UNIVERSITAS MIKROSKIL | Fakultas INFORMATIKA

TI2104 – Pengembangan Aplikasi Web

Komponen React Js dan JSX:

- Pengertian dan fungsi komponen dalam react js
- Membuat dan menggunakan komponen
- Pengenalan JSX dan integrasinya dengan komponen
- Styling Component – dasar CSS

Strata - 1

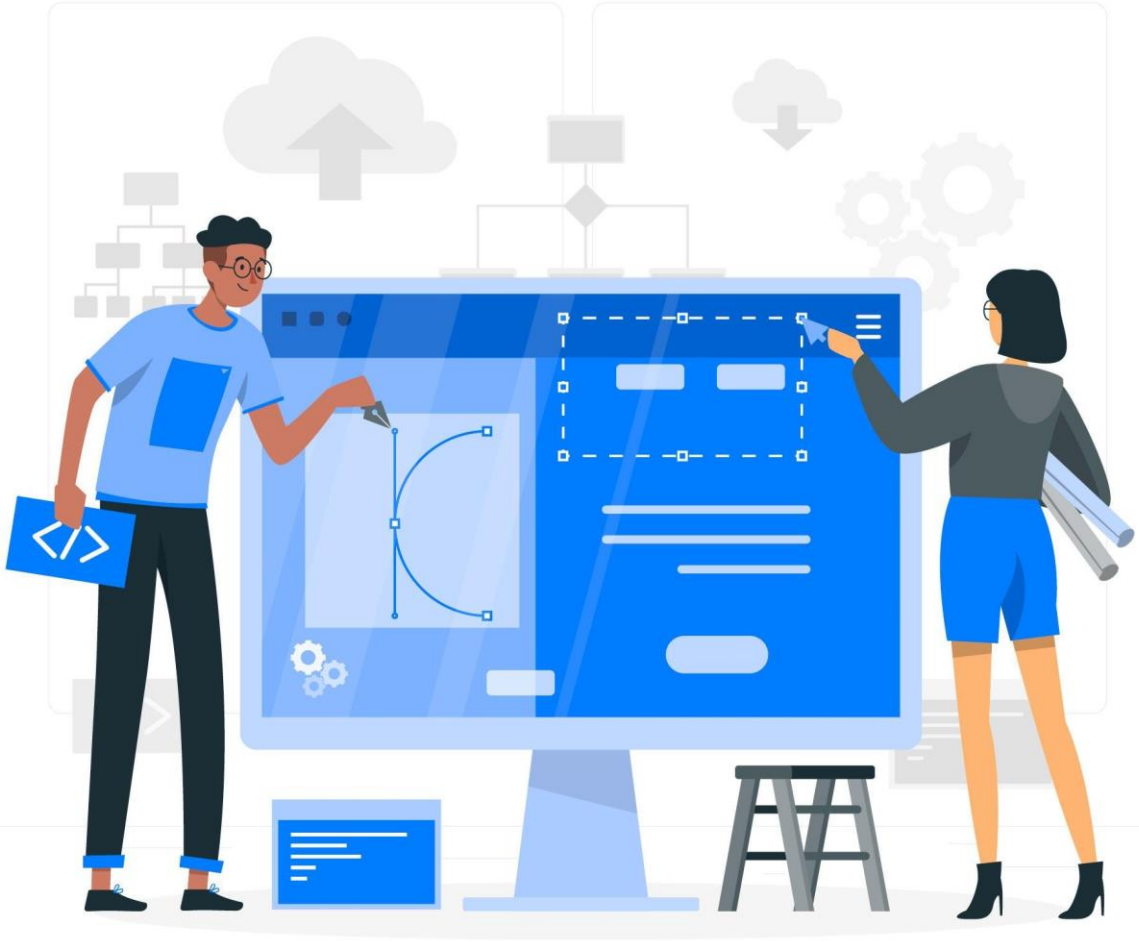
Teknologi Informasi

User Interface (UI)

Chapter 1

Pengenalan User Interface (UI)

User Interface (UI) adalah segala sesuatu yang memungkinkan pengguna berinteraksi dengan sebuah produk, baik itu perangkat keras (seperti **ponsel** atau **komputer**) maupun perangkat lunak (seperti **aplikasi** atau **situs web**). Ini mencakup elemen-elemen visual dan interaktif yang memungkinkan pengguna untuk berkomunikasi dengan sistem.

