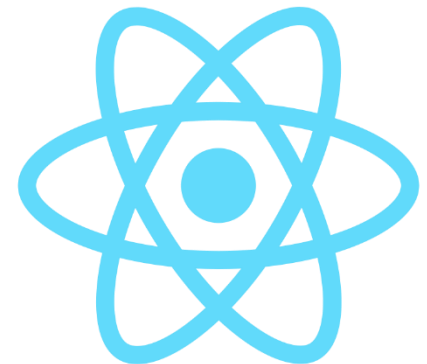


# React

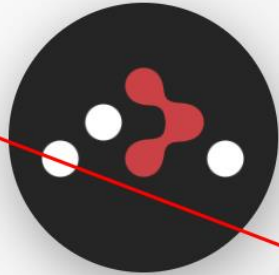
Riccardo Cattaneo



# Router

Fino a questo momento ci siamo limitati ad avere e gestire una sola pagina. Ma React ci permette di gestire più pagine attraverso la libreria **Router**.

Questa libreria non è presente di default nella libreria React, quindi va importata a livello di progetto e quindi aggiunta nel file **package.json**. Andiamo su google e cerchiamo «react router»



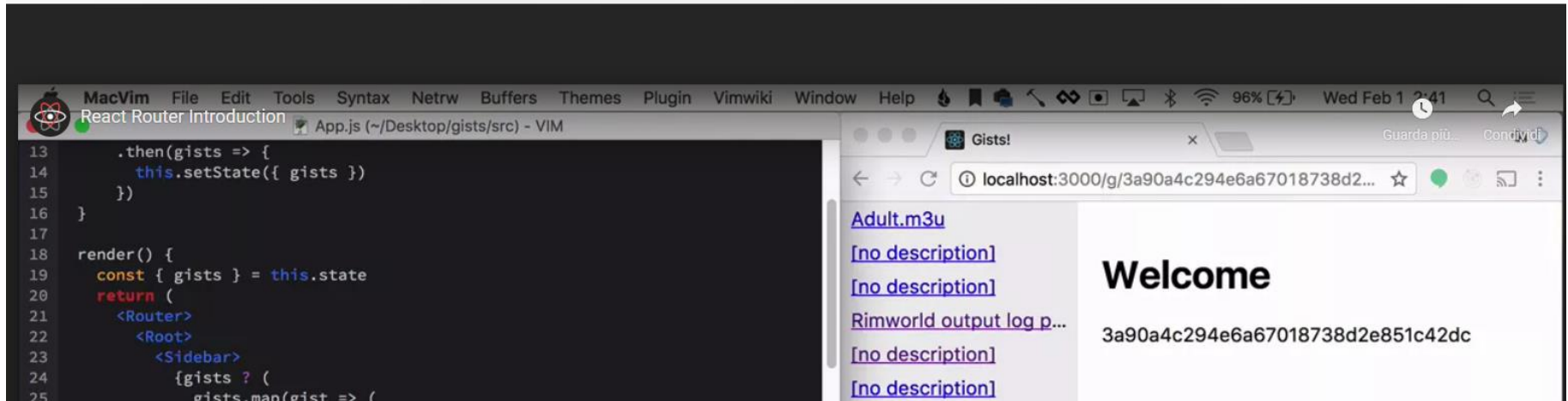
LEARN ONCE, ROUTE ANYWHERE

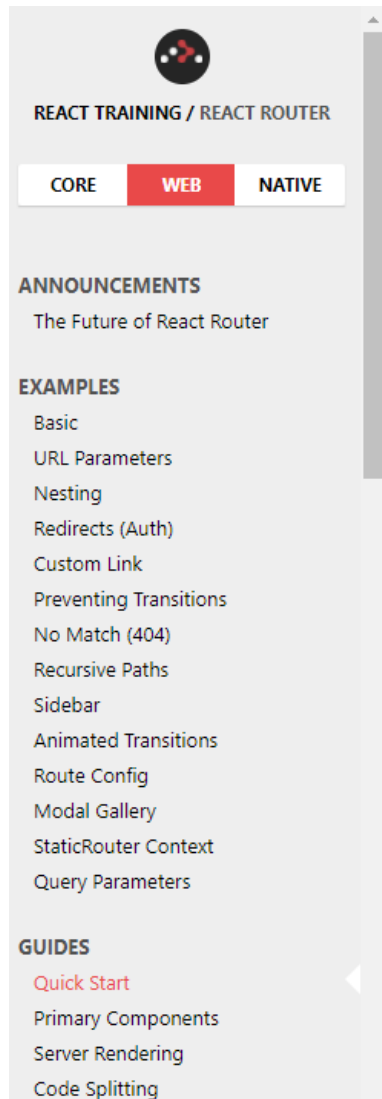
# REACT ROUTER

Components are the heart of React's powerful, declarative programming model. React Router is a collection of **navigational components** that compose declaratively with your application. Whether you want to have **bookmarkable URLs** for your web app or a composable way to navigate in **React Native**, React Router works wherever React is rendering--so take your pick!

