Front End Bootcamp
PT. Mahardika Solusi Teknologi

Ariel Arliyanus, S.Kom

Chapter 13 – React JS

Chapter 13
React JS

Apa itu ReactJS?

ReactJS atau **React** adalah **Javascript library** yang digunakan untuk membangun **user interface** (antar muka), jadi react itu adalah sebuah library bukan sebuah framework

reactjs sendiri di kembangkan oleh facebook. react dibuat oleh Jordan Walke, seorang software engineer di Facebook, yang merilis prototipe awal React yang disebut “FaxJS “.

berikut ini hal-hal yang penting dalam reactjs:

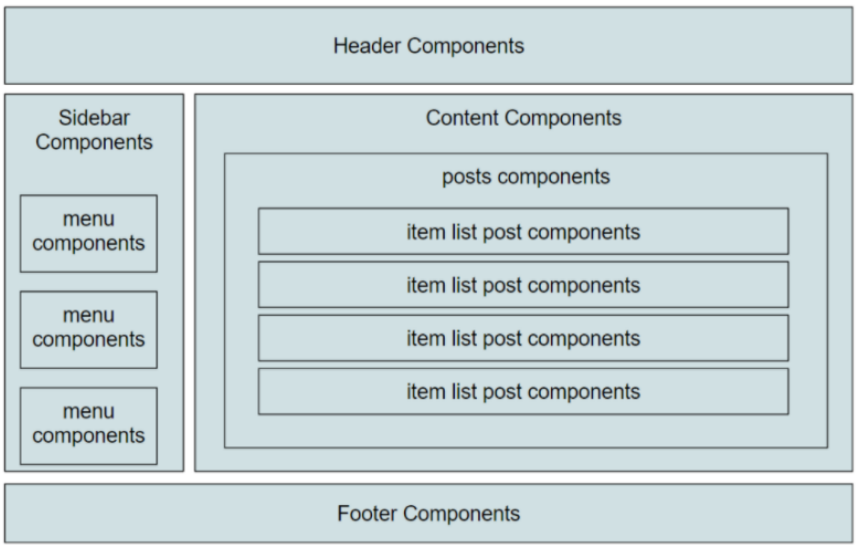
1. Components

Components bisa di bilang hal yang sangat penting di reactjs. Components adalah semacam konsep bagaimana cara untuk membuat tiap bagian dari UI (tampilan) bisa bekerja secara mandiri dan terisolasi dari bagian-bagian yang lain.

Sebenarnya secara sadar atau tidak sadar bagi yang sering bekerja dengan kode HTML, selalu melakukan pemisahan bagian dari halaman-halaman misalnya *Header*, *Content*, dan *Footer*. Biasanya tujuannya agar kode tidak menumpuk di satu halaman sehingga di pisahkan ke file-file kecil yang lebih spesifik fungsinya dan kode lebih *maintainable* jika ada perbaikan.

Khusus di reactjs bahkan hampir bagian terkecil di tampilan disarankan untuk dipisahkan. components ini memiliki sifat reusable yang membuat developer tidak perlu menyalin halaman atau komponen yang sama berulang-ulang

berikut ini ilustrasi components di reactjs:



2. JSX

JSX merupakan ekstensi javascript yang membuat kita dapat menuliskan tag HTML di dalam javascript.

```
const element = <h1>Hello, world!</h1>;
```

Pada reactjs ini kita akan JSX dan mungkin ada sedikit perbedaan di dalamnya, tetapi untuk yang sudah memahami html ini tidak akan terlihat sulit.

3. Virtual DOM

Jika mulai mendalami React, akan ada pertanyaan apa itu *virtual* DOM? *Virtual* DOM adalah representasi DOM secara *virtual*.

Kenapa React menggunakan *virtual* DOM? JavaScript itu sebenarnya cepat. Yang membuat JavaScript terasa lambat adalah ketika JavaScript mengolah DOM. React membuat *virtual* DOM untuk mempercepat urusan tersebut.

React melakukan semua operasi di dalam *virtual* DOM. Setelah operasi tersebut selesai, React menulis perubahan tersebut di DOM.

Proses di React jika ada perubahan suatu elemen di dalam DOM:

1. React sudah mempunyai representasi dari DOM di virtual DOM.

2. React menerima representasi DOM yang berisi perubahan.

3. React membandingkan perbedaan kedua representasi lama dan baru.

4. Hasil dari perbandingan tersebut dimasukkan ke antrian.

5. Terakhir React akan me-*render* ulang *patch* tersebut ke DOM

Analoginya begini, saya punya pensil dan spidol. Ketika saya menulis di kertas menggunakan spidol dan melakukan kesalahan (*typo*) di tengah paragraf maka saya harus mengganti dengan kertas yang baru dan menulis lagi dari awal.