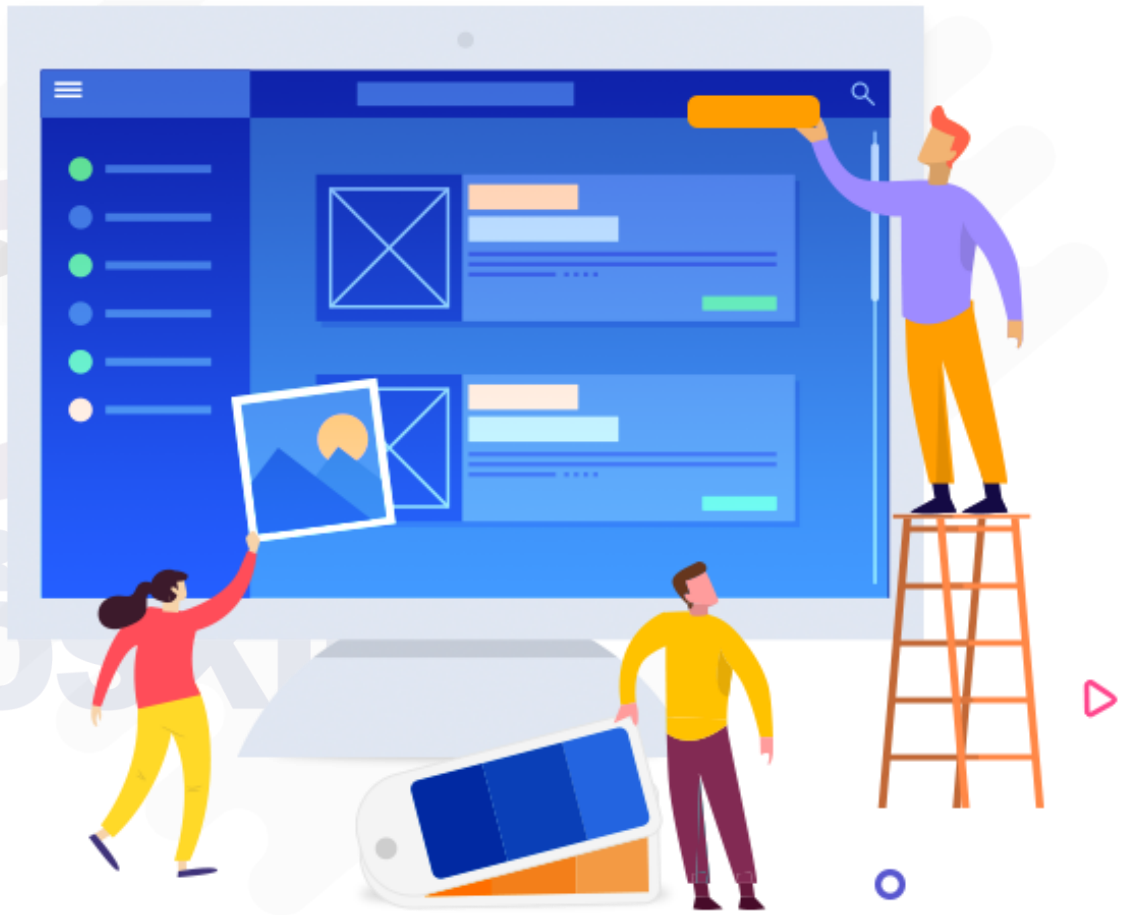


Peran User Interface

Peran utama UI adalah memberikan cara yang efisien dan efektif bagi pengguna untuk berinteraksi dengan produk atau sistem.

UI yang baik dapat meningkatkan pengalaman pengguna dengan membuatnya lebih intuitif, efisien, dan menarik.

UI yang buruk dapat mengakibatkan frustrasi pengguna dan bahkan penurunan penggunaan produk.



Contoh User Interface

Aplikasi Ponsel: Contoh UI dalam aplikasi ponsel termasuk layar beranda dengan ikon aplikasi, menu navigasi, tombol-tombol, dan antarmuka yang ramah pengguna.

Situs Web: Pada situs web, UI melibatkan elemen-elemen seperti tata letak, menu navigasi, tombol-tombol, formulir, gambar, dan teks.

Perangkat Lunak Komputer: Di perangkat lunak desktop, UI mencakup jendela aplikasi, bilah menu, bilah alat, tombol-tombol, dan elemen lain yang memungkinkan pengguna untuk berinteraksi dengan program.

Peran Reactjs dalam pembuatan UI Interaktif

- **Komponen yang reaktif :** React memungkinkan Anda untuk membagi UI menjadi komponen-komponen yang mandiri dan dapat digunakan kembali.
- **Manajamen State :** React memudahkan manajemen state dalam aplikasi. Anda dapat dengan mudah mengubah dan memperbarui status komponen
- **Rendering Efisien :** Virtual DOM (Document Object Model) untuk merender UI, ni membuat proses merender lebih efisien.
- **Event handling :** React memungkinkan Anda untuk menangani peristiwa (event) dengan mudah.
- **Reusabilitas Komponen :** Anda dapat membuat komponen yang dapat digunakan kembali untuk membangun UI yang kompleks. Ini meningkatkan efisiensi pengembangan dan memungkinkan Anda untuk mempertahankan kode yang konsisten.
- **Integrasi yang Mudah :** React dapat dengan mudah diintegrasikan dengan berbagai alat dan pustaka lain dalam ekosistem JavaScript. Kita dapat memanfaatkan alat seperti React Router untuk menangani routing di aplikasi kita.
- **Pengembangan Berbasis Komponen :** React memungkinkan pengembang bekerja secara lebih terstruktur, memudahkan pengujian unit, pemeliharaan, dan skalabilitas aplikasi.
- **Penampilan UI yang Konsisten :** React membantu menjaga penampilan UI yang konsisten di seluruh aplikasi, karena Anda mendefinisikan tampilan dalam komponen yang dapat digunakan Kembali
- **Ekosistem yang Kaya :** React memiliki ekosistem yang kuat dan aktif dengan berbagai alat dan pustaka tambahan yang mendukung pengembangan UI yang interaktif.

