PRACTICUM MODULE

FRAMEWORK BASED PROGRAMMING

Module
5

- How to install react router – basic Router
- URL Parameters
- Use Nesting Router
- Use Redirects (Auth)

POLITEKNIK NEGERI MALANG

React Router

In the web, if you want to switch pages from one page to another, a routing process is needed. Routing itself is the process of mapping a URL to a page with content / UI (User Interface) according to the intended URL. If you want to make a route you need additional libraries because it is not directly available. It turns out that there are several libraries that can be used, among which the most familiar libraries are *react-router* and *reach / router*. Actually, what is used for routing in React is usually the *react- router-dom* child of *react-router*, which besides *react -router-dom*, there is also *react- router-native* which can be used for Android and iOS application development. The *react- router- dom* library can be installed by running the command

yarn add react-router-dom

or

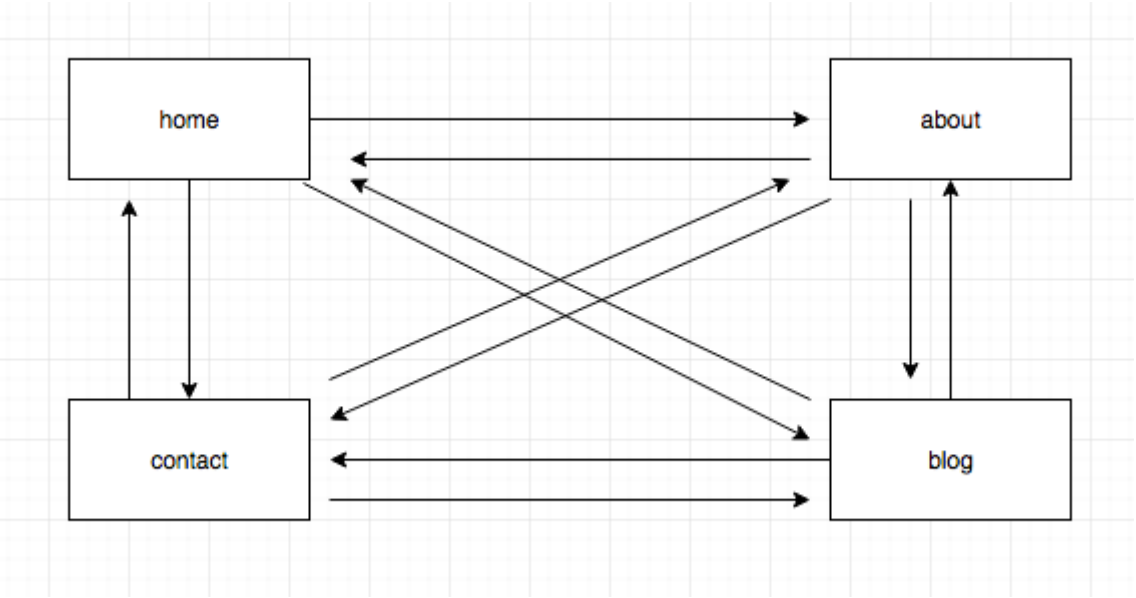
npm i react-router-dom

Basically, *react-router-dom* has 2 types of routers that we can use, namely *HashRouter* and *BrowserRouter*. Both have their respective uses depending on what type of Web we are going to create. For example, if we want to create a static web or there is no server to render dynamic data, then we should use a *HashRouter*. Conversely, if we create a web that uses dynamic data with a backend server, then using *BrowserRouter* is the right choice.

For example building a simple catalog application where a user can move dynamically from one page to another. The list of pages that the user can usually access is:

- Home or landingpage
- About, page to display information about the website.
- Contact, a page that displays the contacts (website owners or managers) who can be contacted
- Blog, page to display posts or news.

Navigation will be placed on the menu item so that the movement can be from any page to any destination. The following is an illustration of the navigation scheme above.



