



ReactJs

Gold - Chapter 7 - Topic 1

**Selamat datang di Chapter 7 Topik 1 online
course Fullstack Web dari Binar Academy!**



Ke Pasar Minggu bersama pacar baru, selamat datang di Chapter 7!



Sebagai pembuka course, chapter 7 bakal ngajak kamu buat **membuat sebuah aplikasi dengan menggunakan ReactJS dan juga menerapkan OAuth di dalam React dan Express.**, mulai dari pengenalan ReactJs, React Router, Backend Integration, Oauth, dan Redux.

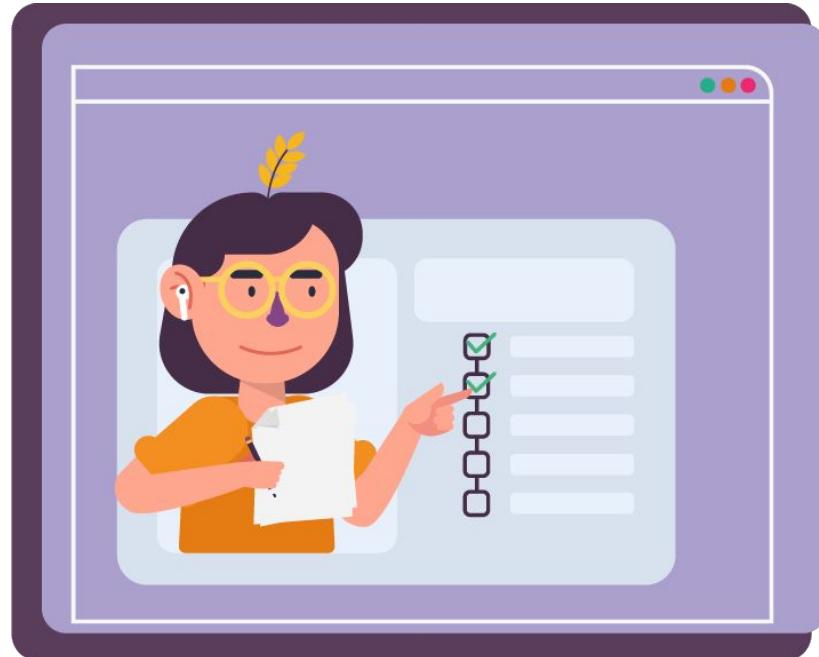
Di topik pertama ini, kita bakal bahas dulu tentang **apa itu ReactJs, beserta penggunaannya**.

Letsgowww~



Detailnya, kita bakal bahas hal-hal berikut ini:

- Pemahaman dasar ReactJs
- Component, state, dan property dalam ReactJs
- Konsep dan cara melakukan styling
- Mengenal UI framework dan cara menggunakan





React tuh kira-kira akan ngebahas apa ya? Yaudah deh, pengertiannya dulu kali ya~

React adalah sebuah library yang digunakan untuk membuat User Interface. Bisa UI dari website, aplikasi mobile, maupun desktop.

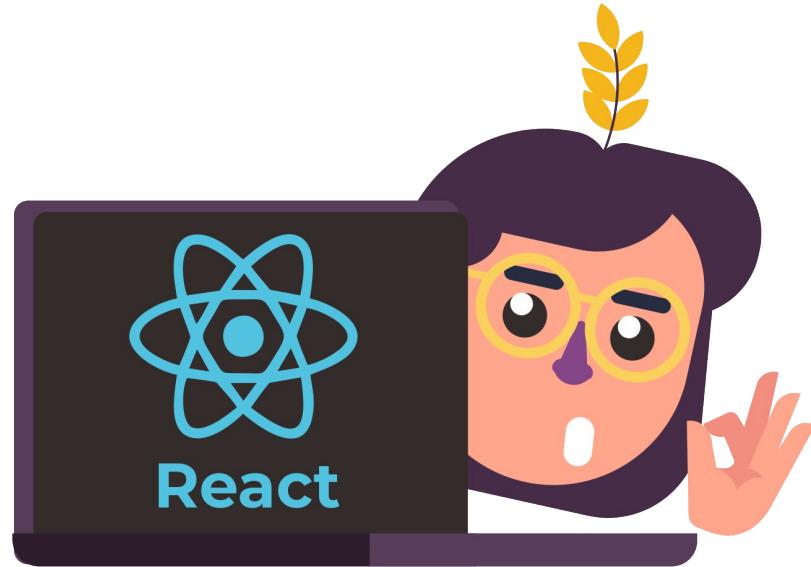


Kenalan sama React yuk~

React ini dibuat dan dikembangkan sama Facebook (ehem, sorry. Meta maksudnya) dengan tujuan **mempermudah pembuatan UI dari sebuah aplikasi.**

Tadi kan udah disebutin kalo aplikasi tuh ada banyak jenisnya, pada bootcamp ini kita hanya akan mempelajari React untuk membuat aplikasi web ya, karena kita kan lagi belajar Full-stack Web Engineering.

