Front End Bootcamp
PT. Mahardika Solusi Teknologi

Ariel Arliyanus, S.Kom

Chapter 19 – React JS Authentication & Deployment

Chapter 19
React JS Authentication & Deployment

Authentication adalah proses dimana seorang user (melalui berbagai macam akses fisik berupa komputer , melalui jaringan , atau melalui remote access) mendapatkan hak akses kepada suatu entity (dalam hal ini jaringan suatu corporate)

dari definisi diatas mungkin malah membingungkan padahal kita sering sekali menggunakannya dalam kehidupan sehari-hari. contohnya kita pernah register akun atau login pada suatu web atau app. proses dimana kita masuk ke sebuah sistem dengan validasi login atau register (jika belum mempunyai akun) adalah authentication

lalu seberapa penting authentication itu, jawabannya adalah tergantung aplikasi yang digunakan kita, jika didalam sana terdapat proses editing data dan yang boleh mengedit hanya user tertentu maka authentication sangat di perlukan

nah untuk penerapan dalam reactjs seperti apa, dalam hal ini kita memerlukan packages axios, untuk jenis component bisa pakai function atau class selama menggunakan state management (seperti context)

berikut ini api yang bisa kita gunakan untuk penerapannya:

```
===bisa di akses tanpa token===
GET https://backendexample.sanbersy.com/api/contestants
GET https://backendexample.sanbersy.com/api/contestants
/{ID}

// api yang menghasilkan token

// register untuk user yang belum punya akun
POST https://backendexample.sanbersy.com/api/register
(parameter untuk register name, email dan password)

// login untuk user yang sudah punya akun

POST https://backendexample.sanbersy.com/api/user-login
(parameter untuk login email dan password)

===harus di akses dengan token===
POST https://backendexample.sanbersy.com/api/data-contestant
PUT https://backendexample.sanbersy.com/api/data-contestant/{ID}
DELETE https://backendexample.sanbersy.com/api/data-contestant/{ID}
```

pada daftar api di atas terdapat api yang memerlukan token untuk mendapatkan responsenya, token ini berfungsi untuk validasi apakah user tersebut benar terdapat di sistem dan sudah login

untuk menerapkannya pertama kita buat terlebih dahulu userContext dengan kode seperti di bawah ini:

```
import React, { useState, createContext } from "react";

export const UserContext = createContext();

export const UserProvider = props => {
  const currentUser = JSON.parse(localStorage.getItem("user"))
  const iniateUser = currentUser ? currentUser : null
  const [user, setUser] = useState(iniateUser);

  return (
    <UserContext.Provider value={[user, setUser]}>
      {props.children}
    </UserContext.Provider>
  );
};
```

lalu kita import module UserProvider dan kita langsung digunakan pada app.js seperti di bawah ini:

```
import React from 'react';
import Routes from "./Materi-15/Routes"
import {UserProvider} from "./Materi-Authentication/UserContext"
import './App.css';

function App() {
  return (
    <>
      <UserProvider>
        <Routes/>
      </UserProvider>
    </>
  );
}

export default App;
```

disini kita sudah menginisiasi user lalu langkah berikutnya kita buat component registr dan component login

pada component register kita akan menggunakan api register:

```
import React, { useContext, useState } from "react"
import {UserContext} from "./UserContext"
import axios from "axios"

const Register = () =>{
  const [, setUser] = useContext(UserContext)
  const [input, setInput] = useState({name: "", email: "" , password: ""})

  const handleSubmit = (event) =>{
    event.preventDefault()
    axios.post("https://backendexample.sanbersy.com/api/register", {
      name: input.name,
      email: input.email,
      password: input.password
    }).then(
      (res)=>{
        var user = res.data.user
        var token = res.data.token
        var currentUser = {name: user.name, email: user.email, token }
        setUser(currentUser)
        localStorage.setItem("user", JSON.stringify(currentUser))
      }
    ).catch((err)=>{
      alert(err)
    })
  }

  const handleChange = (event) =>{
    let value = event.target.value
    let name = event.target.name
    switch (name){
      case "name":{
        setInput({...input, name: value})
        break;
      }
      case "email":{
        setInput({...input, email: value})
        break;
      }
      case "password":{
        setInput({...input, password: value})
        break;
      }
      default:{break;}
    }
  }
```

```
return(
    <>
      <div style={{margin: "0 auto", width: "25%", padding: "50px"}}>
        <form onSubmit={handleSubmit}>
          <label>name: </label>
          <input type="text" name="name" onChange={handleChange}
value={input.name}/>
          <br/>
          <label>email: </label>
          <input type="email" name="email" onChange={handleChange}
value={input.email}/>
          <br/>
          <label>Password: </label>
          <input type="password" name="password" onChange={handleChange}
value={input.password}/>
          <br/>
          <button>Register</button>
        </form>
      </div>
    </>
  )
}

export default Register
```