Gambar 1. Navigation Scheme

from the illustration above we will create five components, that is :

- home component
- about component
- blog component
- contact components

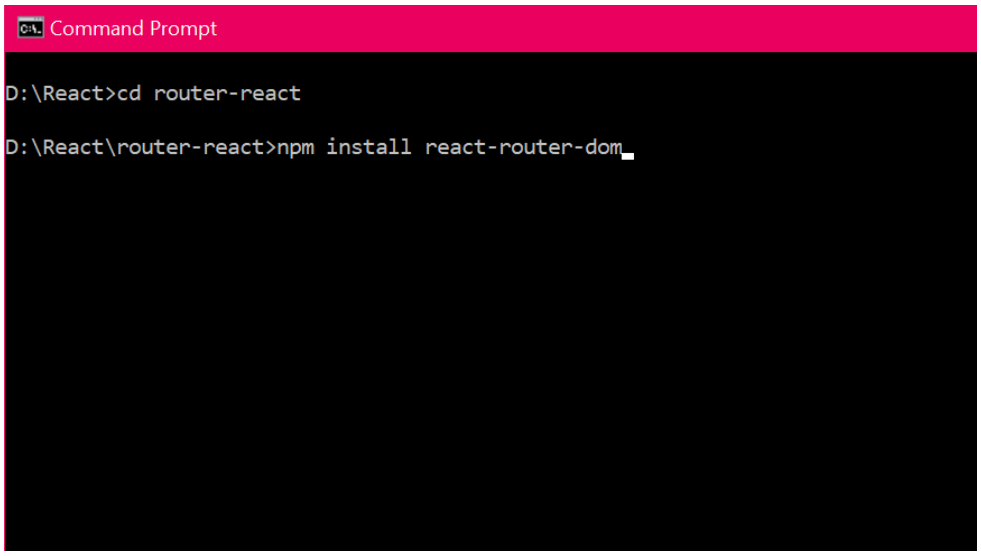
Practicum

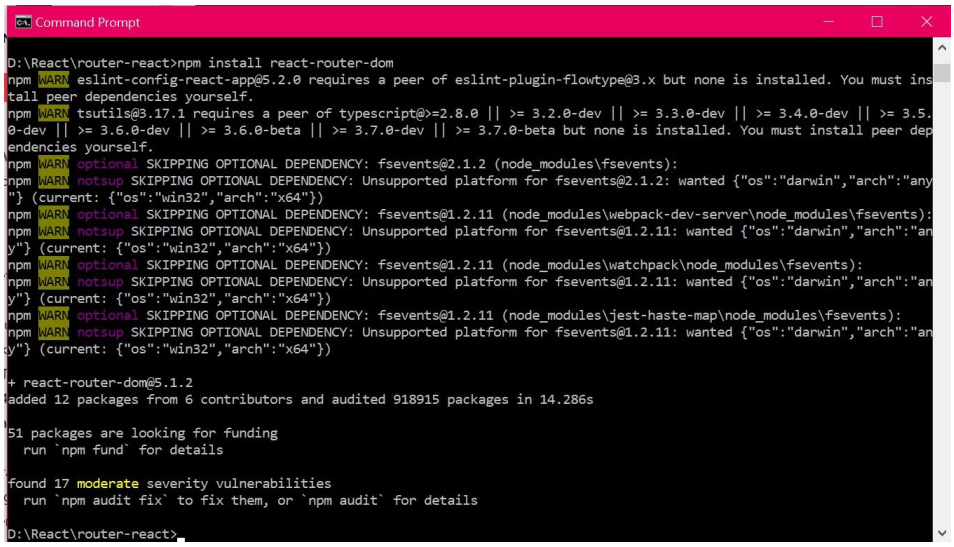
In building a web SPA (Single Page Application) we need the React Router library. For that, the first step we take is installing the React Router library.

1.1 Practicum Steps

a. Basic Router

1. After we successfully installed React (NPM install), we need to add the React Router library by adding npm install react-router-dom.Setelah kita sukses melakukan instalasi react (*npm install*), kita perlu menambahkan library react router dengan menambahkan *npm install react-router-dom*.





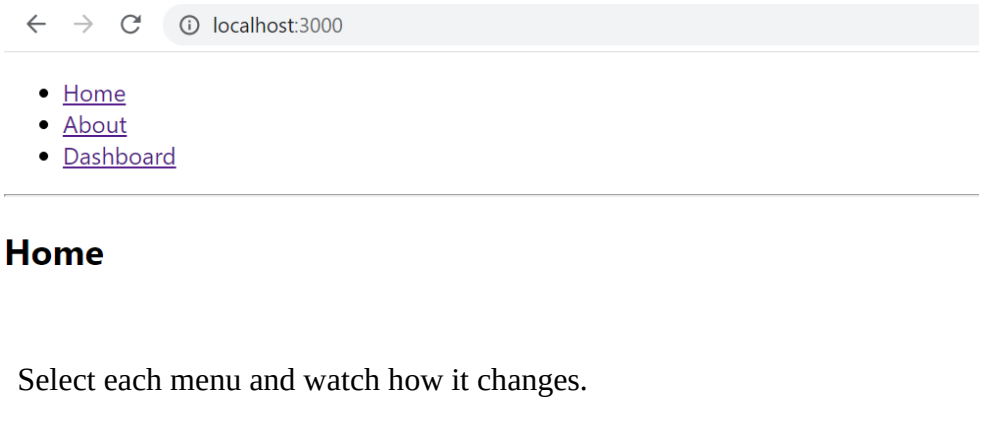
2. Then open app.js and type the code like the following picture.

