**PRAKTIKUM**

**PENGEMBANGAN APLIKASI BERGERAK**

****

**oleh:**

Muhammad Alvin Ramadhan (1204210025)

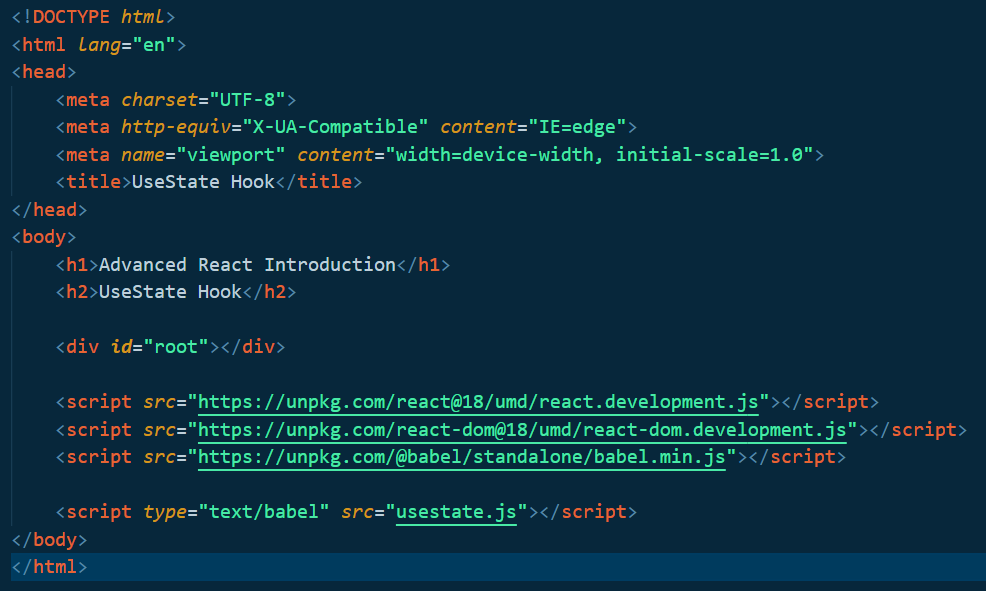
**PROGRAM STUDI SISTEM INFORMASI**

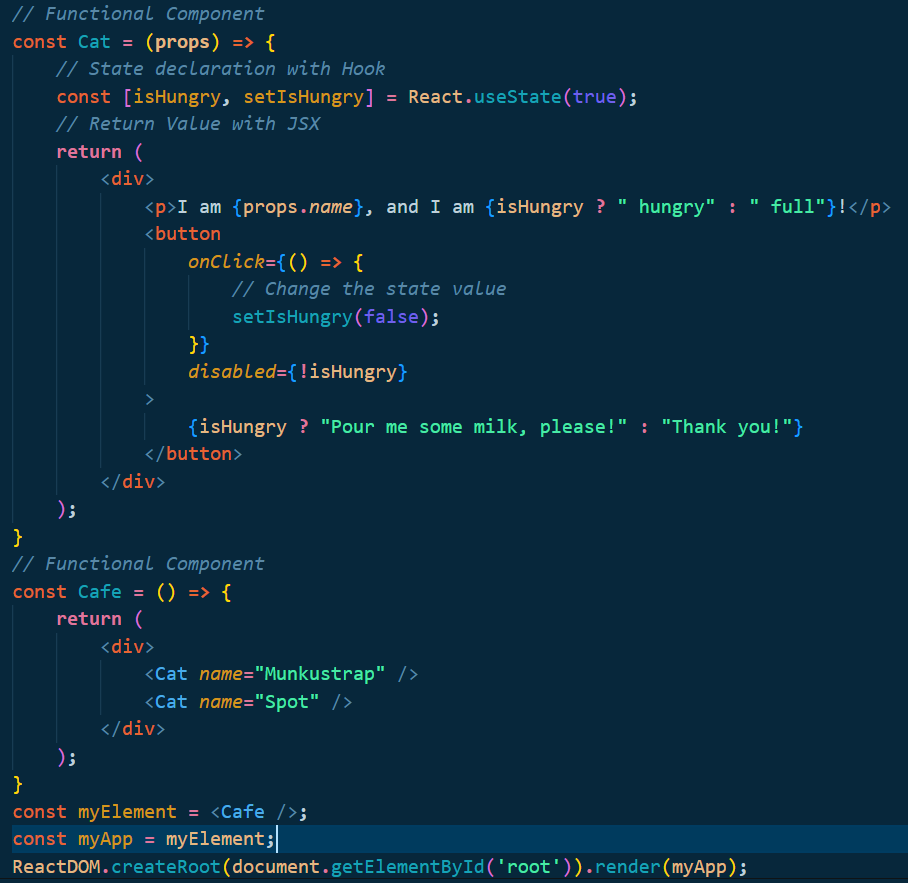
**FAKULTAS TEKNOLOGI INFORMASI DAN BISNIS**