Namun jika saya menggunakan pensil, saya bisa menghapus kesalahan tersebut dengan penghapus tanpa mengulang lagi dari awal.

4. States & Lifecycle

State adalah asal dari suatu data. Komponen pada react tentunya membutuhkan data (tidak semua namun rata-rata membutuhkan data). Data tersebut dapat berasal dari mana saja. State adalah salah satu sumber data tersebut. Selain state data dari komponen juga dapat berasal dari props (bukan property).

Lifecycle dalam sebuah komponen dalam react, terdapat tiga lifecycle berikut: Inisialisasi / mounting (ketika komponen dibuat/ditambahkan pertama kali pada DOM)

Update / rerender (ketika terdapat perubahan state/prop yang mengakibatkan perubahan pada DOM)

Unmounting (ketika komponen akan dihapus dari DOM)

Pada setiap lifecycle tersebut, komponen react akan mengeksekusi method/fungsi yang berbeda yang kita sebut sebagai lifecycle methods. Beberapa method mempunyai prefix will dan did yang menunjukkan kapan method tersebut akan dieksekusi.

Installasi dan setup ReactJS

sebelum ke installasi dan setup ada beberapa requirements atau prasyaratnya yaitu:

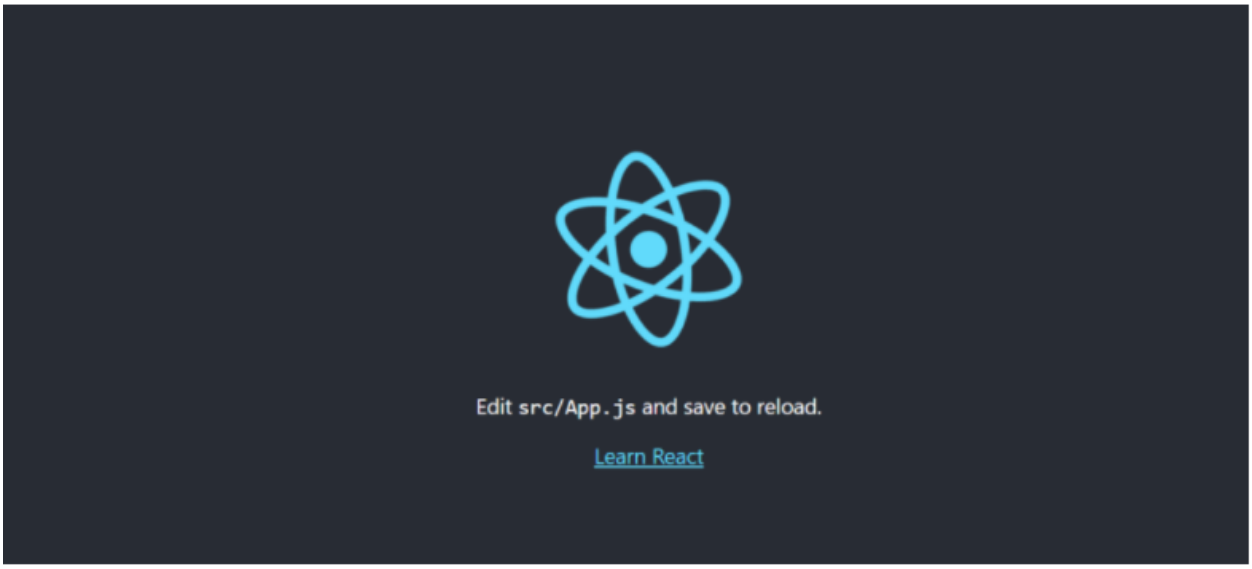
- nodeJS (versi LTS terbaru)
- text editor (visual studio code, sublime, atom dll)
- terminal (bisa cmd, git bash atau terminal di text editor)
- berikut ini step installasi dan setup ReactJS:

1. installasi

- pertama masuk ke terminal lalu pindah ke folder yang diinginkan untuk installasi project reactjs, lalu jalankan perintah seperti di bawah ini
- `npx create-react-app my-app`
- untuk text “my-app” dapat di ganti sesuai dengan nama project yang kita inginkan

2. Menjalankan project reactjs yang sudah di install

- pada terminal pindahkan posisinya menuju folder my-app (jika nama appnya berbeda bisa di sesuaikan dengan). lalu ketikkan perintah
- `npm start`