WEB

NATIVE





React

## Quick Start

To get started with React Router in a web app, you'll need a React web app. If you need to create one, we recommend you try [Create React App](#). It's a popular tool that works really well with React Router.

First, install create-react-app and make a new project with it.

## Installation

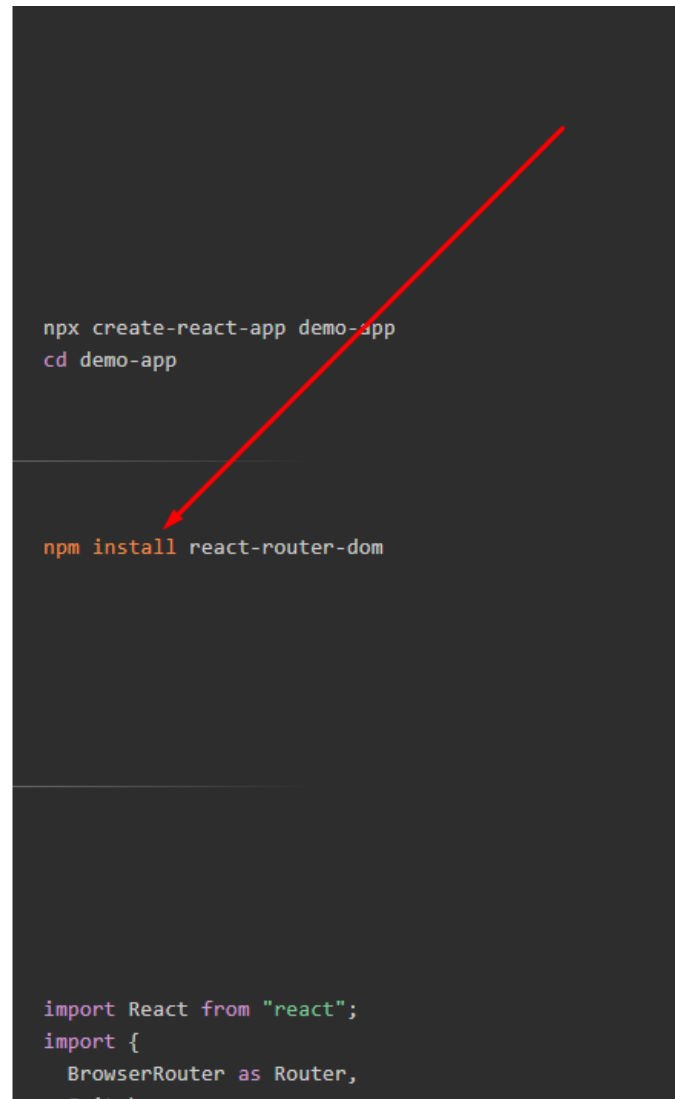
You can install React Router from [the public npm registry](#) with either npm or [yarn](#). Since we're building a web app, we'll use react-router-dom in this guide.

Next, copy/paste either of the following examples into `src/App.js`.

### 1st Example: Basic Routing

In this example we have 3 "pages" handled by the router: a home page, an about page, and a users page. As you click around on the different `<Link>`s, the router renders the matching `<Route>`.

Note: Behind the scenes a `<Link>` renders an `<a>` with a real href, so people using the keyboard for navigation or screen readers will still be able to use this app.