lalu kita menggunakan api login pada component login:

```
import React, { useContext, useState } from "react"
import {UserContext} from "./UserContext"
import axios from "axios"

const Login = () =>{
  const [, setUser] = useContext(UserContext)
  const [input, setInput] = useState({email: "" , password: ""})

  const handleSubmit = (event) =>{
    event.preventDefault()
    axios.post("https://backendexample.sanbersy.com/api/user-login", {
      email: input.email,
      password: input.password
    }).then(
      (res)=>{
        var user = res.data.user
        var token = res.data.token
        var currentUser = {name: user.name, email: user.email, token }
        setUser(currentUser)
        localStorage.setItem("user", JSON.stringify(currentUser))
      }
    ).catch((err)=>{
      alert(err)
    })
  }

  const handleChange = (event) =>{
    let value = event.target.value
    let name = event.target.name
    switch (name){
      case "email":{
        setInput({...input, email: value})
        break;
      }
      case "password":{
        setInput({...input, password: value})
        break;
      }
      default:{break;}
    }
  }
```

```
  return(
    <>
      <div style={{margin: "0 auto", width: "25%", padding: "50px"}}>
        <form onSubmit={handleSubmit}>
          <label>Email: </label>
          <input type="email" name="email" onChange={handleChange}
value={input.email}/>
          <br/>
          <label>Password: </label>
          <input type="password" name="password" onChange={handleChange}
value={input.password}/>
          <br/>
          <button>Login</button>
        </form>
      </div>
    </>
  )
}

export default Login
```

lalu untuk menggunakan tokennya kita bisa coba pada api data-contestant dan contoh penerapannya seperti di bawah ini:

```
      axios.post(`https://backendexample.sanbersy.com/api/data-contestant`,
{name: input.name}, {headers: {"Authorization" : "Bearer "+ user.token}})
      .then(res => {
        var data = res.data
        setDataPeserta([...dataPeserta, {id: data.id, name: data.name}])
        setInput({id: null, name: ""})
      })
```

untuk penerapannya seperti itu dan penggunaan token pada kode axios diatas berguna untuk validasi apakah user sudah login, atau loginnya sudah expired sehingga kita dapat handle kemungkinan tersebut

Referensi:

https://medium.com/@fahmiprasetiiio/authentication-dan-authorization-afa2029ff876

React JS Deployment

didalam membangun suatu aplikasi atau web app tentunya ada masa dimana kita akan deployment aplikasi kita, biasanya dalam deployment kita membuat dua server staging dan production, staging di gunakan untuk testing sebelum nantinya aplikasi di gunakan di production yang tentunya akan di gunakan user

pada materi kali ini kita akan mencoba deployment project kita dalam keperluan staging, sebenarnya banyak opsi untuk membuat aplikasi kita bisa diakses orang lain, kita bisa menggunakan Cloud, VPS, hosting, SaaS (Software as a Service) dll.

pada materi ini kita akan melakukan deployment ke netlify sebagai SaaS. sebelum lanjut cara ke deployment, silahkann anda untuk mengakses netlify.com dan login ke akun netlify anda, jika belum punya silahkan register akun terlebih dahulu

untuk deployment dengan netlify terdapat dua cara yaitu:

Drag & Drop

cara pertama adalah dengan drag folder build dari project reactjs ke sites di netlify. oleh karena itu kita jalankan perintah ini untuk build project reactjs

```
npm run build
```

tunggulah proses tersebut hingga selesai

lalu drag folder build tersebut ke sites di netlify anda

Netlify CLI

cara kedua adalah menggunakan netlify cli sama seperti sebelumnya build reactjs terlebih dahulu

