Praktikum 1: Event Handler

Langkah 1

Kita mencoba membuat tombol sederhana yang belum bisa melakukan apa-apa alias belum kita buat event handler untuk tombol tersebut. Sebagai contoh, berikut adalah sebuah tombol yang belum melakukan apa pun. Kita buat folder/file baru di src/component/button.js

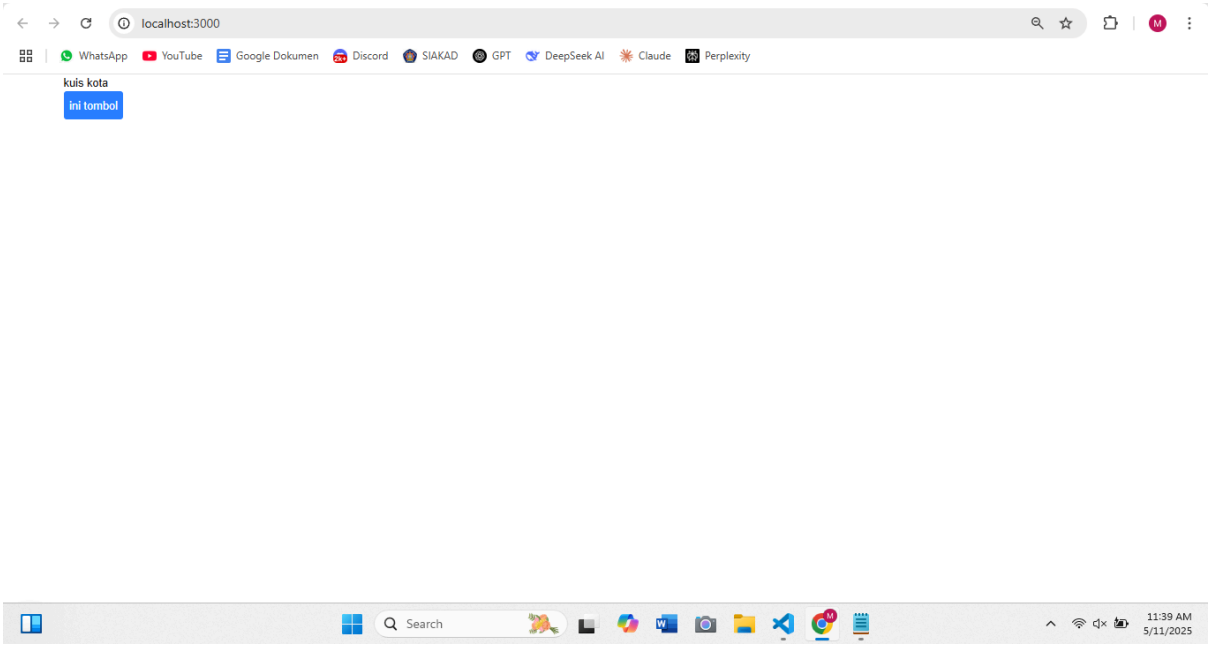
```
export default function Tombol_1() {
  return (
    <button className="bg-blue-500 hover:bg-blue-700
text-white p-2 rounded">ini tombol</button>
  );
}
```

Selanjutnya pada file src/app/page.js kita ubah menjadi seperti berikut

```
import Tombol_1 from "@/components/button";

export default function Home() {
  return (
      <div className="container mx-auto">
      <h2>kuis kota</h2>
      <Tombol_1 />
      </div>
  );
}
```

Kemudian kita jalankan perintah "npm run dev" dan kita buka alamat localhost:3000 pada browser. Maka akan tampil hasil seperti berikut



Langkah 2

Kita bisa menambahkan event pada tombol tersebut. Seperti contoh kita buat ketika tombol di klik, akan memunculkan notif/alert. Kita dapat membuat pesan ketika pengguna mengeklik dengan mengikuti tiga langkah berikut:

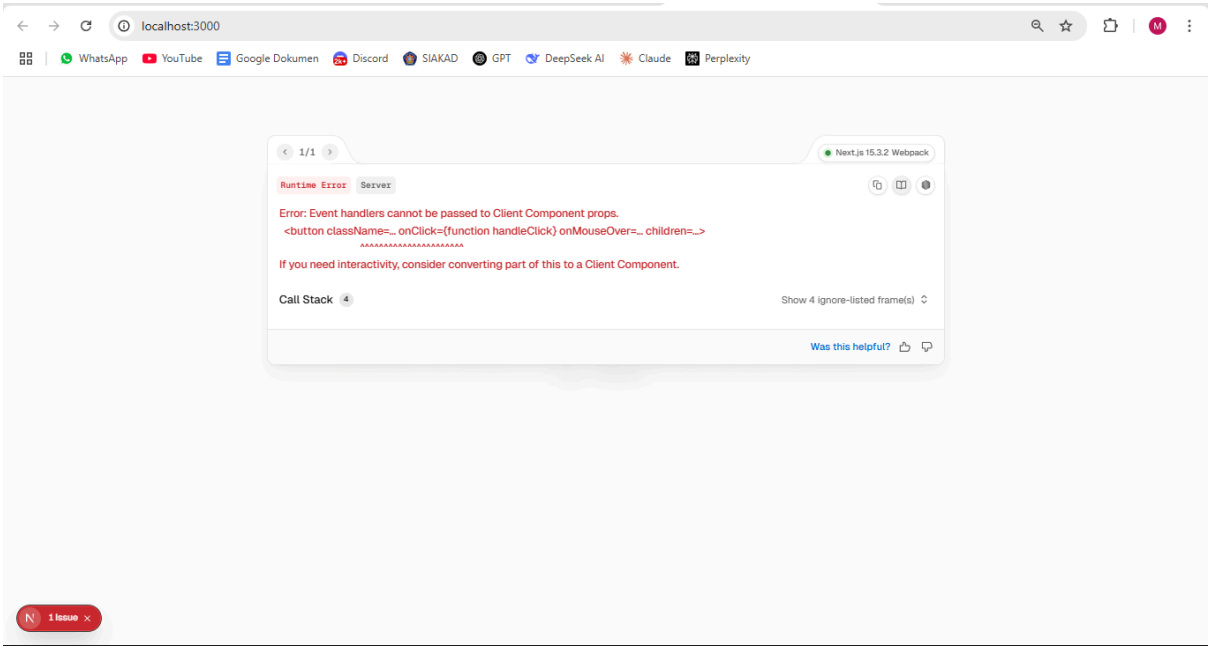
1. Deklarasikan sebuah fungsi bernama handleClick di dalam komponen Button kita.
2. Implementasikan logika di dalam fungsi tersebut (gunakan alert untuk menampilkan pesan).
3. Tambahkan handler onClick={handleClick} ke tag JS < button >

Perhatikan kode button.js berikut

```
export default function Tombol_1() {
  // Menambahkan fungsi untuk menangani klik tombol
  function handleClick() {
    alert("Tombol telah ditekan!!!");
  }
  function handleMouseOver() {
    alert("Eits, mau pencet tombol ini ya??");
  }
```

```
  return (
    <button className="bg-blue-500 hover:bg-blue-700
text-white p-2 rounded"
    onClick={handleClick}
    onMouseOver={handleMouseOver}
    >
    ini tombol
    </button>
  );
}
```

Jika terjadi error seperti gambar berikut



Maka kita butuh mengatur agar komponen yang kita gunakan menjadi komponen client. Untuk menjadikan komponen client, kita cukup memberikan perintah ini "use client"; pada baris pertama file page.js

```
"use client";
import Tombol_1 from "@/components/button";``
export default function Home() {
  return (
      <div className="container mx-auto">
      <h2>kuis kota</h2>
```

```
      <Tombol_1/>
      </div>
  );
}
```

Kita mendefinisikan fungsi handleClick dan kemudian mengopernya sebagai prop ke < button >. Method handleClick adalah sebuah event handler pada tombol tersebut.
Nama Method event handler sebaiknya memiliki format tertentu, seperti contoh memiliki nama yang diawali dengan kata handle, diikuti oleh nama event yang akan dilakukan. Contoh

- event handler untuk menangani ketika ada event klik tombol onClick={handleClick},
- event handler untuk menangani ketika ada event onMouseEnter={handleMouseEnter}, dan lain sebagainya.