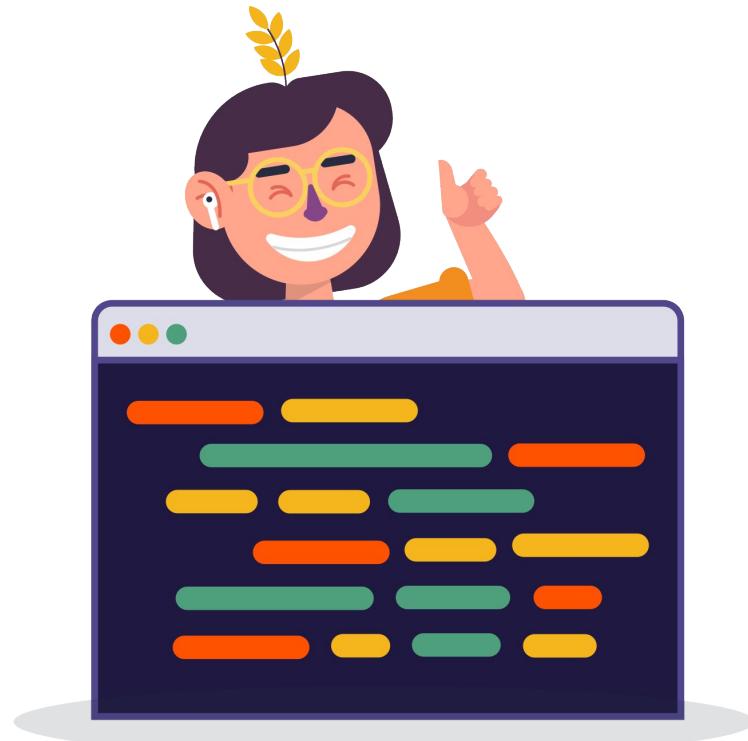
Nah, untuk memahami dan menguasai React untuk mengembangkan sebuah aplikasi web, kita perlu tahu dulu nih tentang react dan apa sih yang bisa dia lakuin, yuk simak pembahasan berikut~





Single Page Application (SPA)

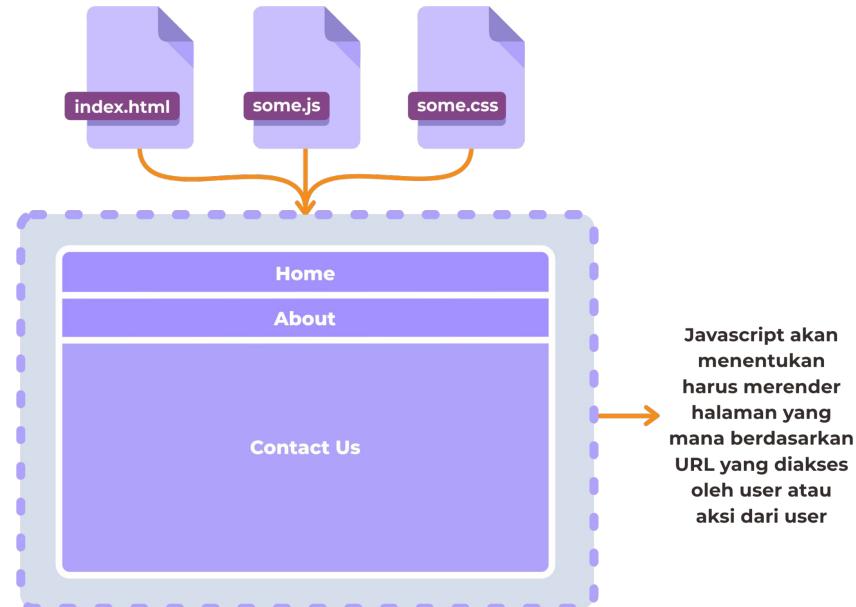
By default react adalah single page application. Maksudnya adalah, **ketika kita membuka sebuah website yang dibuat menggunakan ReactJS, maka website tersebut hanya memiliki 1 (satu) halaman.**



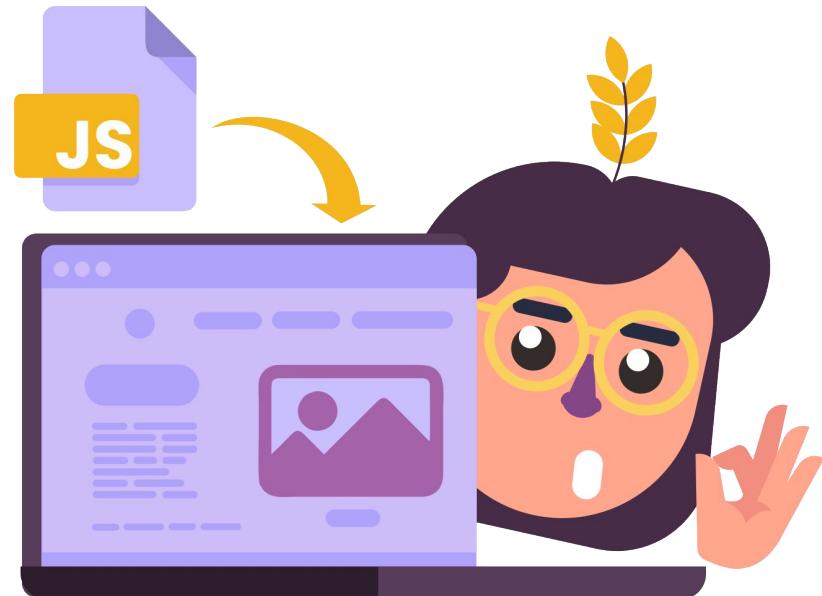


Eitsss, single page application bukan berarti aplikasi tersebut hanya memiliki 1 (satu) halaman secara harfiah. Yang dimaksud dengan 1 (satu) halaman adalah **aplikasi tersebut hanya menggunakan 1 (satu) file HTML saja untuk menampilkan semua halaman.**

Jadi, ketika kamu membuka Single Page Application, kamu hanya mendownload 1 (satu) file HTML, beberapa file CSS dan beberapa file Javascript. Dimana ketiga file tersebut akan berguna dalam mendefinisikan UI dari sebuah Website



Javascript berperan paling penting disini, jadi hampir tidak mungkin Single Page Application ditulis tanpa menggunakan Javascript. Karena **Javascript berperan untuk memanipulasi tampilan dari sebuah website**. Jadi ketika kita mengakses halaman lain, Javascript akan mengolah HTML menjadi halaman lain tanpa mengganti file HTML yang kita download.