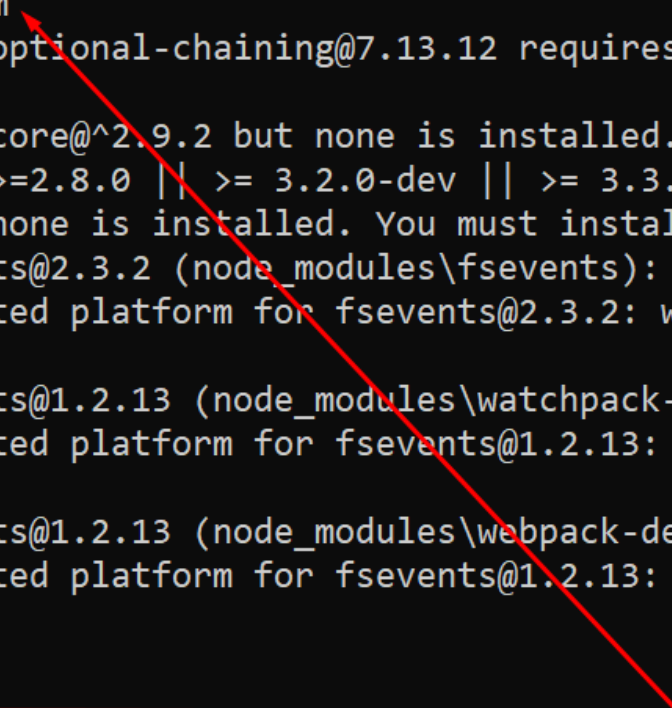


```
C:\Users\Rick\Desktop\ciao>npm install react-router-dom
npm WARN @babel/plugin-bugfix-v8-spread-parameters-in-optional-chaining@7.13.12 requires a peer of @babel/core@>=7.13.0 but none is installed. You must install peer dependencies yourself.
npm WARN bootstrap@5.0.1 requires a peer of @popperjs/core@^2.9.2 but none is installed. You must install peer dependencies yourself.
npm WARN tsutils@3.21.0 requires a peer of typescript@>=2.8.0 || >= 3.2.0-dev || >= 3.3.0-dev || >= 3.6.0-beta || >= 3.7.0-dev || >= 3.7.0-beta but none is installed. You must install peer dependencies yourself.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@2.3.2 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@2.3.2: wanted >= 14.5.0 < 15.0.0 (current: 14.17.0) on x64
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.13 (node_modules\watchpack-chokidar\chokidar\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.13: wanted >= 14.5.0 < 15.0.0 (current: 14.17.0) on x64
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.13 (node_modules\webpack-dev-server\node_modules\webpack\node_modules\watchpack-chokidar\chokidar\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.13: wanted >= 14.5.0 < 15.0.0 (current: 14.17.0) on x64

+ react-router-dom@5.2.0
added 1 package from 1 contributor and audited 1960 packages in 8.901s

139 packages are looking for funding
  run `npm fund` for details

found 87 vulnerabilities (82 moderate, 5 high)
  run `npm audit fix` to fix them, or `npm audit` for details
```



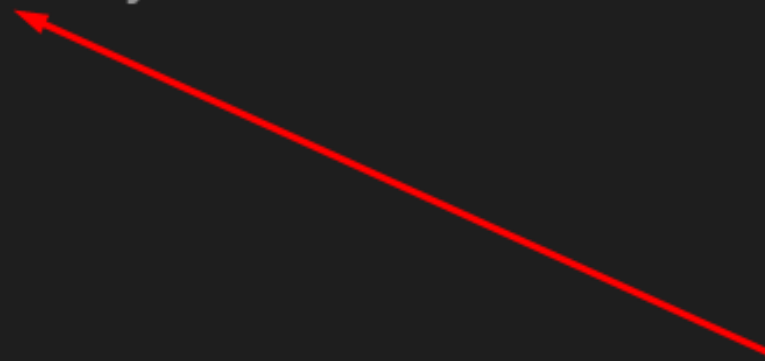
Gli elementi della libreria router-dom che esamineremo sono : BrowserRouter, Route, Switch.

```
import React from 'react'
import {BrowserRouter, Route, Switch} from 'react-router-dom'

const EsempioRouter = () => {
  return (
    <div>

    </div>
  )
}

export default EsempioRouter
```



# BrowserRouter

**BrowserRouter** è il componente in grado di gestire il Routing della nostra applicazione, mentre **Route** ci permette di creare e gestire uno specifico indirizzo di destinazione. Questa è la sintassi :

```
import React from 'react'
import {BrowserRouter, Route, Switch} from 'react-router-dom'
import ChiSono from './ChiSono'

const EsempioRouter = () => {
  return (
    <BrowserRouter>
      <Route path='/chisono' component={chisono}/>
    </BrowserRouter>
  )
}

export default EsempioRouter
```



Per completare l'esempio andiamo a creare un componente chiamato ChiSono e mettiamo al suo interno un testo di prova :

```
import React from 'react'

const ChiSono = () => {
  return (
    <div>
      Presentazione del Corso Html CSS Javascript React
    </div>
  )
}

export default ChiSono
```



Per fare una prima prova possiamo aprire ora il nostro browser e nella url principale ci aggiungiamo /chisono così come configurato nel BrowserRouter :

A browser address bar with a light gray background. On the left, there are three navigation icons: a left arrow, a right arrow, and a circular refresh icon. To the right of these icons is a small circular information icon (an 'i' inside a circle) followed by the text 'localhost:3000/chisono' in a blue, monospace-style font.