**INSTITUT TEKNOLOGI TELKOM SURABAYA**

**2023**

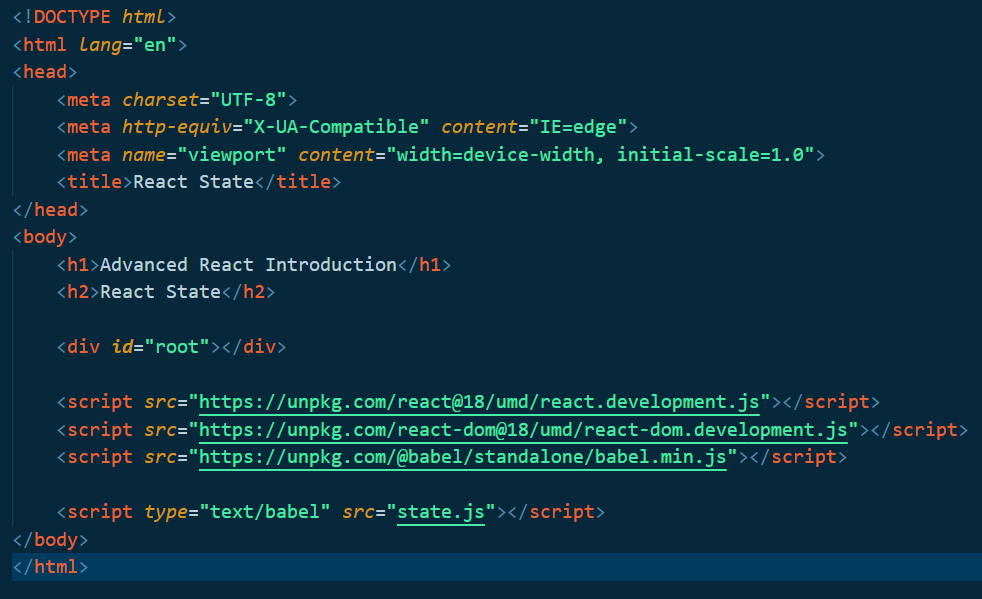
**1. Usestate Hook**

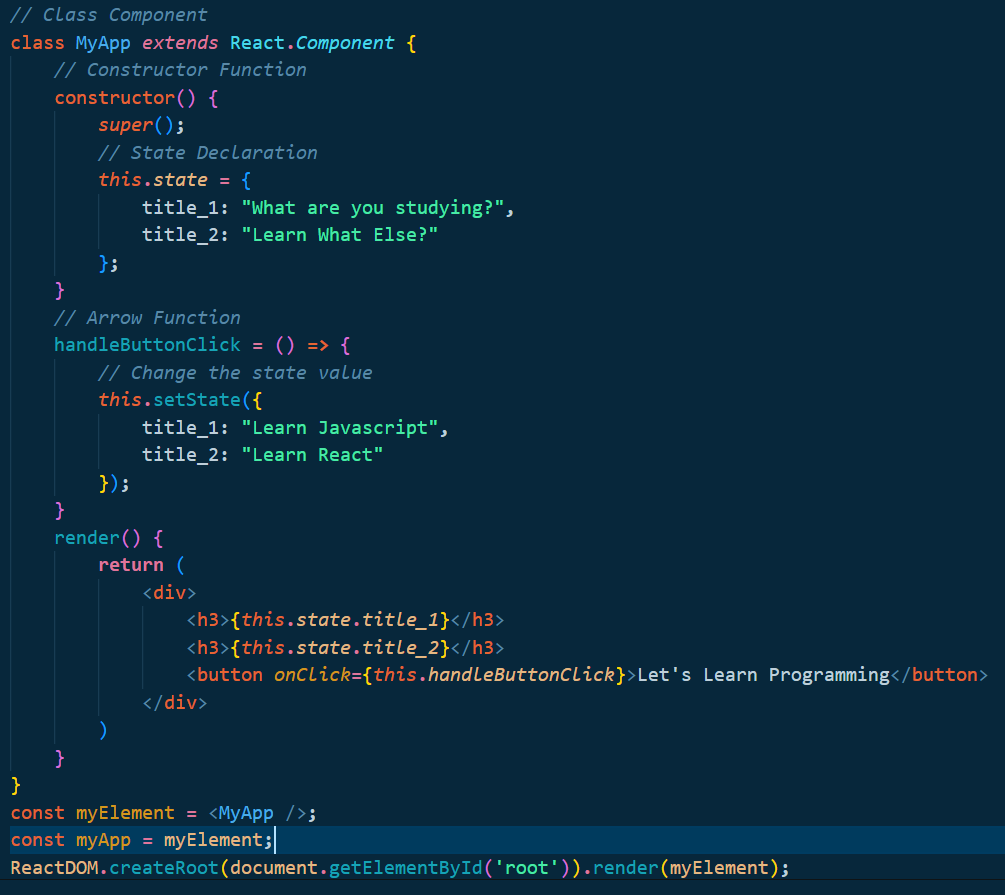
****



Diawal isHungry bernilai true kemudian Ketika diklik maka akan berubah false dan menampilkan teks seperti diatas

