State Management

State Management di React

Apa itu State?

- State adalah data yang dikelola di dalam komponen React.
- Ini adalah bagian penting dalam aplikasi React untuk memastikan UI bereaksi terhadap perubahan data.

State Management di React

Mengapa State Management Penting?

- Menjaga Ul agar tetap konsisten dengan data yang ada.
- Memudahkan pengelolaan data, terutama di aplikasi yang kompleks.

Mengenal `useState`

`useState` adalah hook untuk menambahkan dan mengelola state lokal di dalam komponen fungsional.

Contoh Kode:

```
const [count, setCount] = useState(0)
```

Ini mendefinisikan state `count` dengan nilai awal 0 dan fungsi `setCount` untuk memperbarui nilai tersebut.

useState

Membuat Counter dengan `useState`

```
function Counter() {
  const [count, setCount] = useState(0)

  return (
    <div>
      <p>Anda telah mengklik {count} kali</p>
      <button onClick={() => setCount(count + 1)}>Klik saya</button>
    </div>
  )
}
```

Beralih ke `useReducer`

Apa itu `useReducer`?

`useReducer` adalah alternatif `useState` yang lebih cocok untuk mengelola state yang kompleks.

Membandingkan useState dan `useReducer`

`useState` ideal untuk state yang sederhana, sedangkan `useReducer` lebih sesuai untuk kasus dengan logika state yang lebih kompleks atau state yang terdiri dari beberapa nilai yang saling terkait.

`useReducer`

```
const [state, dispatch] = useReducer(reducer, initialState)
```

Penggunaan dasar `useReducer` dengan `reducer` dan `initialState`.

useReducer

```
const initialState = { count: 0 }

function reducer(state, action) {
  switch (action.type) {
    case 'increment':
      return { count: state.count + 1 }
    case 'decrement':
      return { count: state.count - 1 }
    default:
      throw new Error()
  }
}
```

useReducer

```
function Counter() {
  const [state, dispatch] = useReducer(reducer, initialState)
  return (
    <>
      Count: {state.count}
      <button onClick={() => dispatch({ type: 'increment' })}>+</button>
      <button onClick={() => dispatch({ type: 'decrement' })}>-</button>
    </>
  )
}
```

Pengenalan Redux

Konsep State Global

Redux adalah pustaka yang memungkinkan pengelolaan state global, yang dapat diakses dan dimutasi oleh banyak komponen dalam aplikasi.

Prinsip Dasar Redux

- Konsep utama Redux meliputi Actions, Reducers, dan Store.
- Menggunakan pola Flux untuk memperbarui state.

```
const store = createStore(reducer)
```

Kompleksitas Redux

Boilerplate dan Kompleksitas

Salah satu tantangan menggunakan Redux adalah setup yang lebih rumit dan banyaknya boilerplate code yang diperlukan.

Redux Toolkit untuk Kemudahan

Mengapa Redux Toolkit?

Redux Toolkit dirancang untuk mempermudah setup Redux dan mengurangi jumlah boilerplate code.

Perbandingan dengan Redux Tradisional

Lebih sederhana dan efisien dalam penggunaan dibandingkan Redux tradisional.

```
const store = configureStore({ reducer: rootReducer })
```

Redux Toolkit

```
const counterSlice = createSlice({
  name: 'counter',
  initialState,
  reducers: {
    increment: (state) => {
      state.value += 1
    },
    decrement: (state) => {
      state.value -= 1
    },
  },
})

const { actions, reducer } = counterSlice
const store = configureStore({ reducer })
```