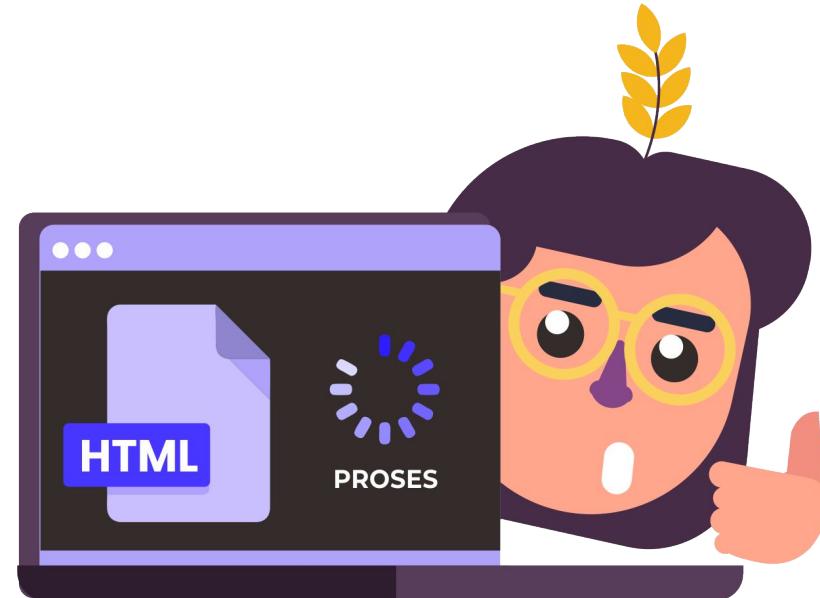




Client Side Application

Nah, selain Single Page Application, by default react menggunakan metode Client Side Rendering. Hmmm apa tuuuuh?

Artinya **HTML akan diolah di device user**. Jadi server ga perlu pusing mikirin tampilan HTML-nya akan kaya gimana, karena semua akan diurus oleh aplikasi react yang jalan di HP atau Laptop masing-masing user.





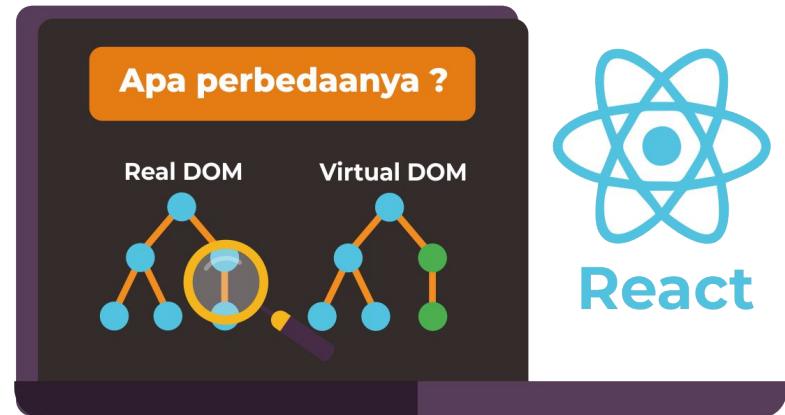
Ada satu hal lagi yang perlu kamu ketahui dalam konsep Client Side Rendering dalam react. **React tidak langsung manipulasi DOM di browser untuk membuat UI, tapi dia manipulasi Virtual DOM dulu.** Kok gitu?

Jadi gini temen-temen, ketika kamu memanipulasi DOM di browser, maka browser akan merender ulang tampilan yang ada di dalam viewport browser, nah proses ini memakan resource di dalam komputer, dan jauh lebih lambat apabila banyak object yang dimanipulasi.





Maka dari itu, React menggunakan yang namanya Virtual DOM untuk mencegah browser merender ulang tampilan di dalam viewport ketika tidak diperlukan. React akan cek terlebih dahulu apakah ada perbedaan antara Virtual DOM dengan DOM nya. Jika dilihat ada perbedaan, maka React akan melakukan manipulasi DOM, agar bisa sesuai dengan Virtual DOM. Tapi kalau sudah sesuai, ya tentu ga perlu render ulang.





Syntax JSX

Masih inget cara memanipulasi HTML pake DOM? Apa yang kamu ingat tentang hal tersebut? Mungkin kita bisa menyetujui satu hal tentang manipulasi HTML pake DOM, yaitu ribet.

Coba cek kode di samping, kita mencoba membuat sebuah tag div yang berisi h1 dengan konten "Hello World"

Ribet banget kan? Kita perlu inget method-method dari document, dan juga method-method dari node. Nggak dulu deh, kalo disuruh ngapal-ngapalin lagi 🤦



```
const h1 = document.createElement("h1");
h1.textContent = "Hello World";

const div = document.createElement("div");
div.appendChild(h1);
```



Nah, coba bandingin dengan kode di samping ini yang fungsinya sama persis kayak kode di atas.

Simpel dan mudah dibaca bukan? Kok bisa kita tulis HTML langsung di dalam Javascript? Jawabannya adalah **karena kita menggunakan JSX**.