(?) Cari Tahu

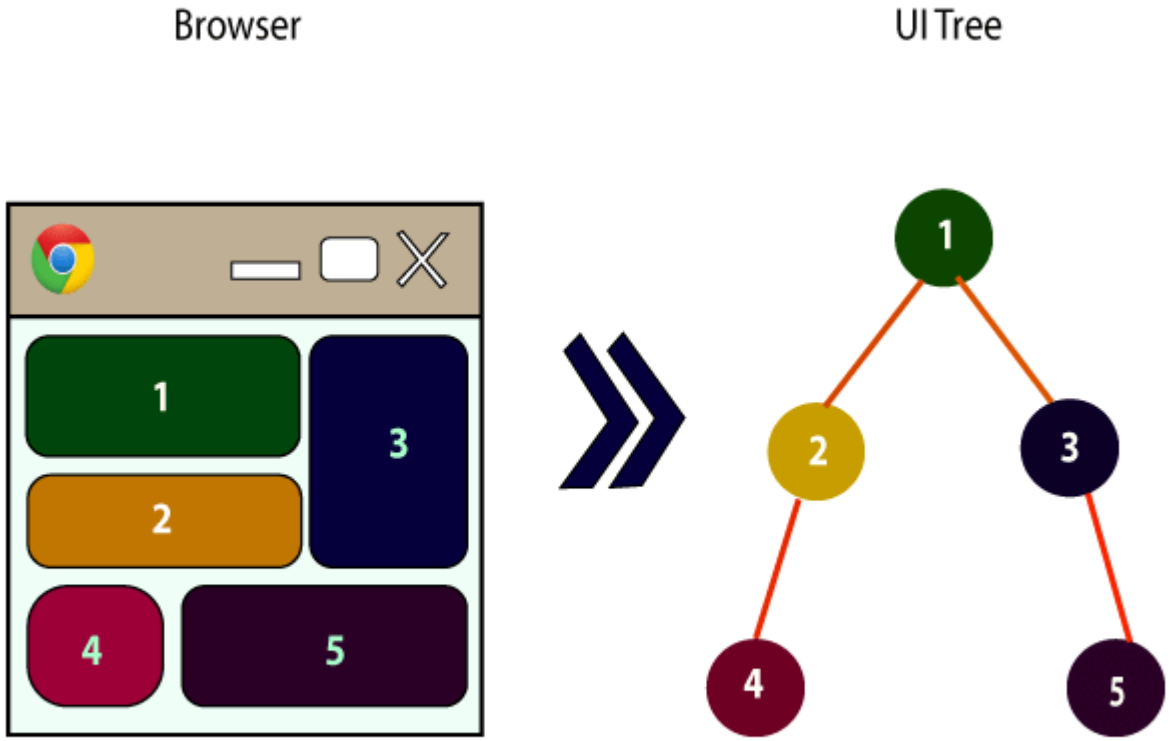
- Apakah React mampu membuat UI yang baik ?
- 5 Aplikasi besar yang dibuat dengan react ?
- Pada UI mana yang lebih penting, keindahan atau fungsionalitas ?
- Bagaimana peran React dalam pembuatan UI ?

React Element

Chapter 2

Pengenalan React Element

React Element adalah objek yang merepresentasikan elemen tampilan dalam React.



static.javatpoint.com

Membuat React Element

Seluruh User Interface (UI) aplikasi react dibangung menggunakan **React Element**.

React Element dapat berupa Header, Footer, Button ataupun Paragraf.

React Element Mempresentasikan apa yang kita lihat di Layar.

```
React.createElement(/* type, */ /* property, */ /* content */);
```

- **Type** merupakan type element
- **Property** merupakan property dari element
- **Content** merupakan content dari element

```
React.createElement("h1", { id: "title", className: "title" }, "Hello World!");
```

Membuat Element Bertingkat

Content pada React element dapat disi dengan type data apapun, termasuk React element lainnya. Hal ini lah yang diperlukan untuk membuat elemen bertingkat.

html

```
<div class="contanier">
  <h1 class="title">Hello World</h1>
  <p>
    Lorem, ipsum dolor sit amet consectetur
    adipisicing elit. Aliquid eligendi veniam
    minus. Odio eaque quo,
  </p>
</div>
```

React element

```
const text = React.createElement(
  "p",
  null,
  "Lorem, ipsum dolor sit amet consectetur
  adipisicing elit. Aliquid eligendi veniam
  minus. Odio eaque quo"
);
const title = React.createElement("h1", {
className: "title" }, "Hello World");
const container = React.createElement("div", {
className: "container" }, [
  title,
  text,
]);
```

Rendering Element

Untuk menampilkan **React element** pada browser, kita perlu membuat **root** yang nantinya digunakan sebagai penampung.

Untuk membuat **root**, gunakan fungsi **ReactDOM.createRoot** yang ada pada modul “react-dom/client”.

```
import ReactDOM from "react-dom/client";

const root = ReactDOM.createRoot(document.
getElementById("root"));
root.render(container);
```

(?) Cari Tahu

- Mana yang lebih mudah digunakan HTML, JS Dom atau React element ?
- Bagaimana cara menuliskan Content, Atributte dan Tag pada React Element ?
- Apa perbedaan React element dan React Component ?