Selain itu, sebagai alternatif, Kita juga dapat mendefinisikan *event handler* secara *inline* dalam JS secara langsung seperti berikut:

```
  return (
    <button
    className="bg-blue-500 hover:bg-blue-700 text-white
p-2 rounded"
    onClick={handleClick}
    // onMouseOver={handleMouseOver}
    onMouseLeave={()=> {
      alert("Loh, kok sudah pergi?")
    }
    }
    >
    ini tombol
    </button>
  );
```

> **Ingat!**
>
> Method `event handle` **HARUS** dioper (ditulis nama fungsinya, tanpa tanda kurung `()`), **bukan** dipanggil/*call* (nama fungsi ditulis dengan tanda kurung `()`).

Fungsi di oper (**benar**)	Fungsi dipanggil/call (**salah**)
`< button onClick={handleClick} >`	`< button onClick={handleClick()} >`

Perbedaannya tipis. Pada contoh pertama, fungsi handleClick dioper sebagai event handler onClick. Ini memberitahu React untuk mengingatnya dan hanya memanggil fungsi tersebut ketika pengguna mengklik tombolnya.

Pada contoh kedua, tanda () di akhir handleClick() akan menjalankan fungsi tersebut langsung saat proses render, tanpa adanya klik. Ini karena JavaScript di dalam tag JSX { dan } dieksekusi secara langsung.

Praktikum 2

Kita buat fungsi baru pada component button.js

```
export function Tombol_2({isipesan, namaTombol}) {
  return (
    <button className="bg-blue-500 hover:bg-blue-700
text-white p-2 rounded"
    onClick={() => {
      alert(isipesan);
    }}>
      {namaTombol}
    </button>
  );
}

export default function Tombol_1() {
  // Menambahkan fungsi untuk menangani klik tombol
  function handleClick() {
    alert("Tombol telah ditekan!!!");
  }
  function handleMouseOver() {
    alert("Eits, mau pencet tombol ini ya??");
  }
  return (
    <button
    className="bg-blue-500 hover:bg-blue-700 text-white
p-2 rounded"
    onClick={handleClick}
    // onMouseOver={handleMouseOver}
    onMouseLeave={()=> {
      alert("Loh, kok sudah pergi?")
    }}>
    ini tombol
    </button>
```

```
  );
}
```

> **Ingat!**
>
> Pada `component`, hanya ada 1 fungsi yang memiliki `default` !

Parameter isiPesan dan namaTombol bisa diisi oleh layout yang ada di page.js nanti, sehingga komponen Tombol_2 bernilai dinamis.

Sekarang kita modifikasi file page.js seperti berikut

```
"use client";
import Tombol_1, { Tombol_2 } from "../components/button";

export default function Home() {
  return (
      <div className="container mx-auto">
      <h2>kuis kota</h2>
      <Tombol_1 />
      <hr /><hr /><br/>
      <Tombol_2 isipesan="Ini pesanku"
namaTombol="Pesan"/>
      </div>
  );
}
```

Coba cek di browser dan amati apa yang terjadi?