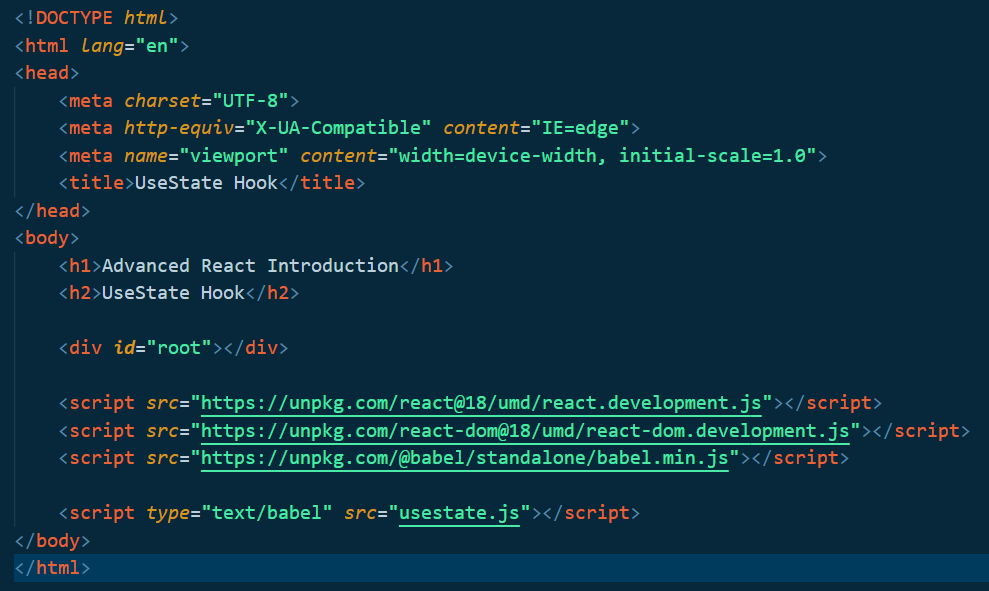
**2. React state**

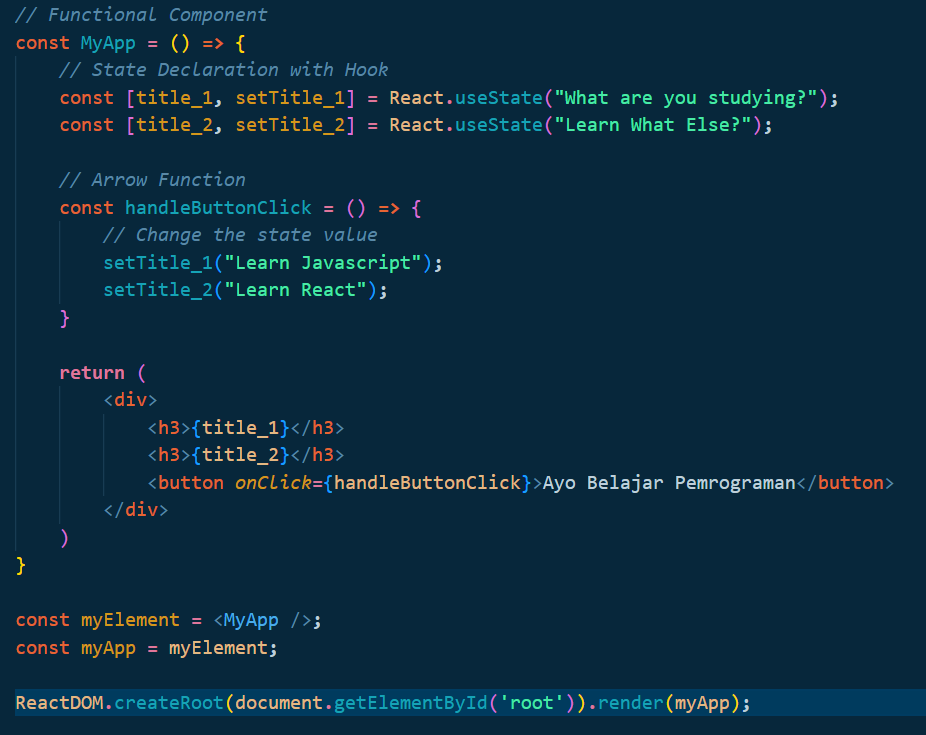
****

****

State memungkinkan komponen untuk berinteraksi dengan pengguna, menanggapi peristiwa, dan memperbarui dirinya sendiri secara dinamis.