JSX

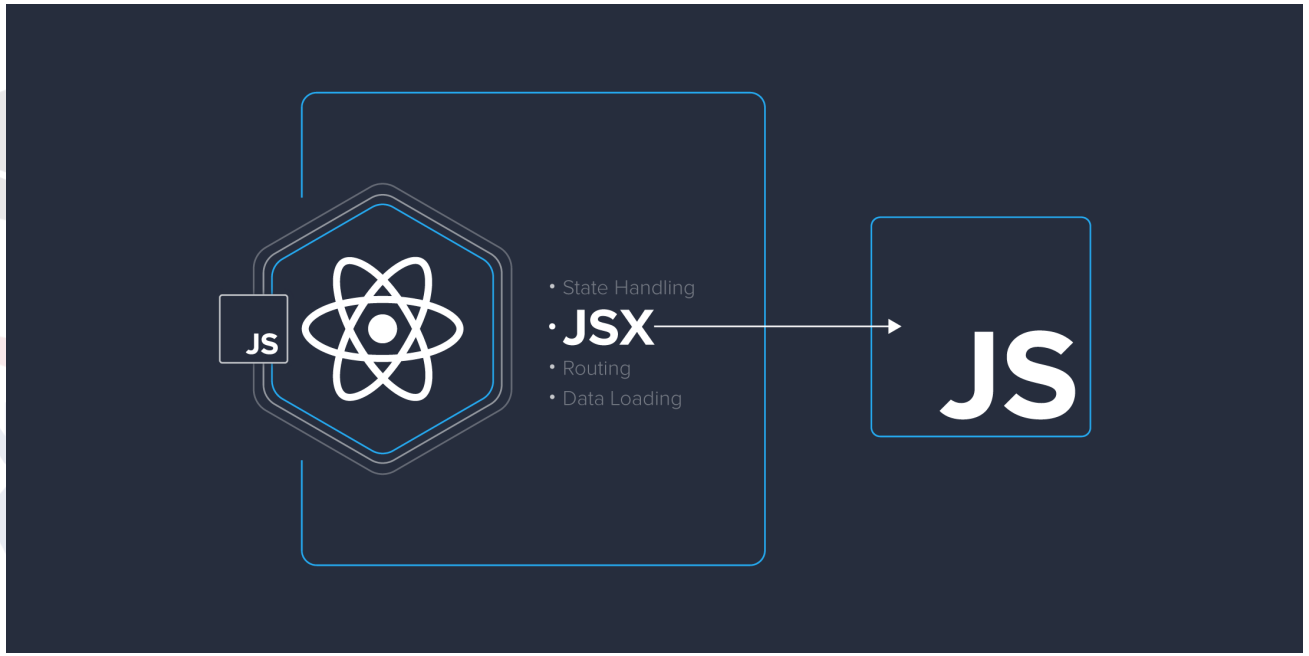
Chapter 3

Pengenalan JSX

JSX adalah singkatan dari Javascript **XML**.

JSX menggabungkan Javascript dan **HTML**.

JSX memungkinkan mendefenisikan tampilan **komponen** dalam bentuk yang mirip dengan **HTM**L, hal ini sangat memudahkan **pengembangan** dan **pemahaman kode**.



React Element dengan JSX

Membuat Element dengan React element terasa **lebih sulit** daripada menuliskan HTML biasa.

JSX adalah **solusi** pembuatan elemen yang jauh lebih mudah dan bersifat deklaratif.

Pada JSX **tipe elemen** ditentukan dengan **tag**.

Children Element dapat ditambahkan **antara** tag pembuka dan penutup.

Atribute dapat di letakkan di dalam tag pembuka.

Jika tag bersifat kosong (tidak memiliki elemen anak), Anda bisa saja menutupnya secara langsung dengan />, seperti XML

```
const name = "Budi"
const element = <h1>Halo, {name}</h1>
```

```
const linkElement = (
  <a className="link" href="https://www.reactjs.org">
    link
  </a>
);
```

```
const imgElement = (
  <img
    src={
      "https://w7.pngwing.com/pngs/403/269/
      png-transparent-react-react-native-logos-brands-i
      n-colors-icon-thumbnail.png"
    }
  />
);
```

Menyisipkan JSX Expression

Setelah dikompilasi, JSX Expression akan menjadi panggilan fungsi JavaScript biasa dan menjadi objek JavaScript.

Dengan JSX kita dapat menyematkan semua ekspresi javascript yang valid di dalam kurung kurawal di JSX.

```
function formatName(user) {
  return user.firstName + " " + user.
  lastName;
}

const user = {
  firstName: "Budi",
  lastName: "Mahardika",
};

const element = <h1>Halo, {formatName
(user)}!</h1>;
```

Element Bertingkat dengan JSX

```jsx
const nestedElement = (
  <div>
    <h1>Halo!</h1>
    <h2>Senang melihatmu di sini.</h2>
  </div>
);


export default nestedElement;
```

(?) Cari Tahu

- Apa itu JSX, mengapa digunakan dalam React ?
- Apa perbedaan JSX dengan HTML Konvensional ?
- Apa yang dimaksud dengan expression dalam JSX, dan bagaimana cara menggunakannya ?