Ouput Halaman Landing React JS

Coding halaman tersebut dapat dilihat di file App.js

```
import React from 'react';
import logo from './logo.svg';
import './App.css';

function App() {
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo" alt="logo" />
        <p>
          Edit <code>src/App.js</code> and save to reload.
        </p>
        <a
          className="App-link"
          href="https://reactjs.org"
          target="_blank"
          rel="noopener noreferrer"
        >
          Learn React
        </a>
      </header>
    </div>
  );
}

export default App;
```

Components dan Props

Components memungkinkan Anda membagi UI menjadi bagian-bagian yang independen dan dapat digunakan kembali, dan anda bisa mengaturnya setiap bagian secara terpisah

sedangkan props merupakan properti-properti dari component yang dapat kita gunakan untuk menampilkan data, biasanya digunakan untuk inisialisasi data

Secara konseptual, komponen seperti function JavaScript. Mereka menerima input dan mengembalikan React Element yang akan muncul di layer

Function Components dan Class Components

untuk mendefinisikan component terdapat dua cara yaitu dengan menggunakan function dan class

berikut ini contoh component yang menggunakan function

```
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}
```

function di atas ialah react component yang valid karena function itu menerima argumen “props” (singkatan dari properties) tunggal dengan data dan akan memberikan return ke React Element. kita bisa menyebut function tersebut dengan “Function Component” karna secara harfiah itu adalah function JavaScript.

dan kita juga bisa menggunakan ES6 class untuk mebangun component. berikut ini contoh component dengan menggunakan class React.Component:

```
class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}
```

Menyusun Components

Component dapat merujuk ke component lain dalam outputnya. Ini memungkinkan kita bisa menggunakan component yang sama secara berulang-ulang (reusable).

berikui ini contoh untuk cara menyusun components (asumsi ini di folder app.js)

```
import React from 'react';
import './App.css';

class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}

function App() {
  return (
    <div>
      <Welcome name="Sarah" />
      <Welcome name="Michael" />
      <Welcome name="John" />
    </div>
  );
}

export default App;
```

Mengolah data dan mengekstrak component

mungkin timbul pertanyaan bagaimana cara mengolah data dan di munculkan ke component, lalu bagaimana jika ingin menggunakan component yang berbeda-beda dan di gunakan gabungkan satu tempat. mari kita simak contoh di bawah ini

misal ada data seperti di bawah ini:

```
var person = [
  {name: "sarah", age: 25},
  {name: "michael", age: 30},
  {name: "john", age: 33}
]
```

lalu ada dua component seperti di bawah ini:

welcome component

```
class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}
```

age component

```
class Age extends React.Component {
  render() {
    return <h1>your age is {this.props.age}</h1>;
  }
}
```

untuk mengolah data person di atas dan menggabungkan dua component tersebut kita bisa membuat component baru yang melakukan dua hal tersebut.

berikut ini contoh pengolahan data dan penggabungan dua component tersebut:

```
var person = [
  {name: "sarah", age: 25},
  {name: "michael", age: 30},
  {name: "john", age: 33}
]

class UserInfo extends React.Component {
  render() {
    return (
      <>
        {person.map(el=> {
          return (
            <div style={{border: "1px solid #000", padding: "20px"}}>
              <Welcome name={el.name}/>
              <ShowAge age={el.age}/>
            </div>
          )
        })}
      </>
    )
  }
}
```

lalu penerapannya dalam file app.js adalah seperti berikut ini:

```
import React from 'react';
import './App.css';

class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}

class ShowAge extends React.Component {
  render() {
    return <h1>your age is {this.props.age}</h1>;
  }
}
```

```
var person = [
  {name: "sarah", age: 25},
  {name: "michael", age: 30},
  {name: "john", age: 33}
]



class UserInfo extends React.Component {
  render() {
    return (
      <>
        {person.map(el=> {
          return (
            <div style={{border: "1px solid #000", padding: "20px"}}>
              <Welcome name={el.name}/>
              <ShowAge age={el.age}/>
            </div>
          )
        })}
      </>
    )
  }
}

function App() {
  return (
    <div>
      <UserInfo />
    </div>
  );
}

export default App;
```

import dan export module

seperti yang dilihat diatas maka file app.js akan terlalu penuh, lalu apakah kita bisa membuat file tersendiri untuk UserInfo misal, jawabannya adalah iya

buatlah file UserInfo.js di dalam folder src lalu masukkan kode di bawah ini:

```
import React from 'react';

class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}

class ShowAge extends React.Component {
  render() {
    return <h1>your age is {this.props.age}</h1>;
  }
}

var person = [
  {name: "sarah", age: 25},
  {name: "michael", age: 30},
  {name: "john", age: 33}
]

class UserInfo extends React.Component {
  render() {
    return (
      <>
        {person.map(el=> {
          return (
            <div style={{border: "1px solid #000", padding: "20px"}}>
              <Welcome name={el.name}/>
              <ShowAge age={el.age}/>
            </div>
          )
        })}
      </>
    )
  }
}

export default UserInfo
```

lalu untuk import UserInfo di app.js bisa adalah seperti di bawah ini:

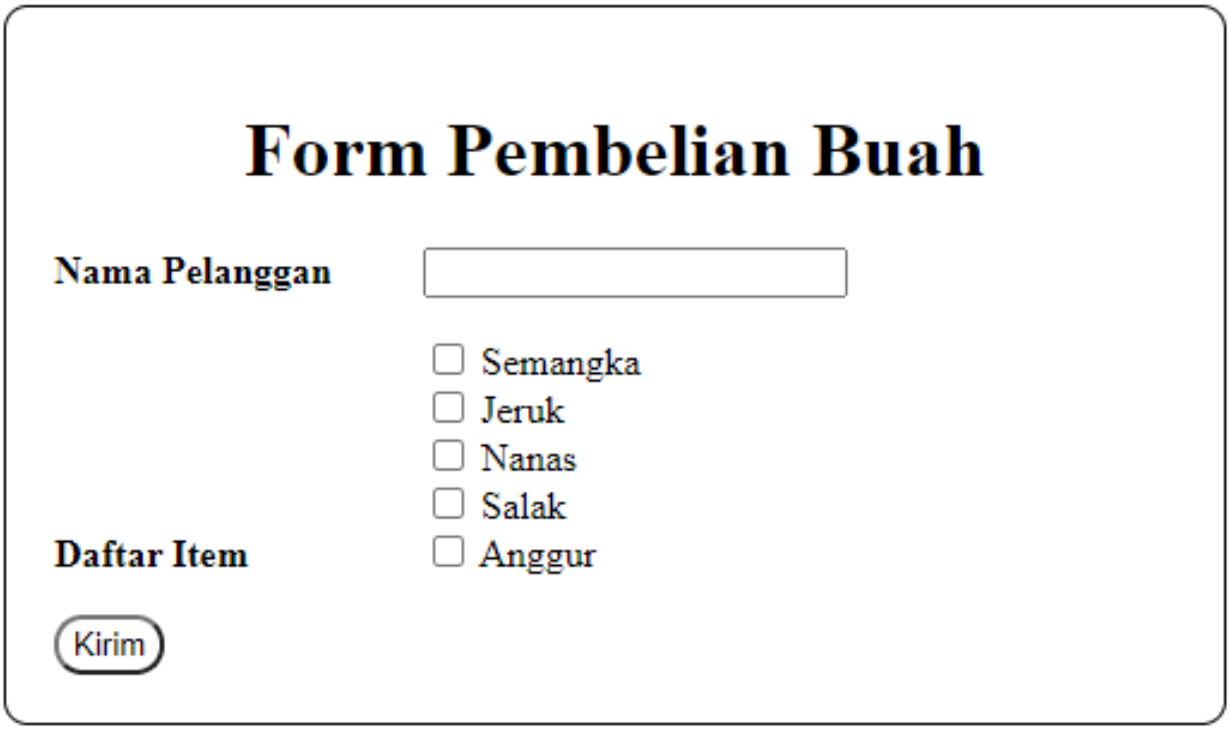
```
import React from 'react';
import './App.css';
import UserInfo from './UserInfo';

function App() {
  return (
    <div>
      <UserInfo />
    </div>
  );
}

export default App;
```

Kerjakan Soal di bawah ini

1. Ubahlah tampilan pada app.js di dalam folder src menjadi seperti tampilan di bawah ini (stylingnya harus sama persis):



pada tugas ini **tidak di perbolehkan** menggunakan bootstrap ataupun sejenisnya

2. Tambahkan table di bawah soal 1, dengan data dataHargaBuah.

Tabel Harga Buah

Nama	Harga	Berat
Semangka	10000	1 kg
Anggur	40000	0.5 kg
Strawberry	30000	0.4 kg
Jeruk	30000	1 kg
Mangga	30000	0.5 kg

```
let dataHargaBuah = [
  {nama: "Semangka", harga: 10000, berat: 1000},
  {nama: "Anggur", harga: 40000, berat: 500},
  {nama: "Strawberry", harga: 30000, berat: 400},
  {nama: "Jeruk", harga: 30000, berat: 1000},
  {nama: "Mangga", harga: 30000, berat: 500}
]
```