React Router

From the docs :

React Router keeps your UI in sync with the URL. It has a simple API with powerful features like lazy code loading, dynamic route matching, and location transition handling built right in. Make the URL your first thought, not an after-thought.

Getting started

npm install -g create-react-app

create-react-app AppName

cd AppName

npm install react-router-dom@next

Required imports

```
import React from 'react';

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

Basic Routing

```
const BasicExample = () => (
  <Router>
    <div>
      <ul>
        <li><Link to="/">Home</Link></li>
        <li><Link to="/about">About</Link></li>
        <li><Link to="/topics">Topics</Link></li>
      </ul>
/* The reason we are using Link here instead of anchor(a) tag is because Link component exposed by the react-router-dom behaves same as anchor tag but stops the page from refreshing. Instead is just replaces the current Component with the one specified for the current path. */
```

Basic Routing

```
      <Route exact path="/" component={Home}/>
      <Route path="/about" component={About}/>
      <Route path="/topics" component={Topics}/>
    </div>
  </Router>
);
```

/* The Route component mainly takes two props to work. First is “path”, .which is the path that is to be matched and second is the “component” that is to be rendered when that specific path has been matched. Now here in this example we can also see a third prop “exact”. If this prop is not passed the Home component will always get rendered because “/” will always be there is the path. So we provide the prop exact which tells the router to render the the Home component if the path matches exactly. */

Url Parameters

```
/* Lets assume our links look like this */
 <ul>
        <li><Link to="/randomId1">Manoj</Link></li>
        <li><Link to="/randomId2">Arun</Link></li>
        <li><Link to="/randomId3">Someone</Link></li>
 </ul>
```

/* and on click we want to show the data of the person whose name was clicked. In stead of making three different components for this we can just make a single component for showing data based on id. */

Url Parameters

```
/* Our route will look like this */
<Route path="/:id" component={UserDetails}/>
/* UserDetails Component */
const UserDetails = ({ match }) => (
  <div>
    <h3>ID: {match.params.id}</h3>
  </div>
)
/* The “match” prop is passed by the react-router to the component */
```

Redirection (Authentication)

```
const PrivateRoute = ({ component, ...rest }) => (
  <Route {...rest} render={props => (
    Auth.isAuthenticated ? (
      React.createElement(component, props)
    ) : (
      <Redirect to={{
        pathname: '/login',
        state: { from: props.location }
      }}/>
    )
  )}/>
)
```

Redirection (Authentication)

```
/* Using PrivateRoute Component */

<PrivateRoute path="/some-protected-route" component={ProtectedComponent}/>

/*
The state that we passes in the redirect component will be available in ProtectedComponent
in this.props.location.state.
*/
```