React Component

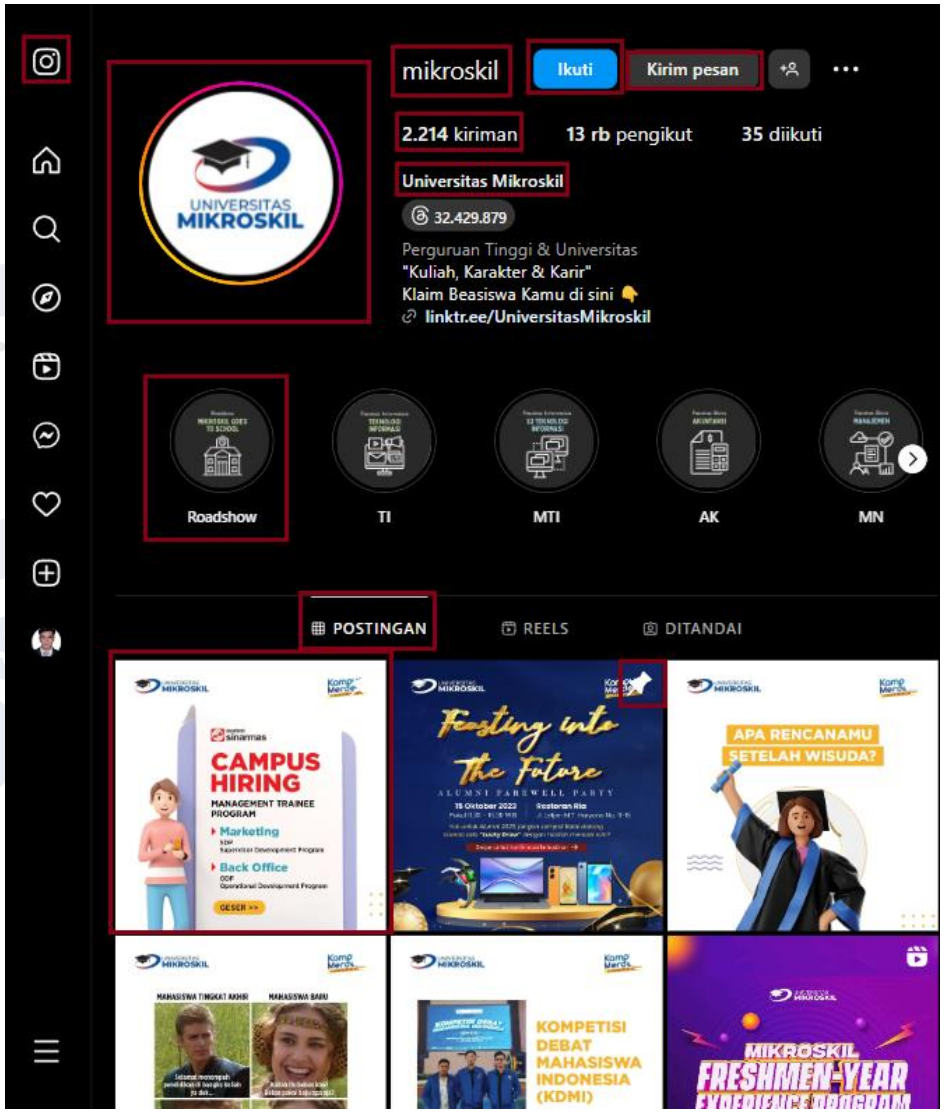
Chapter 4

Pengenalan Komponen

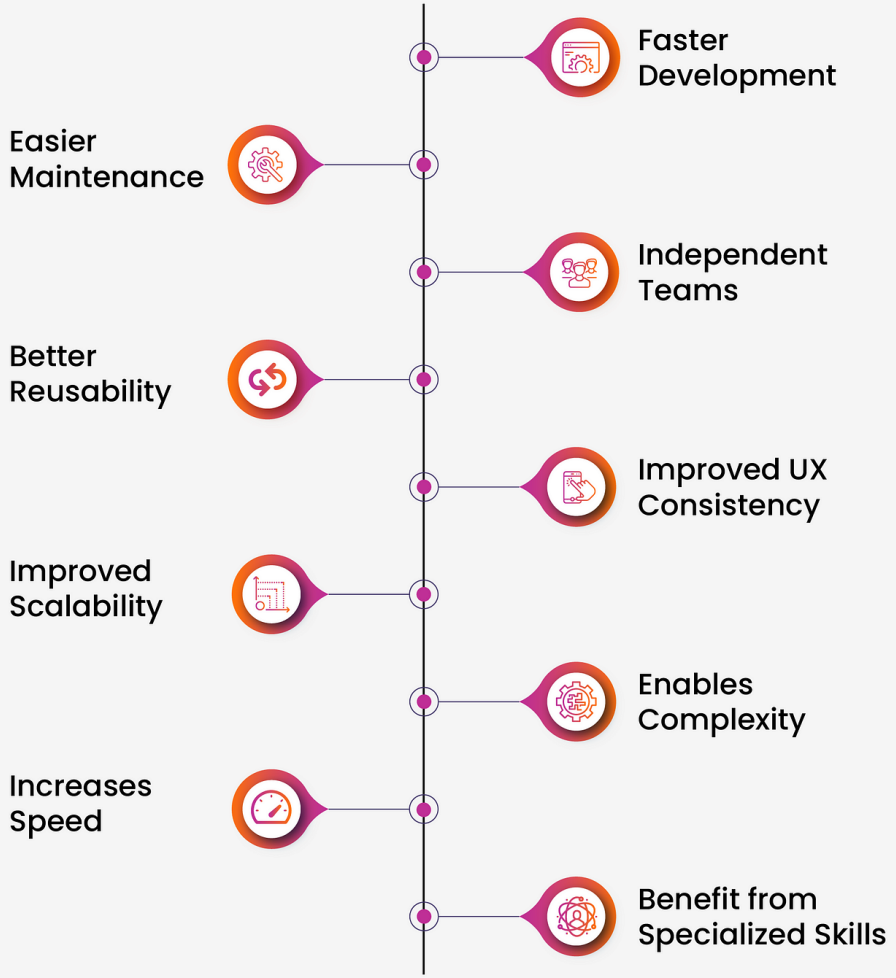
Komponen adalah blok dasar pembangun UI dalam yang dapat digunakan secara independen dan dapat digunakan kembali.

Komponen juga memungkinkan pemisahan logika tampilan dari logika bisnis dalam aplikasi Anda.