Jadi, di dalam react ketika kita mau mendefinisikan UI berupa elemen HTML, ataupun React Component, kita menggunakan JSX. **JSX adalah sintaks yang membantu kita dalam mendefinisikan tampilan HTML di dalam aplikasi React.**

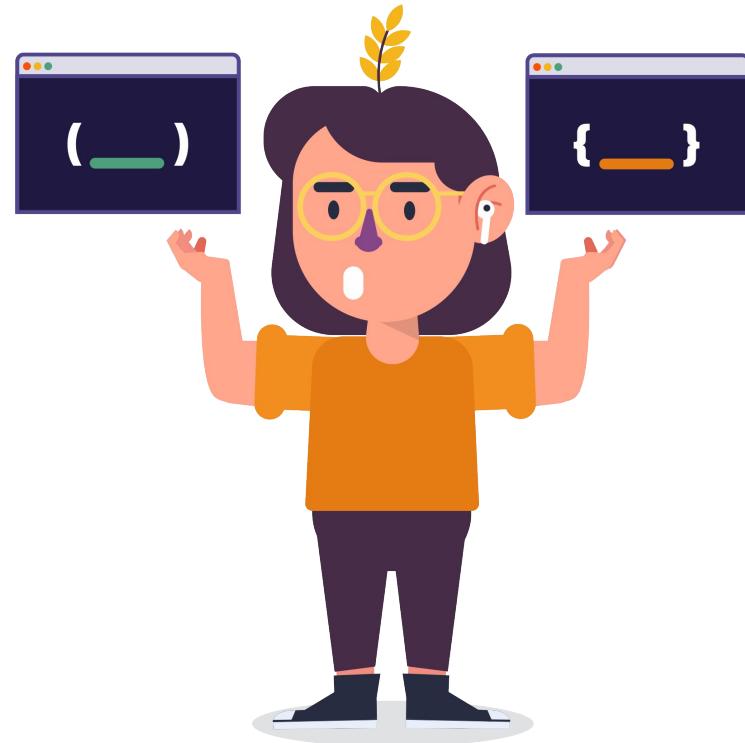


```
const msg = "Hello World";  
const div = (  
  <div>  
    <h1>{msg}</h1>  
  </div>  
);
```



JSX ini ga susah kok buat dipelajaran, berikut **hal-hal yang perlu kamu ingat ketika pake JSX :**

- Ketika menulis HTML di dalam JSX dan HTML tersebut memiliki lebih dari 2 line, selalu bungkus elemen tersebut menggunakan kurung buka dan kurung tutup seperti contoh di atas.
- Untuk menaruh sebuah nilai dalam suatu variabel di dalam sebuah HTML yang ditulis di dalam JSX, gunakan bungkus variabel tersebut dengan menggunakan kurung kurawal, seperti contoh di atas.
- Didalam kurung kurawal adalah ekspresi Javascript yang harus mengembalikan nilai truthy.
- Kamu dapat menuliskan loop statement di dalam JSX.





- Selalu gunakan camelCase dalam penamaan properti di dalam JSX.
- Ketika menambahkan class dalam sebuah element HTML, kamu hanya boleh mendefinisikannya dengan menggunakan atribut bernama className, hal ini dikarenakan class adalah reserved keyword dari Javascript untuk membuat class.
- Setiap ekspresi JSX hanya boleh mengembalikan 1 elemen saja (elemen tersebut dapat memiliki anak elemen) seperti contoh diatas. Namun apabila sangat diperlukan untuk mengembalikan dua elemen sekaligus dalam 1 level yang sama, gunakanlah Fragment.



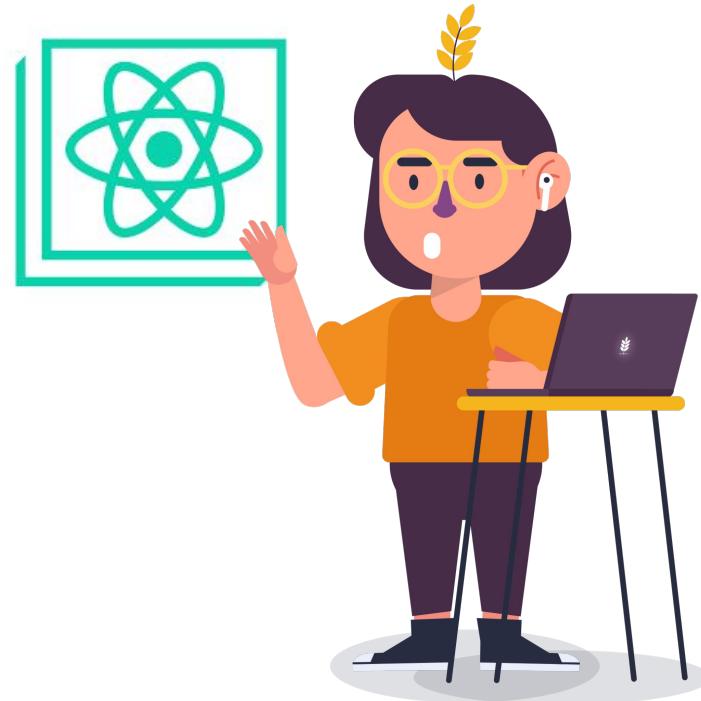


React Development Setup

Dengan membahas topik ini kita akan paham toolchain yang tersedia di dalam React yang digunakan untuk membantu proses pengembangan aplikasi react.

CRA

Nah, untuk membuat aplikasi react, biasanya kebanyakan orang **menggunakan tools yaitu create-react-app (CRA)**. CRA dipakai untuk mengenerate atau bootstrapping aplikasi react. Artinya ketika kita pake tools semacam CRA, kita akan punya 1 project node.js dengan react yang sudah disetup secara otomatis. Jadi ga perlu lagi mikir webpack-webpackan lagi deh hehe..





Seperti yang sudah dibahas tadi, CRA ini membantu kita dalam membootstrap aplikasi react untuk kita kembangkan. Jadi pengembangan yang kita lakukan ga perlu dari nol banget setup-nya, dan tinggal sat set set set.

Untuk menggunakan create-react-app, kita bisa menggunakan npx atau yarn. Wait, npx tuh benda apaan ya?

Jadi, sama kayak npm, npx ini dipakai **untuk menginstal package dan langsung mengeksekusi program dari package tersebut**. Karena create-react-app ini bukan sebuah dependency dari suatu aplikasi, melainkan ia adalah tools yang berupa command line interface, maka dari itu kita menggunakan npx, bukan menggunakan npm.