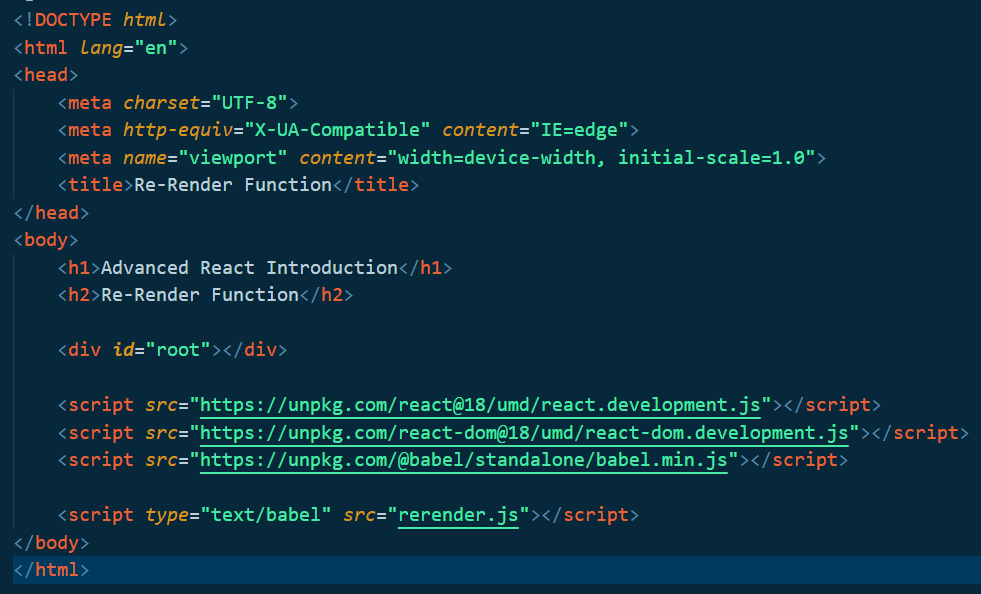
**3. Usestate hook**





usestate diatas memiliki set title yang bernilai "What are you studying?", dan "Learn What Else?" kemudian pada arrow function diberi perintah apabila button diklik maka akan merubah teks sesuai teks diatas