Dalam bahasa yang lebih sederhana, **Komponen** adalah potongan-potongan UI yang dapat dirancang, dibuat, dan gunakan kembali.



Advantages of Component-Based Development



maruti techlabs

Component pada React

- Selain React element, fitur utama lainnya yang sangat membantu kita dalam membangun UI adalah **React Component**.
- Fitur ini adalah kunci jika Anda ingin membangun UI yang **reusable** dan **terisolasi**.
- Lebih mudahnya, React component merupakan fungsi JavaScript yang **mengembalikan React element**.
- Alasan mengapa kita membuat React component sama dengan kapan kita harus membuat sebuah fungsi.
- Konsep component sangat membantu karena kita dapat **memecah (break down) UI** ke bagian-bagian **kecil**.
- Bagian-bagian kecil dari UI tersebut memiliki tanggung jawab yang jelas dan properti yang sudah terdefinisikan.
- Hal ini sangat **penting ketika membangun aplikasi yang besar** karena kita dapat bekerja **fokus** pada bagian terkecil dari aplikasi tanpa **mengganggu** keseluruhan kode yang ada.
- Component pada React terbagi menjadi 2 jenis, yaitu **Class Component** dan **Functional Component**

Membuat React Component

- Menulis fungsi yang mengembalikan React element adalah cara paling mudah membuat react component.

Function Biasa :

```
function HelloWorld() {
  return <h1>Hello World</h1>;
}
```

Arrow Function :

```
const HelloWorld = () => {
  return <h1>Hello World</h1>;
};
```

Catatan: Konvensi dalam penamaan React component selalu diawali dengan huruf kapital. Hal ini bertujuan agar dapat membedakan antara built-in HTML element dan component yang Anda (atau pihak lain) buat sendiri.

Memanggil React Component

React Component dapat digunakan seperti memanggil fungsi biasa ataupun dengan syntax jsx.

Fungsi Biasa

```
root.render(<div>{HelloWorld()}</div>);
```

Syntax JSX

```
root.render(
  <div>
    <HelloWorld />
  </div>
);
```

Penting :
Meskipun React component adalah fungsi JavaScript, tetapi ia harus dipanggil layaknya sebuah tag HTML. Sebuah fungsi yang mengembalikan React element belum bisa dikatakan component bila Anda memanggilnya seperti fungsi biasa.

Cara lain membuat Component React

Class component adalah jenis komponen dalam React yang didefinisikan sebagai kelas JavaScript.

Mereka memiliki **struktur yang lebih kompleks** dibandingkan dengan functional component, dengan kemampuan untuk memiliki state (keadaan) dan metode siklus hidup.

Class component digunakan dalam pengembangan React sebelum pengenalan hooks, dan meskipun sekarang telah **digantikan** oleh functional component.

Class Component

```
class HelloWorld extends React.Component {
  constructor() {
    super();
  }

  render() {
    return <h1>Hello World!</h1>;
  }
}
```

Functional Component VS Class Componet

Fitur / Aspek	Functional Component	Class Component
Sintaksis	Berbasis fungsi (fungsi JavaScript biasa)	Berbasis kelas (dengan extends React.Component)
State	Menggunakan Hook useState atau tidak memiliki state	Memiliki state yang didefinisikan dalam this.state
Lifecycle Methods	Menggunakan Hook useEffect untuk efek samping dan pembaruan komponen	Menggunakan metode siklus hidup seperti componentDidMount, componentDidUpdate, dan componentWillUnmount
Props	Menerima props sebagai argumen pada fungsi	Mengakses props melalui this.props
Pembuatan Komponen	Mudah dan lebih singkat	Memerlukan definisi kelas yang lebih panjang
Kode Bersih	Menghasilkan kode yang lebih bersih dan deklaratif	Memerlukan lebih banyak kode boilerplate
Pengujian (Testing)	Lebih mudah diuji (unit testing) karena merupakan fungsi murni	Memerlukan pengujian yang lebih rumit karena metode siklus hidup
State Management	Menggunakan Hook useState atau konteks untuk manajemen state	Menggunakan this.setState untuk memperbarui state

Properties Component

Agar bisa memberikan **nilai yang dinamis** pada component kita bisa menggunakan fitur **properties** atau yang biasa lebih dikenal dengan istilah **props**.

Props dapat digunakan untuk mengirimkan nilai atau data apapun ke **child component**.

Props di defenisikan dalam tag komponen saat di render.

```
function UserGreeting(props) {
  return (
    <p>
      Hello I'am {props.name}, i am {props.age} years old
    </p>
  );
}

function App() {
  return (
    <div>
      <UserGreeting name="Budi Sudarsono" age={26} />
      <UserGreeting name="Ismed Sofyan" age={22} />
    </div>
  );
}

const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(<App />);
```

(?) Cari Tahu

- Apa itu react component, dan mengapa begitu penting dalam pengembangan aplikasi react ?
- Bagaimana komponen dapat berkomunikasi satu sama lain dalam React ?
- Bagaimana cara menggunakan state dan props dalam React Component ?
- Bagaimana Menyusun component secara efektif dalam project yang lebih besar ?

Styling Component Dasar

Chapter 4

Styling

Dalam dunia pengembangan aplikasi web modern, tampilan antarmuka pengguna (UI) yang menarik dan fungsional adalah hal yang sangat penting.

Untuk mencapai ini dalam lingkungan React, kita perlu memahami bagaimana cara menghias komponen-komponen kita agar sesuai dengan visi desain yang diinginkan.