Kalo kamu pengen kepoin CRA lebih lanjut, kamu bisa buka link ini : [Create React App](#)





CRA dengan NPX

Untuk menjalankan CRA dengan npx, kamu cuma perlu execute perintah di samping ini aja. Ketika kamu ngejalankan perintah tersebut, akan ada folder baru bernama "nama-project-mu" dan di dalam folder tersebut akan terdapat project node.js yang sudah disetup dengan react, jadi kamu tinggal modifikasi apa yang ada di dalamnya saja.



```
npx create-react-app nama-project-mu
```



Dan struktur foldernya akan seperti di samping ini.



```
.
├── package.json
├── package-lock.json
├── public
│   ├── favicon.ico
│   ├── index.html
│   ├── logo192.png
│   ├── logo512.png
│   └── manifest.json
└── robots.txt
├── README.md
└── src
    ├── App.css
    ├── App.js
    ├── App.test.js
    ├── index.css
    ├── index.js
    ├── logo.svg
    ├── reportWebVitals.js
    └── setupTests.js

2 directories, 17 files
```



Dan jika kalian menjalankan perintah `npm start`, dan membuka `localhost:3000` di dalam maka tampilannya akan gambar di samping ini. Nanti kamu tinggal edit file `src/App.js` saja.



Edit `src/App.js` and save to reload.

[Learn React](#)



CRA dengan yarn

Pake yarn juga ga jauh beda dengan npx, perintahnya jadi kayak gambar di samping. Langkah-langkah berikutnya sama kok hehehe, cuma bedanya **diawal kalian harus pake yarn.**



```
yarn create react-app nama-project-mu
```



Vite (Mutually Exclusive with CRA)

Vite ini merupakan build tools yang dipakai untuk mengembangkan aplikasi frontend dan framework lain, ga cuma react aja.

Nah, karena kita belajarnya react, maka dari itu ya otomatis kita pake vite buat ngegenerate project react.

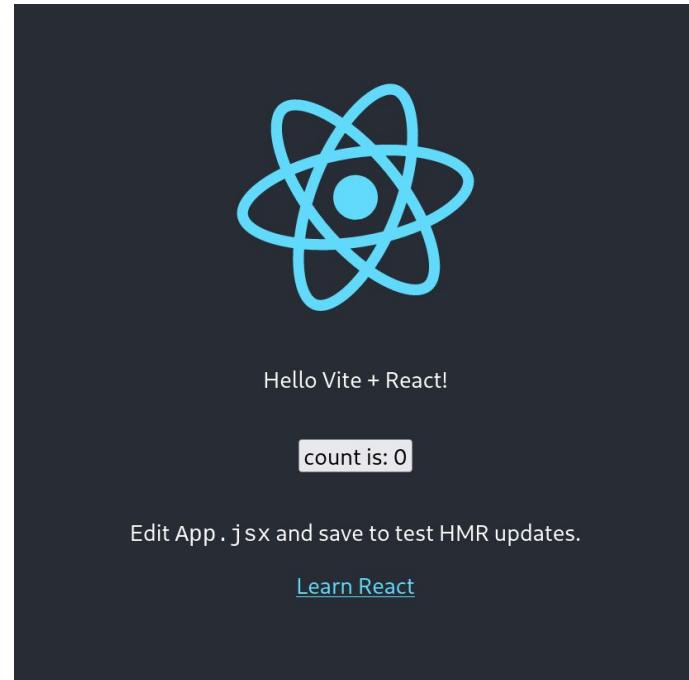
Untuk membuat projectnya bisa menggunakan perintah seperti gambar di samping.



```
yarn create vite-app nama-react-project --template react
```



Dan jika kamu menjalankan perintah `npm run dev`, dan membuka `localhost:3000` di dalam maka tampilannya akan gambar di samping ini. Nanti kamu tinggal edit file `src/App.jsx` aja.





**Sob kalian sebagai React Developer
wajib paham ini nih~**

**Yup, Component, State, dan
Property** adalah hal esensial yang
perlu kamu ketahui tentang react.
Jangan pernah ngaku kalo kamu
seorang React Developer kalo gatau
terms ini hehehe.

Kalau gitu kita langsung bahas yuk!





Kita mulai dari definisi hal esensial dalam React ya~

- **Component** : Anggep aja ini element di dalam HTML, mendefinisikan view dari sebuah aplikasi
- **Property** : Atribut dari sebuah component
- **State** : Data dari sebuah component. Biasanya ditaruh di komponen global

- A **Component**
- B **Property**
- C **State**





Component

Component itu seperti **elemen di dalam HTML**, namun elemen ini lebih advance dan lebih singkat dibandingkan kita menulis plain HTML elemen. Sebagai contoh, ketika kita pake bootstrap, untuk membuat card, kita akan tulis code-nya seperti ini :

```
<div class="card" style="width: 18rem;">
  
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">Some quick example
text to build on the card title and make up
the bulk of the card's content.</p>
    <a href="#" class="btn btn-primary">Go
somewhere</a>
  </div>
</div>
```



Dan, jika kita membuat React Component, kita bisa menyederhanakan code diatas menjadi seperti ini.

Lebih deklaratif bukan?

```
<Card  
  title="Hello"  
  description="Lorem ipsum dolor sit amet"  
  btnText="Go Somewhere"  
  btnHref="https://google.com"  
  imgSrc="https://placeimg.com/640/480/any"  
  imgAlt="Hello"  
/>
```



Ada dua cara dalam membuat component, yaitu dengan menggunakan Class Component, atau Functional Component. Kita ga bakal bahas cara membuat component dengan Class Component, karena itu sudah kuno dan ga jaman. Sesi ini kita hanya bakal bahas yang Functional Component aja.

