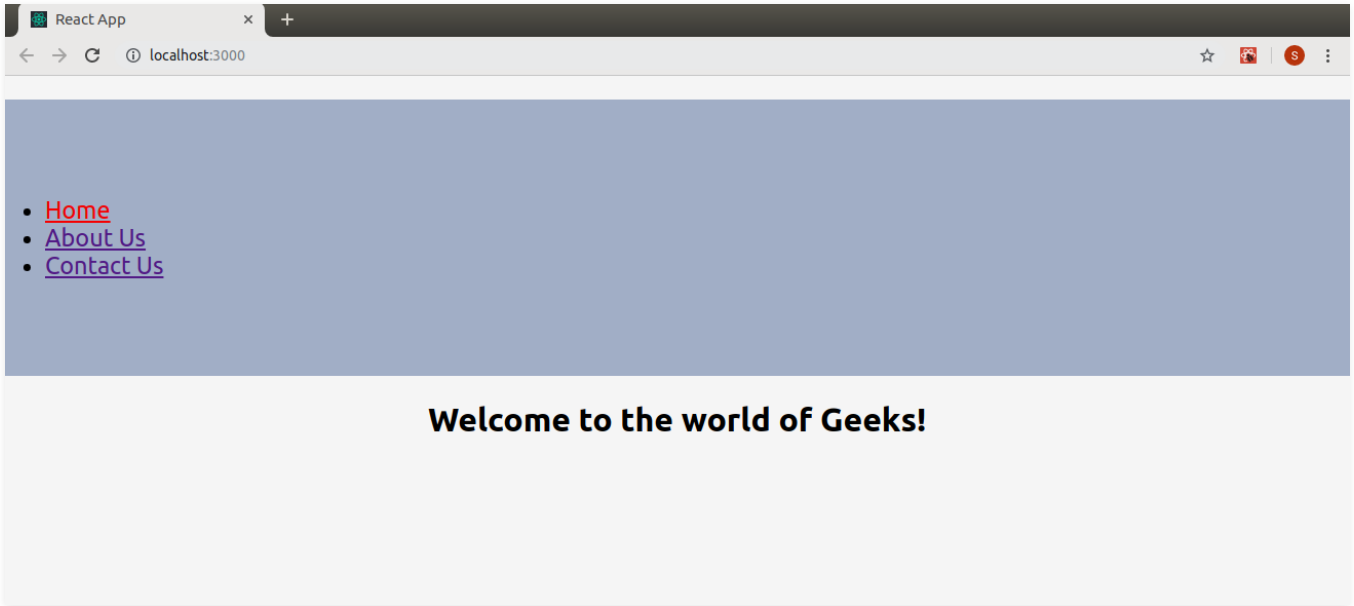
Switch groups together several routes, iterates over them and finds the first one that matches the path. Thereby, the corresponding component to the path is rendered.

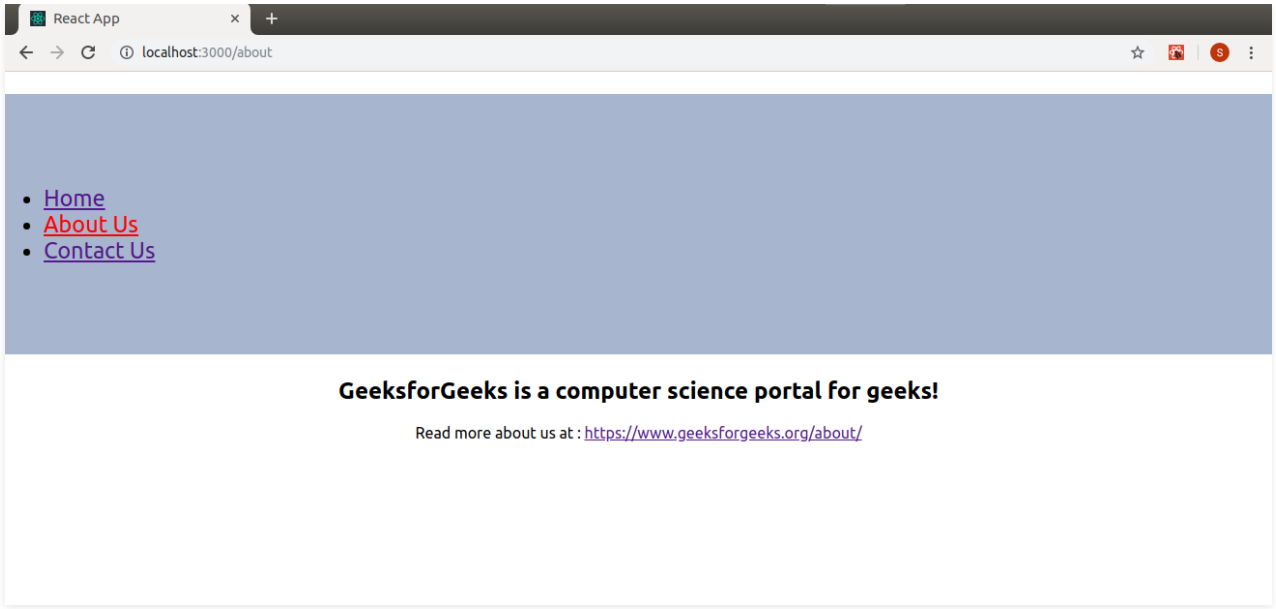
After adding all the components here is our complete source code:

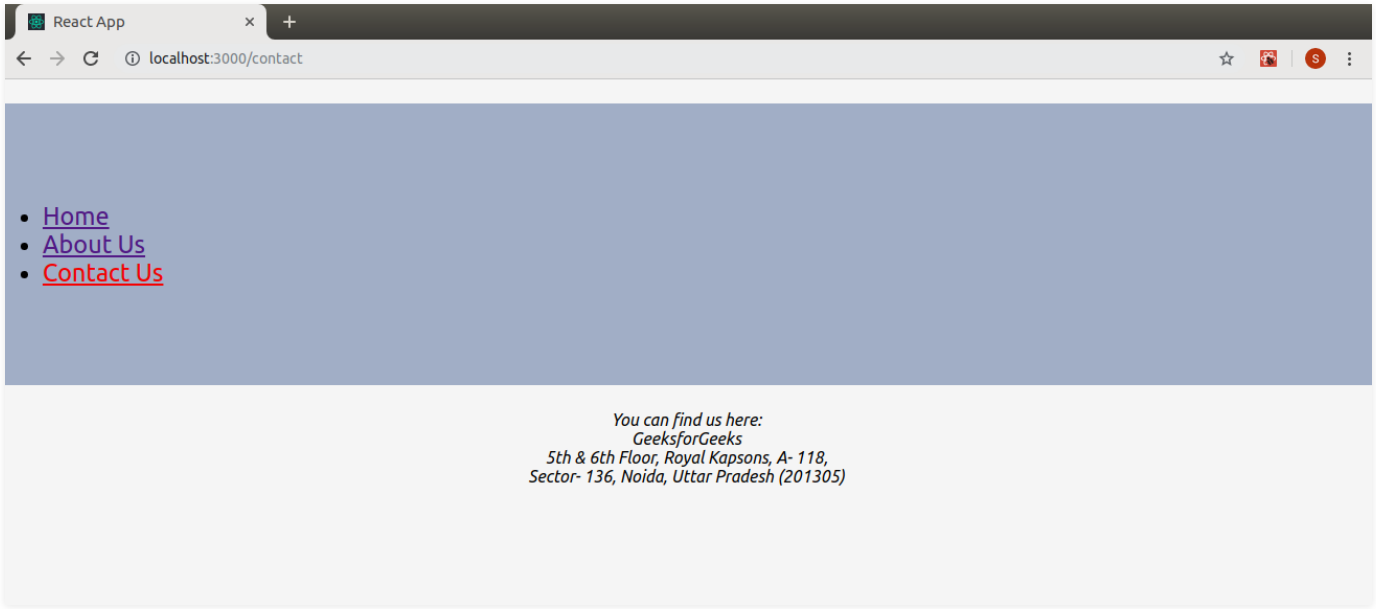
```
import React, { Component } from 'react';
import { BrowserRouter as Router, Route, Link, Switch } from 'react-router-dom';
import Home from './component/home';
import About from './component/about';
import Contact from './component/contact';
import './App.css';

class App extends Component {
  render() {
    return (
       <Router>
           <div className="App">
            <ul className="App-header">
              <li>
                <Link to="/">Home</Link>
              </li>
              <li>
                <Link to="/about">About Us</Link>
              </li>
              <li>
                <Link to="/contact">Contact Us</Link>
              </li>
            </ul>
            <Switch>
              <Route exact path='/' component={Home}></Route>
              <Route exact path='/about' component={About}></Route>
              <Route exact path='/contact' component={Contact}></Route>
            </Switch>
```

```
            </div>
          </Router>
      );
    }
}

export default App;
```

Now, we you can click on the links and navigate to different components. React Router keeps your application UI in sync with the URL.







Finally, we have successfully implemented navigation in our React application using React Router.

Previous

Context in React

Next

ReactJS | Types of Routers

RECOMMENDED ARTICLES

01 **Routing in Angular JS using Angular UI Router**
16, May 18

02 **Build Your First Router in Node.js with Express**
13, Aug 21

03 **What is the difference between ngRoute and ui-router?**
20, Sep 19

04 **How to set or update page title using UI-Router?**
22, Jun 20

05 **Express.js | router.use() Function**
17, Jun 20

06 **Express.js router.param() function**
15, Jul 20

07 **Express.js express.Router() Function**
03, Jul 20

08 **Express.js router.all() Function**
08, Jul 20

Article Contributed By :

shruti10gandotra
@shruti10gandotra

Vote for difficulty

Current difficulty : Medium

Easy Normal Medium Hard Expert

Improved By : aman1122singhas, chaudharinikita9999

Article Tags : Picked, react-js, Web Technologies

Improve Article Report Issue

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

GeeksforGeeks
5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

Company

About Us

Careers

Privacy Policy

Contact Us

Copyright Policy

Learn

Algorithms

Data Structures

Languages

CS Subjects

Video Tutorials

Web Development

Web Tutorials

HTML

CSS

JavaScript

Bootstrap

Contribute

Write an Article

Write Interview Experience

Internships

Videos