Styling React Component

Ada banyak cara yang berbeda untuk mengatur tampilan komponen, mulai dari pendekatan modern hingga tradisional.

Salah satu pendekatan yang masih sangat relevan dan digunakan luas adalah **CSS biasa.**

Dalam materi ini, kita akan mencoba konsep dan praktik styling komponen dalam React dengan menggunakan **CSS biasa**.

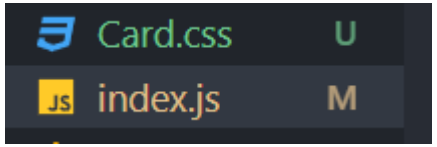
Styling React Component with CSS

Kita bisa membuat css di file berbeda.

Kemudian Melakukan Import di file component kita.

Sama halnya seperti menggunakan CCS pada HTML.

File



komponen

```
import "./Card.css";

function Card(props) {
  return (
    <div className="card">
      <h1 className="title">{props.title}</h1>
    </div>
  );
}

function App() {
  return (
    <div className="card-container">
      <Card title="Card 1" />
      <Card title="Card 2" />
      <Card title="Card 3" />
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(<App />);
```

style

```
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}
.card-container {
  width: 600px;
  gap: 12px;
  margin: 0 auto;
}

.card {
  box-shadow: rgba(0, 0, 0, 0.16) 0px 1px 4px;
  height: 280px;
  flex: 1;
}

.card-container,
.card {
  display: flex;
  align-items: center;
  justify-content: center;
}

.title {
  font-size: 28px;
  color: rgb(62, 61, 61);
}
```

(?) Cari Tahu

- Apa praktik terbaik dalam melakukan styling terhadap react komponen ?
- Apa saja Teknologi dalam melakukan styling yang bisa di integrasikan kepada React ?
- Bagaimana mengoptimalkan tampilan React untuk performa yang lebih baik dan waktu muat yang lebih cepat ?
- Bagaimana mengintegrasikan gambar dan icon dalam styling react ?