Lalu, di dalam src/App.jsx, kita tinggal impor dan gunakan component tersebut layaknya JSX. Source Code dari code diatas dapat dilihat di [link berikut](#)



```
import React from "react";

const Card = (props) => {
  const { title, description, imgSrc, imgAlt, btnText,
  btnHref } = props;

  return (
    <div className="card" style="width: 18rem;">
      <img src={imgSrc} className="card-img-top" alt={imgAlt} />
      <div className="card-body">
        <h5 className="card-title">{title}</h5>
        <p className="card-text">{description}</p>
        <a href={btnHref} className="btn btn-primary">
          {btnText}</a>
      </div>
    </div>
  )
}

export default Card;
```



State

State adalah **bagian dari component, yang berfungsi untuk menyimpan dan mengubah data.** Contoh component pada bagian sebelumnya adalah stateless component, artinya adalah component tanpa state.

Bagaimana jika kita ingin membuat sebuah game sederhana yang digunakan untuk menghitung berapa kali seorang user dapat click dalam satu waktu?

Kita perlu membuat stateful component yang digunakan untuk menyimpan informasi berapa kali user click dan sebagainya.





Untuk membuat stateful component, kita perlu react hooks yang bernama useState. **Fungsinya untuk mendefinisikan suatu state di dalam react component.**

Fungsi ini meminta satu parameter, yaitu default value dari state tersebut. Anggap saja state adalah sebuah variable yang mana ketika kita ingin mengubahnya, kita perlu menggunakan suatu fungsi agar react melakukan update pada UI-nya.

Nah useState ini mengembalikan arra. Array tersebut di index pertamanya akan bernilai state itu sendiri (yang digunakan pada UI), dan index keduanya bernilai fungsi yang digunakan untuk mengubah nilai state. Lihat contoh berikut.

```
● ● ●

import { useState } from "react"
import logo from './logo.svg';
import './App.css';

function App() {
  const [count, setCount] = useState(0);

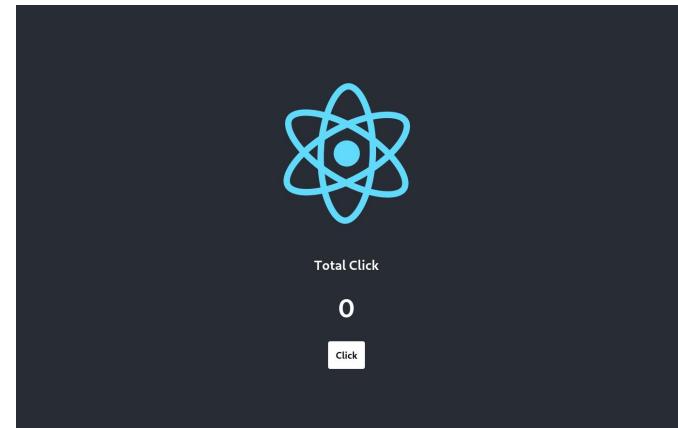
  return (
    <div className="App">
      <header className="App-header">
        <img src={logo} className="App-logo"
          alt="logo" />

        <div>
          <h4>Total Click</h4>
          <h1>{count}</h1>
          <button className="App-button" onClick={() => setCount(count + 1)}>Click</button>
        </div>
      </header>
    </div>
  );
}

export default App;
```



Pada code di atas terdapat dua variable count, dan setCount, dua variable ini adalah output dari useState hook. Variable count akan bernilai 0 by default berdasarkan cara kita mendefinikan state, dan dia akan berubah berdasarkan click yang terjadi di dalam button yang ada pada component tersebut.





Property

Nah ngomong-ngomong soal property, setiap **komponen yang kita buat itu bisa dipakai untuk menyimpan property.**

Gunanya buat apa? Entahlah, bisa aja dipakai untuk mendefinisikan UI dari sebuah component, atau konfigurasi dari sebuah component itu sendiri, macam-macam pokoknya.

Believe it or not, kamu udah mencoba membuat component dengan property di contoh sebelumnya lho! Yuk kita coba buat lagi, kali ini kita buat property untuk mendefinisikan warna dari sebuah button ya~

```
● ● ●

import React from 'react';
import PropTypes from "prop-types"

const backgroundColor = { primary: "#61DBFB",
secondary: "#fffff", black: "#000000" }
const textColor = { primary: "#000000", secondary:
"#000000", black: "#fffff" }

const Button = ({ children, variant, onClick,
className, ...props }) => {
  const styles = {
    backgroundColor: backgroundColor[variant] ||
backgroundColor.primary,
    color: textColor[variant] || textColor.primary,
  };

  return (
    <button style={styles} {...props} onClick={onClick}>
      {children}
    </button>
  )
}

Button.propTypes = {
  variant: PropTypes.oneOf([
    "primary",
    "secondary",
    "black"
  ]),
  onClick: PropTypes.func,
}

Button.defaultProps = {
  variant: "primary",
  onClick: () => {},
}

export default Button;
```



Pada component di slide sebelumnya, terdapat definisi property bernama variant, yang hanya dapat bernilai, **primary**, **secondary**, atau **black**. Oiya, kita pake package bernama prop-types, package ini membantu developer experience kita dalam mendefinisikan dan mendokumentasikan component.

Lalu, setelah kita mendefinisikan component seperti di atas, kita dapat menggunakan component tersebut di component lain yang membutuhkan component tersebut.