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

// Situs ini memiliki 3 halaman, yang semuanya direnderer secara dinamis di browser.
// Meskipun halaman tidak pernah di-refresh, perhatikan bagaimana React Router
// membuat URL selalu terbarui saat Anda menavigasi situs.
// Ini menjaga riwayat browser, memastikan hal-hal seperti tombol kembali
// dan bookmark berfungsi dengan baik.

export default function BasicExample() {
  return (
    <Router>
      <div>
        <ul>
          <li>
            <Link to="/">Home</Link>
          </li>
          <li>
            <Link to="/about">About</Link>
          </li>
          <li>
            <Link to="/dashboard">Dashboard</Link>
          </li>
        </ul>
        <hr />
```

```

        <Switch>
          <Route exact path="/">
            <Home />
          </Route>
          <Route path="/about">
            <About />
          </Route>
          <Route path="/dashboard">
            <Dashboard />
          </Route>
        </Switch>
      </div>
    </Router>
  );
}

```

```
// Anda dapat menganggap komponen ini sebagai "halaman(konten)" di situs Anda.

function Home() {
  return (
    <div>
      <h2>Home</h2>
    </div>
  );
}

function About() {
  return (
    <div>
      <h2>About</h2>
    </div>
  );
}

function Dashboard() {
  return (
    <div>
      <h2>Dashboard</h2>
    </div>
  );
}
```

3. Do npm start, then check the results will be as below.



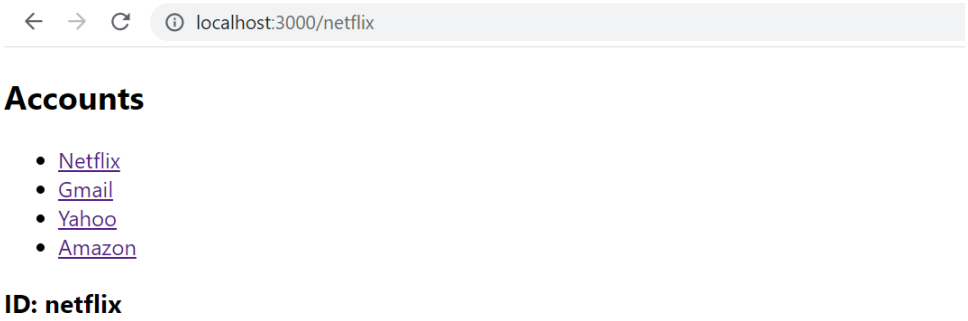
b. URL Parameters

1. In this experiment, we are trying to create a React Router link using params, so that we only create one template which can change its content according to what we click on.
2. Type the code like the following picture.

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

// Params adalah placeholder di URL yang dimulai dengan titik dua,
//seperti param `: id` yang didefinisikan dalam route dalam contoh ini.
export default function ParamsExample() {
  return (
    <Router>
      <div>
        <h2>Accounts</h2>
        <ul>
          <li>
            <Link to="/netflix">Netflix</Link>
          </li>
          <li>
            <Link to="/gmail">Gmail</Link>
          </li>
          <li>
            <Link to="/yahoo">Yahoo</Link>
          </li>
          <li>
            <Link to="/amazon">Amazon</Link>
          </li>
        </ul>
```

```
31
32          <Switch>
33            <Route path="/:id" children={<Child />} />
34          </Switch>
35        </div>
36      </Router>
37    );
38  }
39
40  function Child() {
41    let { id } = useParams();
42
43    return (
44      <div>
45        <h3>ID: {id}</h3>
46      </div>
47    );
48  }
```

3. Check the results, it will be as below.



The result is almost the same as the previous practicum, but with a different method.

c. Use Nesting Router

1. If we want to fork links in React Router, then we can use this method as a solution.
2. Type the code like the following picture.