Presentazione del Corso Html CSS Javascript React

Se aggiungiamo un altro componente, ad esempio la Home Page che vogliamo che si carica di default senza mettere nulla nella nostra nella nostra ul possiamo fare in questo modo :

```
import React from 'react'
import {BrowserRouter, Route, Switch} from 'react-router-dom'
import ChiSono from './ChiSono'
import Home from './Home'

const EsempioRouter = () => {
  return (
    <BrowserRouter>
      <Route path="/" component={Home}/>
      <Route path="/chisono" component={ChiSono}/>
    </BrowserRouter>
  )
}

export default EsempioRouter
```

Ma se proviamo a chiamare tutte e due le url possiamo notare che, se chiamo la home page visualizzo solo la home page, ma se chiamo localhost:3000/chisono viene visualizzato sia la home che il chisono, questo avviene perché l'url localhost:3000/chisono trova corrispondenza sia in localhost:3000/ sia in localhost:3000/chisono. Se ad esempio aggiungo un terzo componente e lo chiamiamo prodotti, cosa succede se chiamo la url localhost:3000/prodotti ? Proviamo...

```
const ChiSono = () => {  
  return (  
    <div>  
      Presentazione del Corso Html CSS Javascript React  
    </div>  
  )  
}  
  
export default ChiSono
```

```
const EsempioRouter = () => {  
  return (  
    <BrowserRouter>  
      <Route path="/" component={Home}/>  
      <Route path="/chisono" component={ChiSono}/>  
      <Route path="/prodotti" component={Prodotti}/>  
    </BrowserRouter>  
  )  
}
```



localhost:3000/chisono

Home page

Presentazione del Corso Html CSS Javascript React

```
import React from 'react'
```

```
const Prodotti = () => {
```

```
  return (
```

```
    <div>
```

```
      Pagina dei Prodotti
```

```
    </div>
```

```
  )
```

```
}
```

```
export default Prodotti
```

```
const EsempioRouter = () => {
```

```
  return (
```

```
    <BrowserRouter>
```

```
      <Route path="/" component={Home}/>
```

```
      <Route path="/chisono" component={ChiSono}/>
```

```
      <Route path="/prodotti" component={Prodotti}/>
```

```
    </BrowserRouter>
```

```
  )
```

```
}
```