Menyusun Komponen

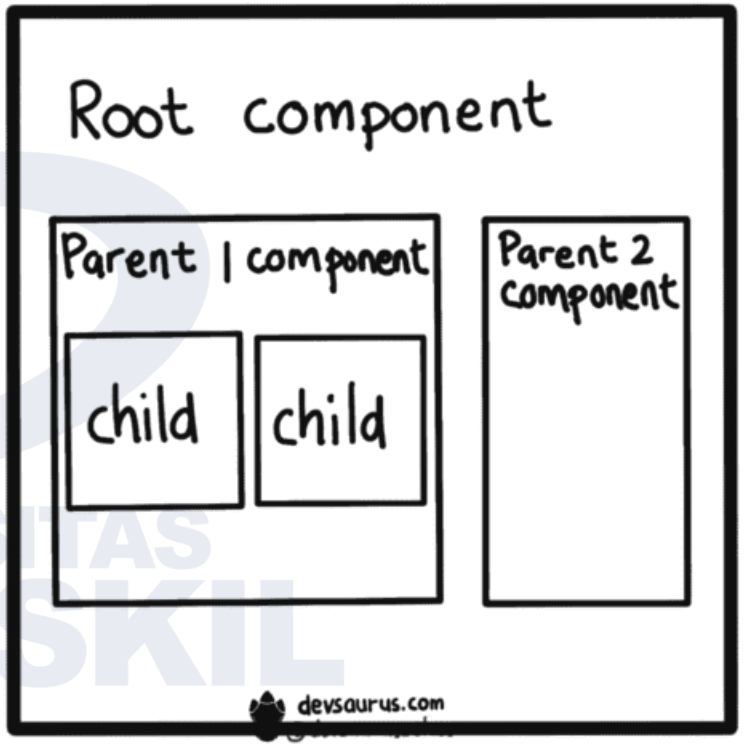
Chapter 5

Pentingnya Menyusun Komponen

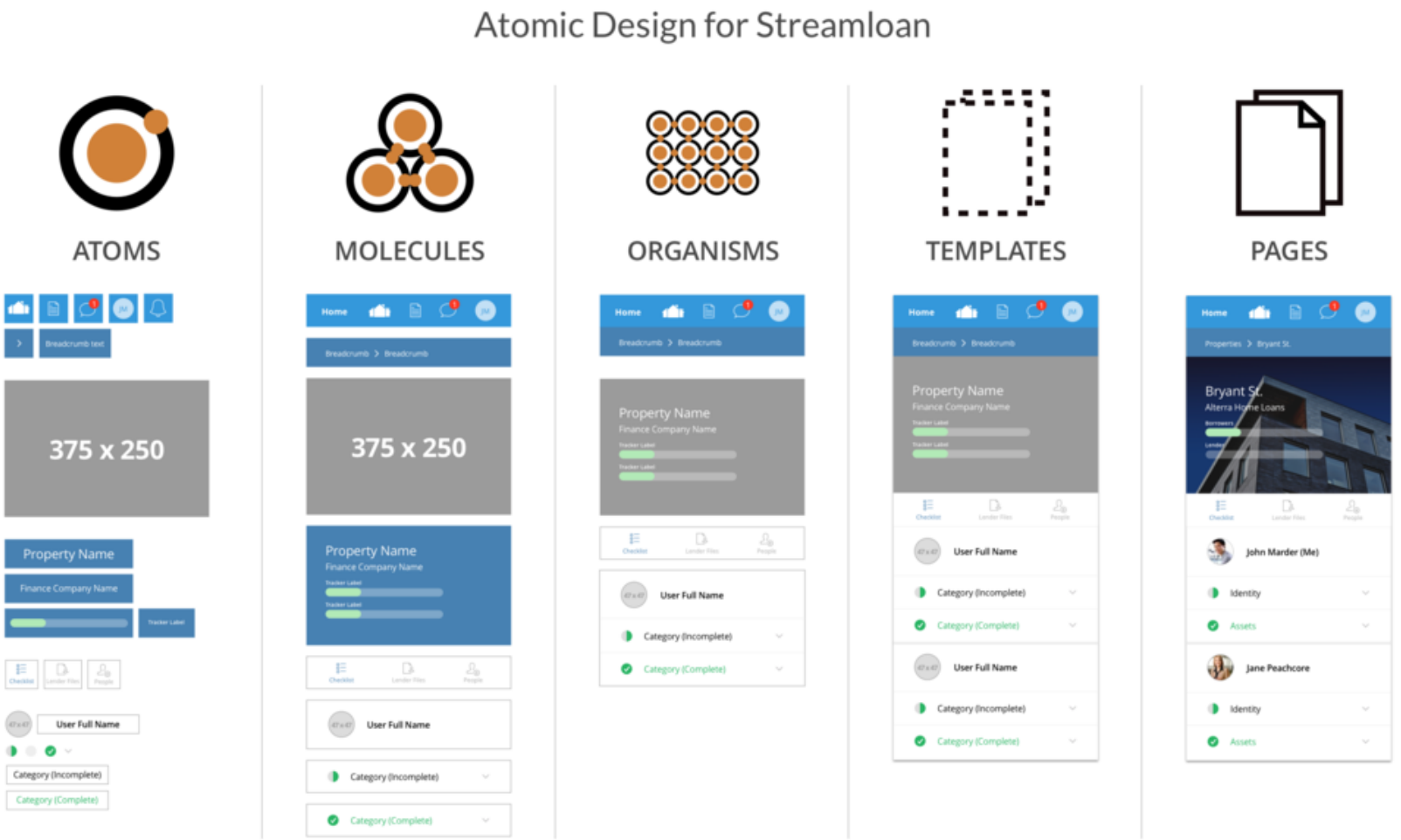
Menyusun komponen dalam React adalah penting karena **meningkatkan modularitas**, **reusabilitas**, dan **keterbacaan kode**.

Hal Ini memungkinkan pengelolaan yang **lebih baik**, pengujian yang **efisien**, **dan perbaikan bug yang lebih cepat**, dan **memudahkan pengembangan aplikasi** yang lebih **skalabel**.

Salah satu **metode paling populer** dalam menyusun komponen adalah **Atomic Design**. Konsep ini membantu dalam mengorganisasi komponen ke dalam tingkatan yang sesuai.



Atomic Design



Atomic Design Cont...

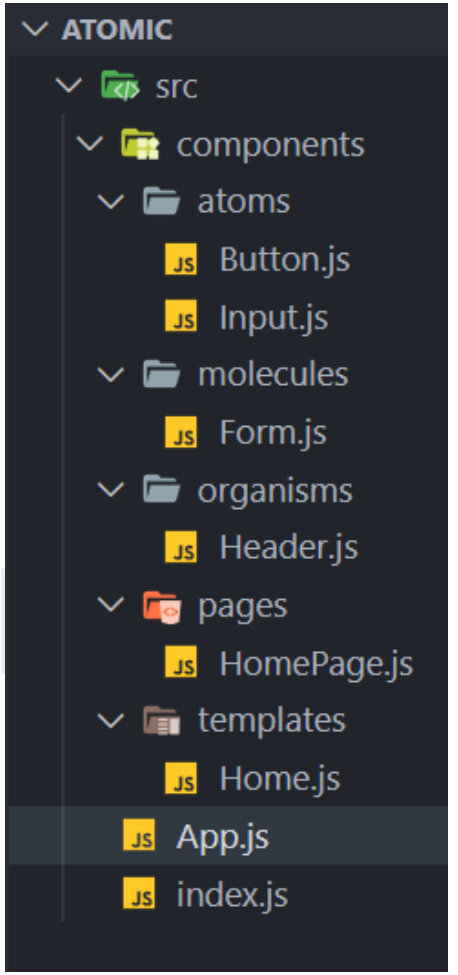
Atoms: Komponen tingkat atom adalah komponen paling dasar yang tidak dapat dibagi lebih lanjut.

Molecules: Komponen tingkat molekul menggabungkan beberapa komponen tingkat atom menjadi satu unit yang lebih besar dan berfungsi.

Organisme: Komponen tingkat organisme adalah komponen yang lebih besar yang menggabungkan beberapa komponen tingkat molekul atau atom.

Templates: Template adalah kerangka yang lebih umum yang mengatur tampilan sebuah halaman atau bagian besar dari aplikasi.

Pages: Komponen tingkat halaman adalah representasi dari halaman atau tampilan akhir yang diberikan kepada pengguna.



(?) Cari Tahu

- Apa yang terjadi jika komponen tidak disusun sebaik mungkin ?
- Apa pendekatan selain Atomic design dalam hal Menyusun komponen React ?
- Bagaimana sebaiknya mengorganisir dan mengelola struktur Folder, File dan Komponen pada React ?
- Bagaimana cara memutuskan sebuah komponen harus menjadi komponen independent atau komponen besar ?
- Bagaiamana cara membuat atau merancang komponen yang reusable ?

Diskusi