**4. Re-render**



*// Functional Component*

const MyApp **=** () **=>** {

*// State declaration with Hook*

    const [title**,** setTitle] **=** React**.**useState("Learn React")**;**

*// Variable declaration*

    let foo **=** "Hei"**;**

    console**.**log('A:'**,** foo)**;**

*// Arrow Function inside Functional Component*

    const handleButtonClick **=** () **=>** {

        console**.**log('B:'**,** foo)**;**

        foo **=** "Hello"**;**

        console**.**log('C:'**,** foo)**;**

**if** (title **===** "Learn React") {

*// Change the state value*

            setTitle("Learn JavaScript")**;**

        }

**else** {

*// Change the state value*

            setTitle("Learn React")**;**

        }

    }

*// Arrow Function inside Functional Component*

    const handleFooClick **=** () **=>** {

        console**.**log(foo)**;**

        alert(foo)**;**

    }

**return** (

        <div>

            <h3>Title: {title}</h3>

            <div>

                <p><button *onClick***=**{handleButtonClick}>Change Title</button></p>

                <p><button *onClick***=**{handleFooClick}>Get Foo Value</button></p>

            </div>

        </div>

    )

}

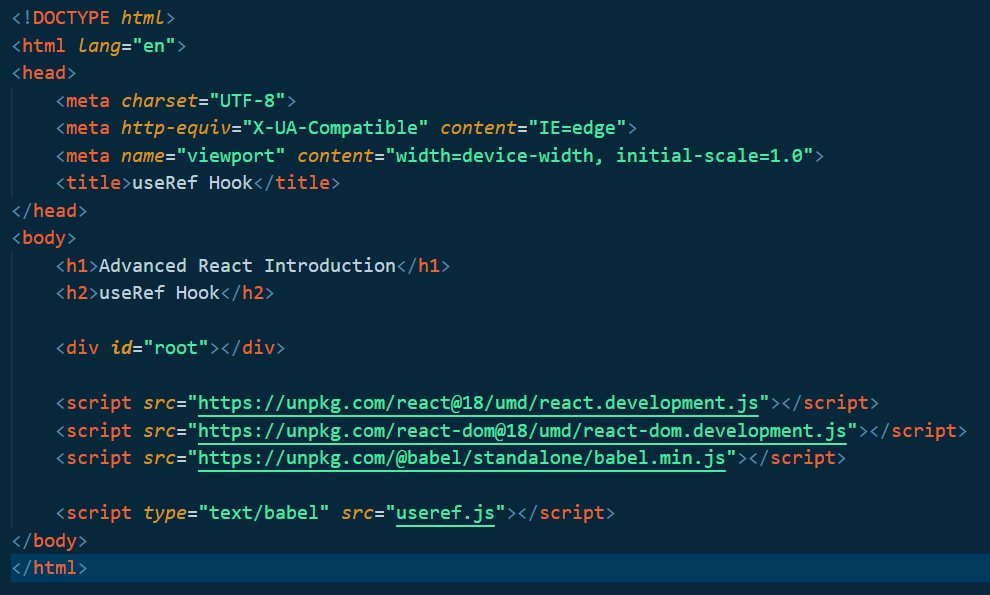
const myElement **=** <MyApp />**;**

const myApp **=** myElement**;**

ReactDOM**.**createRoot(document**.**getElementById('root'))**.**render(myApp)**;**

Ketika terjadi re-render, React akan memperbarui tampilan komponen untuk membawa perubahan pada props atau state.