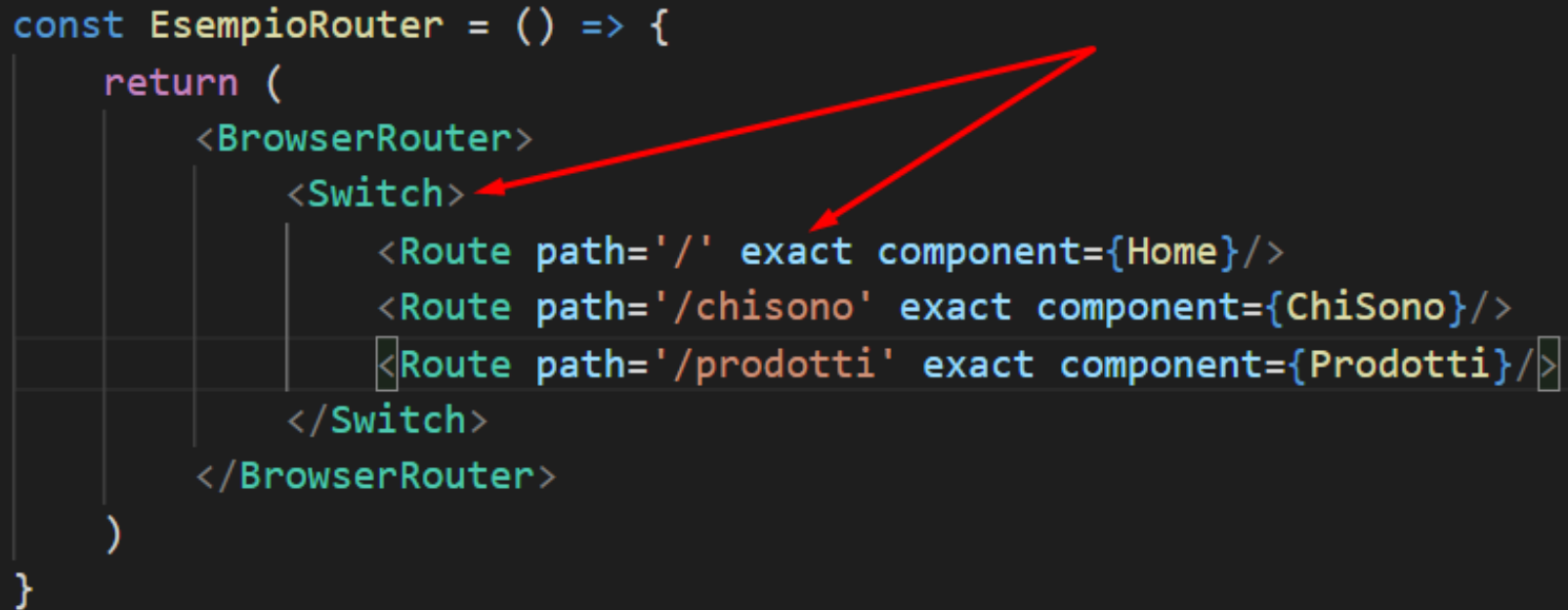
localhost:3000/prodotti

React Home page

Pagina dei Prodotti

A questo punto ci viene in aiuto il terzo componente... lo switch. Con lo switch react ci fa tornare solo la url esatta che trova corrispondenza con la url inviata in questo modo :

```
const EsempioRouter = () => {  
  return (  
    <BrowserRouter>  
      <Switch>  
        <Route path="/" exact component={Home}/>  
        <Route path="/chisono" exact component={ChiSono}/>  
        <Route path="/prodotti" exact component={Prodotti}/>  
      </Switch>  
    </BrowserRouter>  
  )  
}
```

A diagram with two red arrows. One arrow starts from the opening tag of the <Switch> component and points to the opening tag of the <BrowserRouter> component. The second arrow starts from the closing tag of the <Switch> component and points to the closing tag of the <BrowserRouter> component. This illustrates that the Switch component is a child of the BrowserRouter component.

# Barra di Navigazione

Andiamo ora a creare una barra di navigazione che ci permette di navigare senza dover cambiare manualmente l'indirizzo sul browser.


Prendiamo come esempio un menu da bootstrap ed importiamolo in un nuovo componente chiamato menu :

```
const Menu = () => {
  return (
    <div>
      <nav className="navbar navbar-expand-lg navbar-light bg-light">
        <div className="container-fluid">
          <div className="collapse navbar-collapse" id="navbarSupportedContent">
            <ul className="navbar-nav me-auto mb-2 mb-lg-0">
              <li className="nav-item">
                <a className="nav-link active" aria-current="page" href="#">Home</a>
              </li>
              <li className="nav-item">
                <a className="nav-link active" aria-current="page" href="#">Chi Sono</a>
              </li>
              <li className="nav-item">
                <a className="nav-link active" aria-current="page" href="#">Contatti</a>
              </li>
            </ul>
          </div>
        </div>
      </nav>
    </div>
  )
}
```