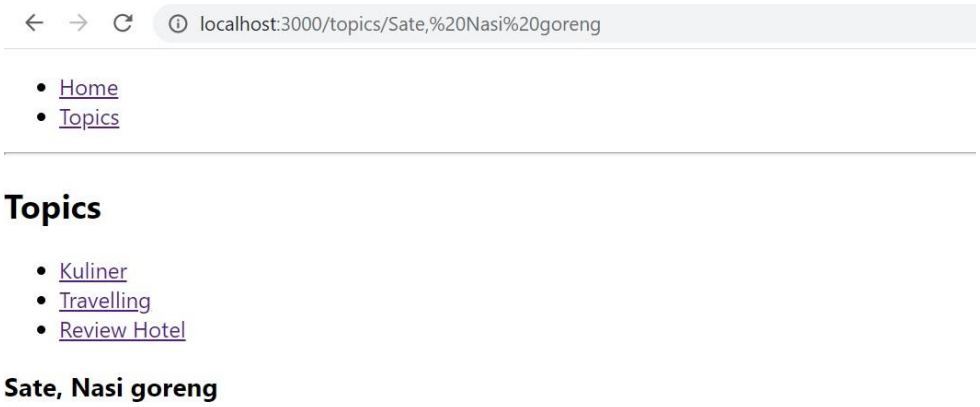
```jsx
import React from "react";
import {
  BrowserRouter as Router,
  Switch,
  Route,
  Link,
  useParams,
  useRouteMatch
} from "react-router-dom";

// Karena route adalah komponen regular react,
//sehingga dapat ditampilkan di mana saja dalam penempatannya,
//termasuk dalam child element.

// Ini akan membantu anda untuk memecah menjadi beberapa bundel
//karena pemisahan kode pada aplikasi React Router sama dengan
//pemecahan kode pada aplikasi react lainnya.

export default function NestingExample() {
  return (
    <Router>
      <div>
        <ul>
          <li>
            <Link to="/">Home</Link>
          </li>
          <li>
            <Link to="/topics">Topics</Link>
          </li>
        </ul>
```

```jsx
        <hr />

        <Switch>
          <Route exact path="/">
            <Home />
          </Route>
          <Route path="/topics">
            <Topics />
          </Route>
        </Switch>
      </div>
    </Router>
  );
}

function Home() {
  return (
    <div>
      <h2>Home</h2>
    </div>
  );
}
```

```jsx
function Topics() {
// `path` untuk membuat jalur <Route> yang terhadap rute induk,
//sedangkan` url` untuk membuat link.
  let { path, url } = useRouteMatch();
  return (
    <div>
      <h2>Topics</h2>
      <ul>
        <li>
          <Link to={`${url}/Sate, Nasi goreng`}>Kuliner</Link>
        </li>
        <li>
          <Link to={`${url}/Wisata alam, Museum`}>Travelling</Link>
        </li>
        <li>
          <Link to={`${url}/Ibis, JW Marriot`}>Review Hotel</Link>
        </li>
      </ul>

      <Switch>
        <Route exact path={path}>
          <h3>Please select a topic.</h3>
        </Route>
        <Route path={`${path}/:topicId`}>
          <Topic />
        </Route>
      </Switch>
    </div>
  );
}
```

```jsx

function Topic() {
  let { topicId } = useParams();

  return (
    <div>
      <h3>{topicId}</h3>
    </div>
  );
}
```

3. Check the result and try click each link.



We can see that the parent link above will determine the results that come out below it, and the link below can bring up different results.

d. Use Redirects (Auth)

1. You can use this method to enter a page that requires authentication. After entering and clicking through to other menus including entering a page that requires authentication, the user's position does not need to re-login (because the position is logged) until the user signs out.

2. Type the code like the following picture.

```
import React from "react";
import {
  BrowserRouter as Router,
  Switch,
  Route,
  Link,
  Redirect,
  useHistory,
  useLocation
} from "react-router-dom";

// Pada aplikasi ini memiliki 3 halaman: public page, private page, dan halaman login.
//Untuk masuk ke private page, Anda harus login terlebih dahulu.

// Pertama, klik public page. Kemudian, kunjungi private page.
//karena anda belum login, jadi Anda diarahkan ke halaman login.
//Setelah login, Anda akan diarahkan kembali ke private page.

// Perhatikan perubahan setiap URL. Jika Anda mengklik tombol kembali,
//apakah anda kembali ke halaman login? Tidak, karena anda sudah login.
//Cobalah, maka anda akan kembali ke halaman yang Anda kunjungi sebelum login, yaitu public page.