**5. UseRef**



*// Functional Component*

const MyApp **=** () **=>** {

*// State Declaration*

    const [title**,** setTitle] **=** React**.**useState("Learn React")**;**

*// Variable Declaration with useRef*

    const foo **=** React**.**useRef("Hei")**;**

    console**.**log('A:'**,** foo)**;**

*// Functional Component*

    const handleButtonClick **=** () **=>** {

        console**.**log('B:'**,** foo**.***current*)**;**

        foo**.***current* **=** "Hello"**;** *// Change variable value*

        console**.**log('C:'**,** foo**.***current*)**;**

**if** (title **===** "Learn React") {

            setTitle("Learn JavaScript")**;**

        }

**else** {

            setTitle("Learn React")**;**

        }

    }

*// Functional Component*

    const handleFooClick **=** () **=>** {

        console**.**log('Get Foo:'**,** foo**.***current*)**;**

        alert(foo**.***current*)**;**

    }

**return** (

        <div>

            <h3>Title: {title}</h3>

            <div>

                <p><button *onClick***=**{handleButtonClick}>Change Title</button></p>

                <p><button *onClick***=**{handleFooClick}>Get Foo Value</button></p>

            </div>

        </div>

    )

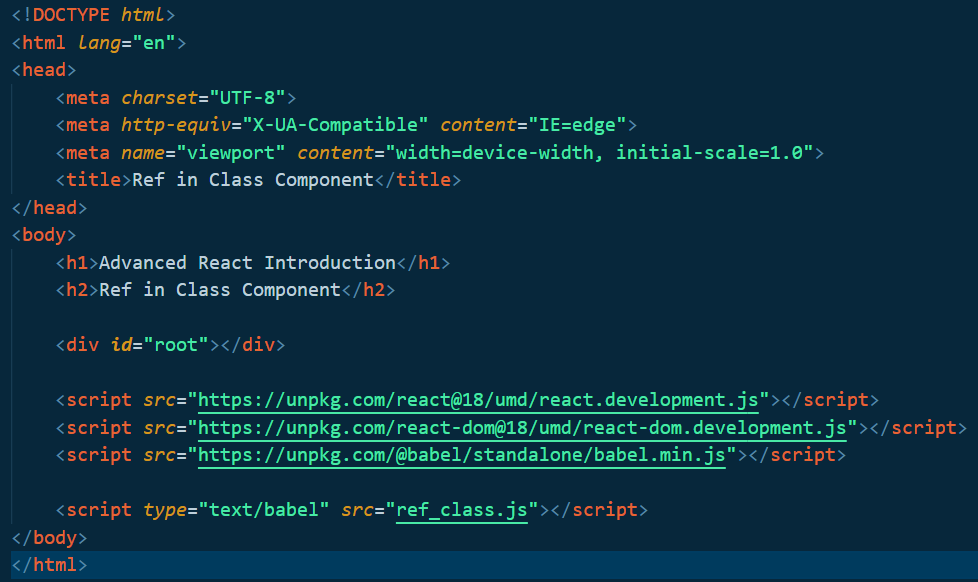
}

const myElement **=** <MyApp />**;**

ReactDOM**.**createRoot(document**.**getElementById('root'))**.**render(myElement)**;**

UseRef adalah salah satu dari hooks yang disediakan oleh React. Ini memungkinkan kita untuk membuat referensi ke elemen atau nilai di dalam komponen fungsional. UseRef sering digunakan untuk mengakses atau memanipulasi DOM nodes atau untuk menyimpan nilai yang akan bertahan antar re-render.

**6. Ref**



*// Class Component*

class MyApp *extends* React**.***Component* {

*// Constructor method*

    constructor() {

*super*()**;**

*// State Declaration*

*this***.***state* **=** {

            title**:** "Learn React"