# Integriamo ora nel menu il BrowserRouter come visto in precedenza :

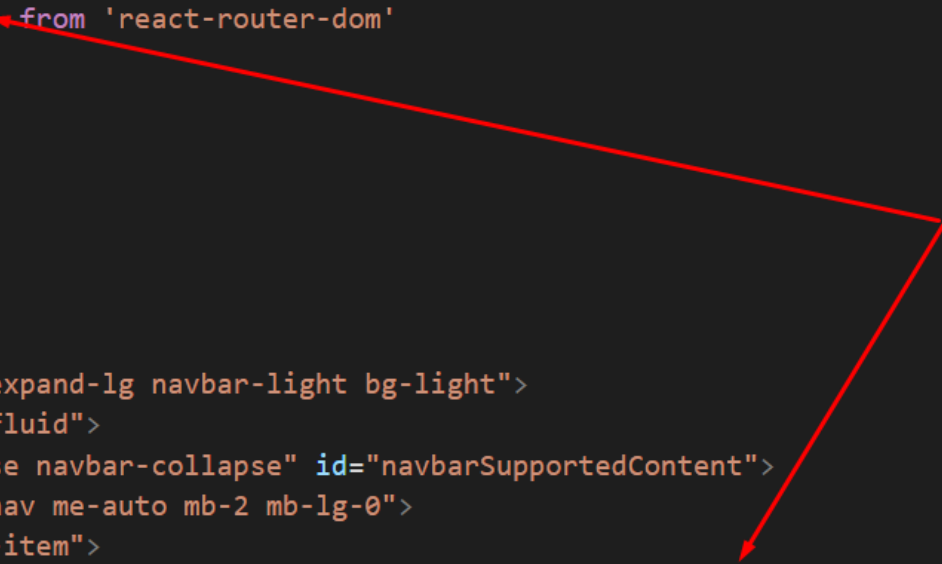
```
const Menu = () => {  
  return (  
    <BrowserRouter>  
      <div>  
        <nav className="navbar navbar-expand-lg navbar-light bg-light">  
          <div className="container-fluid">  
            <div className="collapse navbar-collapse" id="navbarSupportedContent">  
              <ul className="navbar-nav me-auto mb-2 mb-lg-0">  
                <li className="nav-item">  
                  <a className="nav-link active" aria-current="page" href="/">Home</a>  
                </li>  
                <li className="nav-item">  
                  <a className="nav-link active" aria-current="page" href="/chisono">Chi Sono</a>  
                </li>  
                <li className="nav-item">  
                  <a className="nav-link active" aria-current="page" href="/prodotti">Prodotti</a>  
                </li>  
              </ul>  
            </div>  
          </nav>  
        </div>  
        <Switch>  
          <Route path="/" exact component={Home}/>  
          <Route path="/chisono" exact component={ChiSono}/>  
          <Route path="/prodotti" exact component={Prodotti}/>  
        </Switch>  
      </BrowserRouter>  
    )  
  )  
}
```



Se proviamo a navigare all'interno del nostro menu possiamo vedere che tutto funziona correttamente anche se c'è ancora un piccolo problema : ad ogni click la pagina si ricarica, e non è il massimo per un applicazione front end, per risolvere il problema React ci viene in aiuto tramite un altro componente : `<Link to>` che va a sostituire il buon vecchio `<a href>` in questo modo viene caricato solo il componente e non tutta la pagina :

```
import {BrowserRouter, Route, Switch, Link} from 'react-router-dom'
import ChiSono from './lezione10/ChiSono'
import Home from './lezione10/Home'
import Prodotti from './lezione10/Prodotti'

const Menu = () => {
  return (
    <BrowserRouter>
    <div>
      <nav className="navbar navbar-expand-lg navbar-light bg-light">
        <div className="container-fluid">
          <div className="collapse navbar-collapse" id="navbarSupportedContent">
            <ul className="navbar-nav me-auto mb-2 mb-lg-0">
              <li className="nav-item">
                <Link className="nav-link active" aria-current="page" to="/">Home</Link>
              </li>
              <li className="nav-item">
                <Link className="nav-link active" aria-current="page" to="/chisono">Chi Sono</Link>
              </li>
              <li className="nav-item">
                <Link className="nav-link active" aria-current="page" to="prodotti">Prodotti</Link>
              </li>
            </ul>
          </div>
        </div>
      </nav>
    </div>
  )
}
```



# Pagina di errore

Può capitare che l'utente inserisca una url errata o che comunque per qualche motivo il nostro software genera una url non corretta. In questo caso React ci viene in aiuto sempre tramite Route mettendo come path un \* asterisco, in questo modo gli stiamo dicendo che se non trova nessuna corrispondenza (\* vuol dire tutti gli altri) allora vai in questa pagina (che normalmente è una pagina di errore generico) :

```

        </li>
      </ul>
    </div>
  </div>
</nav>
</div>
<Switch>
  <Route path="/" exact component={Home}/>
  <Route path="/chisono" exact component={ChiSono}/>
  <Route path="/prodotti" exact component={Prodotti}/>
  <Route path="*" component={ErrorPage}/>
</Switch>
</BrowserRouter>
)

```

Ricordiamo di mettere la pagina di errore alla fine. Proprio perché asterisco vuol dire TUTTI, se lo mettiamo all'inizio entrerà sempre in quella pagina...