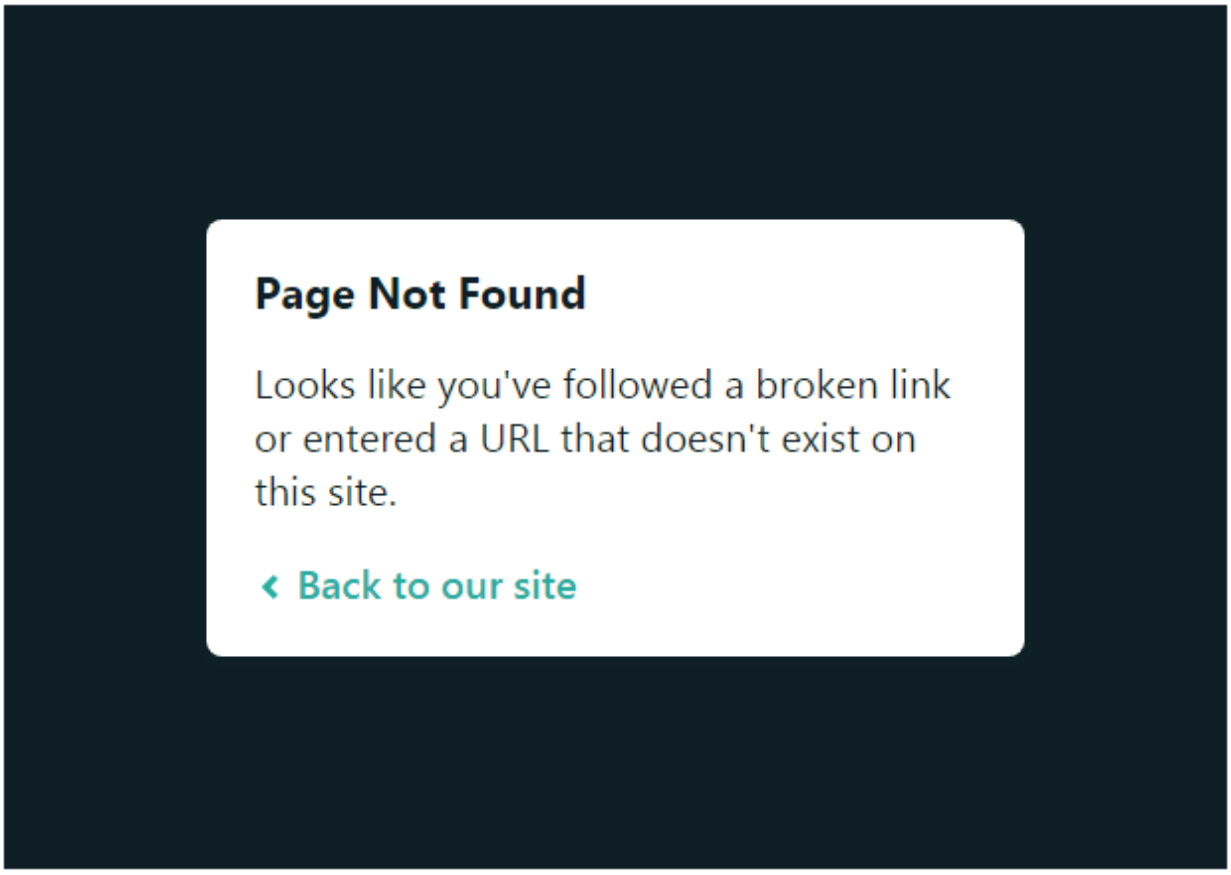
```
npm run build
```
lalu install netlify cli dengan perintah di bawah ini

```
npm install netlify-cli -g
```
setelah itu jalankan perintah deploy seperti di bawah ini

```
netlify deploy
```

Not Found Issue

setelah berhasil deploy biasanya terdapat masalah ***not found*** dimana misal ada route `/movie`, ketika di akses route otomatis akan ada tampilan seperti di bawah ini:



cara mengatasinya adalah pertama adalah buatlah sebuah file bernama ***_redirects*** di dalam folder public

di dalam file tersebut masukkan kode `/* /index.html 200`

setelah itu lakukan build dan deploy seperti sebelumnya

Referensi:

- https://www.freecodecamp.org/news/how-to-deploy-a-react-application-to-netlify-363b8a98a985/
- https://www.netlify.com/blog/2016/07/22/deploy-react-apps-in-less-than-30-seconds/
- https://dev.to/rajeshroyal/page-not-found-error-on-netlify-reactjs-react-router-solved-43oa