        }**;**

*// Ref Declaration*

*this***.***foo* **=** "Hei"**;**

        console**.**log('A:'**,** *this***.***foo*)**;**

    }

*// Arrow Function*

    handleButtonClick **=** () **=>** {

        console**.**log('B:'**,** *this***.***foo*)**;**

*this***.***foo* **=** "Hello"**;** *// Change variable value*

        console**.**log('C:'**,** *this***.***foo*)**;**

**if** (*this***.***state***.***title* **===** "Learn React") {

*this***.**setState({ title**:** "Learn JavaScript" })**;**

        }

**else** {

*this***.**setState({ title**:** "Learn React" })**;**

        }

    }

*// Arrow Function*

    handleFooClick **=** () **=>** {

        console**.**log('Get Foo:'**,** *this***.***foo*)**;**

        alert(*this***.***foo*)**;**

    }

    render() {

**return** (

            <div>

                <h3>Title: {*this***.***state***.***title*}</h3>

                <div>

                    <p><button *onClick***=**{*this***.***handleButtonClick*}>Change Title</button></p>

                    <p><button *onClick***=**{*this***.***handleFooClick*}>Get Foo Value</button></p>

                </div>

            </div>

        )

    }

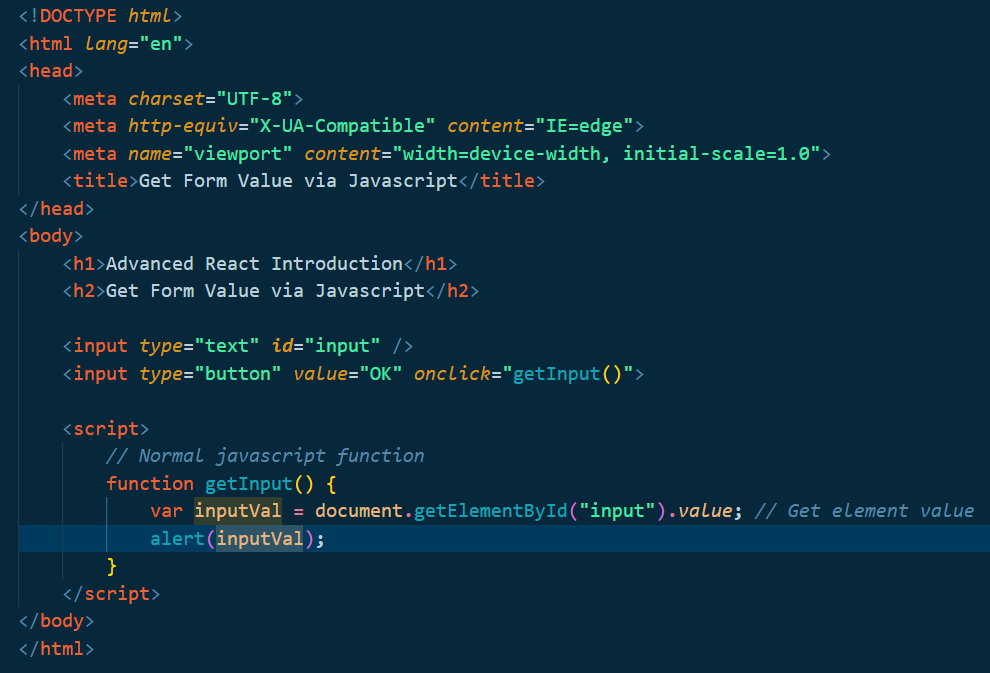
}

const myElement **=** <MyApp />**;**

ReactDOM**.**createRoot(document**.**getElementById('root'))**.**render(myElement)

Ref adalah cara untuk mendapatkan referensi ke elemen DOM atau komponen di dalam pohon render. Hal ini dapat digunakan untuk berbagai tujuan seperti mengakses elemen, mengatur fokus, atau bahkan menghubungkan komponen anak dengan komponen induk.

**7. Form**

****

****

Pada input realtime ini akan merubah bagian judul dengan dengan kata yang di input pada onChange dan akan memanggil function handleInputChange

**8. lifecycle class**

****

komponen kelas memiliki lifecycle yang mencakup berbagai metode yang dipanggil dalam berbagai tahap dalam siklus hidup komponen. Ini memungkinkan kita untuk mengeksekusi kode pada titik-titik tertentu, seperti ketika komponen dimuat, diperbarui, atau dilepas dari DOM.

**9. lifecycle functional**

****

Lifecycle pada react dengan Hooks memungkinkan kita mengendalikan perilaku komponen fungsional. Dengan penggunaan Hooks, kita dapat menggantikan lifecycle kelas komponen