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import MainLayout from "./layouts/MainLayout";
import Button from "./components/Button";

const root =
ReactDOM.createRoot(document.getElementById('root'));
root.render(
<React.StrictMode>
  <MainLayout>
    <Button variant="primary">
      Click
    </Button>

    <Button variant="secondary">
      Click
    </Button>

    <Button variant="black">
      Click
    </Button>
  </MainLayout>
</React.StrictMode>
);
```



Biar mudah untuk mengikutinya, kamu bisa clone [repository ini](#) dan jalankan repository tersebut, maka output dari code tersebut kurang lebih akan seperti ini.

Click

Click

Click



Kalau kamu membuat sebuah User Interface, kira-kira apa aja sih yang akan kamu pikirkan?

Pasti hal yang paling utama adalah gimana caranya mempercantik UI.

Nah untuk mempercantik UI tersebut, kamu perlu tau nih bagaimana menggunakan teknik **styling di dalam react !** Markicusss ~





Sebagai pembuka, ada 3 cara umum untuk melakukan styling pada react, yaitu :

- **Menggunakan Global CSS**, artinya kita impor file CSS di dalam React
- **Menggunakan CSS Module**, artinya kita import file css sebagai module dan tempelin class-nya ke element / component
- **Menggunakan Styled Component**, artinya kita membuat component dengan gaya

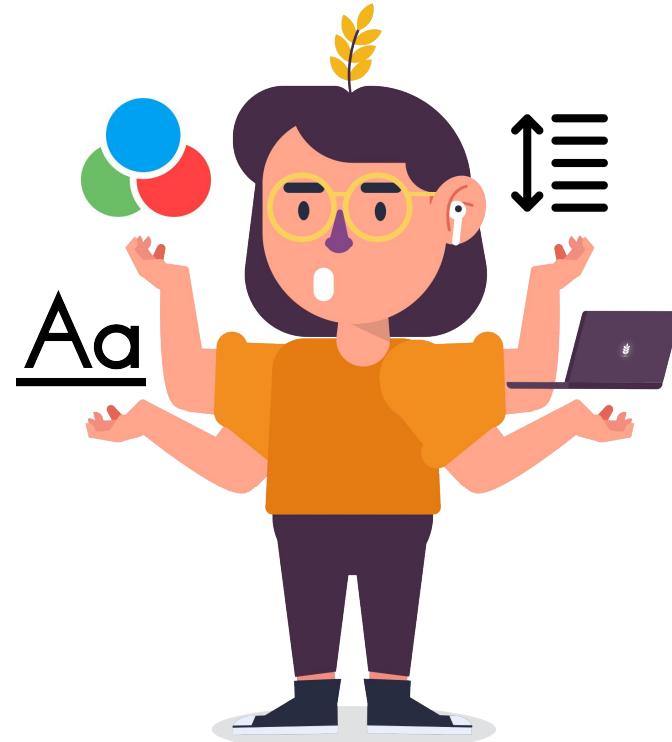




Global CSS

Secara umum Global CSS merupakan **CSS yang kamu impor dan tempel classnya di dalam elemen atau component di dalam React**. Cara ini biasanya digunakan untuk mendefinisikan hal-hal global dalam styling, seperti font, variabel warna, spacing dan sebagainya.

Namun, global CSS tidak akan cocok gaes buat dipake buat styling component-component kecil (Card, Button, Navbar, dll)





Setelah kamu membuat file CSS seperti gambar disamping, kamu tinggal import aja di component manapun yang kamu mau seperti gambar dibawah. Tapi jangan lupa, ini semua akan diterapkan secara global. Kamu bisa cek reponya [disini](#).



src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.css';

const root =
ReactDOM.createRoot(document.getElementById('root'));
root.render(
<React.StrictMode>
  <main>
    <h1>Hello World</h1>
  </main>
</React.StrictMode>
);
```



src/index.css

```
body {
  margin: 0;
  font-family: -apple-system, BlinkMacSystemFont,
'Segoe UI', 'Roboto', 'Oxygen',
'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans',
'Helvetica Neue',
  sans-serif;
-webkit-font-smoothing: antialiased;
-moz-osx-font-smoothing: grayscale;
}

code {
  font-family: source-code-pro, Menlo, Monaco,
Consolas, 'Courier New',
  monospace;
}

main {
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  min-height: 100vh;
  width: 100%;
}
```



CSS Module

Yang kedua adalah dengan menggunakan CSS Module.

CSS module ini cocok untuk styling specific component.

Contohnya ketika kalian ingin membuat 2 macam component bernama Card dan PhotoCard, dan keduanya punya nama class yang sama. Dengan menggunakan css module, styling 2 component tersebut tidak akan menumbuk satu sama lain, berbeda dengan Global CSS.



src/Module.module.css

```
.Module {  
  display: flex;  
  flex-direction: column;  
  align-items: center;  
  justify-content: center;  
  aspect-ratio: 16 / 9;  
  width: 100%;  
  background-color: black;  
}  
  
.Module-title, .Module-description {  
  color: white;  
}  
  
.Module h1, .Module p {  
  max-width: 48rem;  
  margin: inherit auto;  
}
```



Seperti yang bisa kamu lihat dari contoh pada slide ini dan slide di atas, class di dalam *.module.css akan berbentuk menjadi atribut dari module yang kita impor, sebagai contoh, kita punya class .Module, ketika kita gunakan ia di dalam component, kita akan memanggilnya sebagai atribut .Module dari module tersebut.

```
src/Module.module.css

import React from "react";
import style from "./Module.module.css";

const Module = () => (
  <div className={style.Module}>
    <h1 className={style["Module-title"]}>Hello
    World</h1>
    <p className={style["Module-description"]}>
      Lorem ipsum dolor sit amet, consectetur
      adipiscing elit. Sed vehicula, est
      in vehicula scelerisque, dui diam consectetur
      mauris, non sodales velit
      quam vitae felis. Sed iaculis porta luctus.
      Suspendisse vitae leo gravida,
      euismod urna nec, consectetur lorem. Et nec vel
      auctor eros, id molestie
      elit. Duis ipsum massa, cursus ut euismod a,
      rhoncus in massa. Aenean ac
      dapibus leo, ac sollicitudin arcu. Praesent
      feugiat viverra metus, vel
      efficitur libero rutrum ac. Fusce nec velit
      dapibus, volutpat dui egestas,
      aliquet dolor. Ut a consequat tellus, non
      interdum lectus. Ut vestibulum
      malesuada eros in faucibus. In venenatis, neque
      eget malesuada efficitur,
      ligula tortor consequat nulla, vel vulputate
      ipsum ex ut velit. Phasellus
      in nulla suscipit, laoreet tortor ac, ultrices
      mauris.
    </p>
  </div>
);

export default Module;
```