```

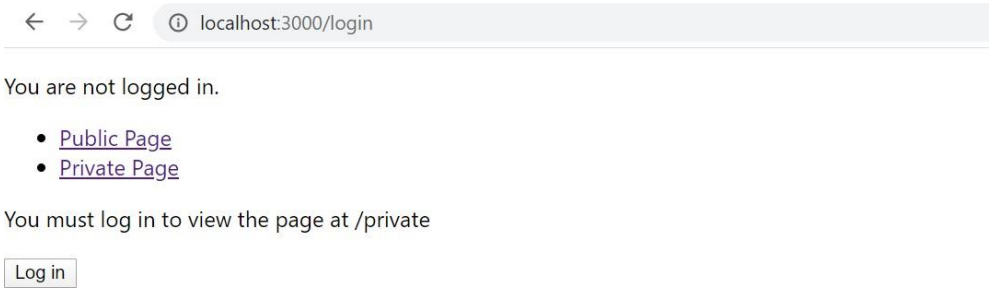
```
export default function AuthExample() {
  return (
    <Router>
      <div>
        <AuthButton />

        <ul>
          <li>
            <Link to="/public">Public Page</Link>
          </li>
          <li>
            <Link to="/private">Private Page</Link>
          </li>
        </ul>

        <Switch>
          <Route path="/public">
            <PublicPage />
          </Route>
          <Route path="/login">
            <LoginPage />
          </Route>
          <PrivateRoute path="/private">
            <ProtectedPage />
          </PrivateRoute>
        </Switch>
      </div>
    </Router>
  );
}
```

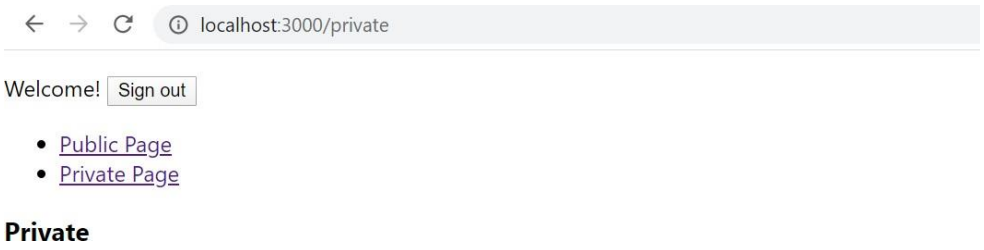
```
const fakeAuth = {
  isAuthenticated: false,
  authenticate(cb) {
    fakeAuth.isAuthenticated = true;
    setTimeout(cb, 100); // fake async
  },
  signout(cb) {
    fakeAuth.isAuthenticated = false;
    setTimeout(cb, 100);
  }
};

function AuthButton() {
  let history = useHistory();

  return fakeAuth.isAuthenticated ? (
    <p>
      Welcome!{" "}
      <button
        onClick={() => {
          fakeAuth.signout(() => history.push("/"));
        }}
      >
        Sign out
      </button>
    </p>
  ) : (
    <p>You are not logged in.</p>
  );
}
```

```
// Pembungkus untuk <Route> yang mengarahkan ke login
// tampilan jika Anda belum terkonfirmasi.

function PrivateRoute({ children, ...rest }) {
  return (
    <Route
      {...rest}
      render={({ location }) =>
        fakeAuth.isAuthenticated ? (
          children
        ) : (
          <Redirect
            to={{
              pathname: "/login",
              state: { from: location }
            }}
          />
        )
      }
    />
  );
}

function PublicPage() {
  return <h3>Public</h3>;
}

function ProtectedPage() {
  return <h3>Private</h3>;
}
```

```
115
116  function LoginPage() {
117    let history = useHistory();
118    let location = useLocation();
119
120    let { from } = location.state || { from: { pathname: "/" } };
121    let login = () => {
122      fakeAuth.authenticate(() => {
123        history.replace(from);
124      });
125    };
126
127    return (
128      <div>
129        <p>You must log in to view the page at {from.pathname}</p>
130        <button onClick={login}>Log in</button>
131      </div>
132    );
133  }
```

3. Check the results, when you click on the private page you are required to log in first, when you are logged in and click on the private page again, you don't need to log back in because we are in a logged position.





1.2 Practical Exercise Question

1. Create a simple marketplace (online shop) site, by applying the concept of SPA (Single Page Application) React Router where several menus are clicked with "Nesting" and "Redirect (Auth)" features.

*****Selamat Mengerjakan*****