Styled Component

Yang terakhir adalah styled component, cukup populer dan cukup mudah digunakan. Yang perlu kalian lakukan hanyalah instal styled-components dan gunakan seperti contoh code di samping.

Simple bukan?

dokumentasi lebih lengkap dapat kamu liat [disini](#). Dan contoh reponya sama kok kayak yang diatas, ada [disini](#).



```
yarn add styled-components
```



src/Styled.js

```
import styled from "styled-components"

const Styled = styled.h1`  
  font-size: 1.5em;  
  text-align: center;  
  color: palevioletred;  
`  
  
export default Styled;
```



Okay pemahaman tentang teknik stylingnya dah josss kan?

Selanjutnya, biar pengembangan aplikasi React jauh lebih cepat, kita bisa built-in component yang tinggal tempel-tempel aja di aplikasi react-mu sebagai component dengan mempelajari **UI Framework!**





Mirip dengan CSS Framework yang isinya class-class yang bisa ditempel di element, pada UI Framework isinya adalah React Component yang bisa kita langsung pakai.

Tanpa basa-basi lagi, kita bisa langsung cobain 2 UI Framework yaitu **Ant Design** dan **MUI**.

Warning : Jangan pernah pake 2 UI Framework dalam 1 project, bakal bikin website kalian lemot.





Ant Design

Pertama-tama untuk mulai menggunakan UI Framework, kita perlu instal UI Framework tersebut di dalam React Project kita. Ant Design adalah salah satu UI Framework yang populer di kalangan developer React.

Detail cara instalasi Ant Design ada [disini](#). Tapi simpelnya ya tinggal instal menggunakan npm atau yarn. Kemudian modifikasi src/index.js, impor antd.css.



```
yarn add antd
```



src/index.js

```
import 'antd/dist/antd.css';
```



Sesudah itu, tinggal panggil aja component-nya seperti component biasa. Sebagai contoh nih, kita akan memanggil component **DatePicker**.



src/App.js

```
import 'antd/dist/antd.css';
```



Dan Boom! UI keren tanpa susah-susah.

Detail component dan cara pakainya bisa kalian liat [disini](#) ya! Dan untuk repo dari code diatas bisa kalian lihat [disini](#).

		May 2022							
Su	Mo	Tu	We	Th	Fr	Sa			
1	2	3	4	5	6	7			
8	9	10	11	12	13	14			
15	16	17	18	19	20	21			
22	23	24	25	26	27	28			
29	30	31	1	2	3	4			
5	6	7	8	9	10	11			

[Today](#)



MUI

Sama seperti Ant Design, ada UI Framework lain yang populer yaitu MUI. **UI Framework ini lengkap dan ada fitur animasi** yang bisa langsung kalian pakai. Cara instalasinya sama aja, tinggal sat set das des instal package seperti gambar disamping.

Dan setelah instalasi package, cas cus modify file src/App.js



```
yarn add @mui/material @emotion/react  
@emotion/styled
```



src/App.js

```
import React from 'react';
import Button from
  '@mui/material/Button';

function App() {
  return <Button
variant="contained">Hello
World</Button>;
}

export default App;
```



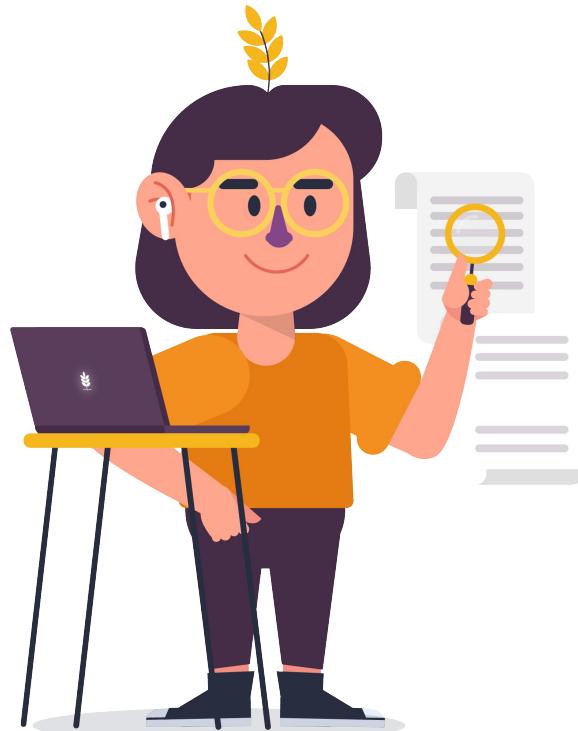
Dan, voilaa! Akan langsung menghasilkan seperti gambar di samping. Keren bukan?

HELLO WORLD

Untuk detail pemakaian bisa kamu liat [disini](#). Dan source code dari code diatas dapat kamu liat [disini](#).

Referensi buat tambahan kamu belajar ~

- <https://reactjs.org/docs/getting-started.html>
- <https://styled-components.com/>
- <https://ant.design/>
- <https://mui.com/>



Terima Kasih!



Next Topic

loading...