Praktikum 3

Langkah 1 - Propagation

Sebagai contoh coba kita modifikasi file button.js seperti berikut

```
export function Tombol_2({isipesan, namaTombol}) {
  return (
    <button className="bg-blue-500 hover:bg-blue-700
text-white p-2 rounded"
    onClick={() => {alert(isipesan)}}>
      {namaTombol}
    </button>
  );
}

export function Tombol_3({isipesan, namaTombol}) {
  return (
    <button
    className="bg-green-400 hover:bg-green-700 text-white
p-2 rounded m-2"
    onClick={() => {
        alert(isipesan);
    }}>
      {namaTombol}
    </button>
  );
}

export default function Tombol_1() {
  // Menambahkan fungsi untuk menangani klik tombol
  function handleClick() {
    alert("Tombol telah ditekan!!!");
  }
  function handleMouseOver() {
```

```
    alert("Eits, mau pencet tombol ini ya??");
  }
  return (
    <button
    className="bg-blue-500 hover:bg-blue-700 text-white
p-2 rounded"
    onClick={handleClick}
    // onMouseOver={handleMouseOver}
    onMouseLeave={()=> {
      alert("Loh, kok sudah pergi?")
    }}>
    ini tombol
    </button>
  );
}
```

Kemudian kita modifikasi fiile page.js

```
"use client";
import Tombol_1, { Tombol_2, Tombol_3 } from
"../components/button";
export default function Home() {
  return (
    <>
      <div className="container mx-auto">
      <h2>kuis kota</h2>
      <Tombol_1 />
      <hr /><hr /><br/>
      <Tombol_2 isipesan="Ini pesanku"
namaTombol="Pesan"/>
      </div>

      <br /><br />
      <div className="bg-red-300"
onClick={()=>alert("Parent Element : Div")}>
```

```
        <Tombol_3 isipesan="Child Element : Tombol-1"
namaTombol="Tombol-1"/>
        <Tombol_3 isipesan="Child Element : Tombol-2"
namaTombol="Tombol-2"/>
      </div>
    </>
  );
}
```

Kemudian kita jalankan di browser, coba klik Tombol-1, dan amati apa yang terjadi?
Kita akan disuguhkan dengan pesan/alert sebanyak 2 kali, yaitu Pesan "Child Element : Tombol-1" dan pesan "Parent Element : Div".
Hal ini terjadi karena baik untuk element div maupun button memiliki event yang sama yaitu onClick, sehingga ketika button diklik maka event handler untuk onClick pada button akan dijalankan. Kemudian baru event handler dari parent (element div) akan dijalankan.

Hal ini disebut dengan propagation, dan biasa terjadi pada elemen *child* dan *parent* yang memiliki *event* yang sama.

Menghentikan *Propagation*

Event handler menerima sebuah objek event sebagai satu-satunya argumen/parameter. Berdasarkan konvensi, objek tersebut biasanya ditulis e yang merupakan kepanjangan dari "event". Anda dapat menggunakan objek ini untuk membaca informasi tentang event tersebut.

Objek event tersebut juga dapat memungkinkan untuk menghentikan propagasi. Jika kita ingin mencegah sebuah event untuk mencapai komponen induknya (propagation), Kita harus memanggil e.stopPropagation() untuk mencegah propagasi.

Langkah 2 - Stop Propagation

```
export function Tombol_3({isipesan, namaTombol}) {
  return (
    <button
```

```
    className="bg-green-400 hover:bg-green-700 text-white
p-2 rounded m-2"
    onClick={(e) => {
        e.stopPropagation();
        alert(isipesan);
      }
    }>
      {namaTombol}
    </button>
  );
}
```

Praktikum 4

Langkah 1

Kita buat file data dummy untuk mencoba state pada src/data/article.js yang berisi seperti berikut

```
export const sculptureList = [{

    name: 'Homenaje a la Neurocirugía',

    artist: 'Marta Colvin Andrade',

    description: 'Although Colvin is predominantly known for abstract themes that allude to pre-Hispanic symbols, this gigantic sculpture, an homage to neurosurgery, is one of her most recognizable public art pieces.',

    url: 'https://i.imgur.com/Mx7dA2Y.jpg',

    alt: 'A bronze statue of two crossed hands delicately holding a human brain in their fingertips.'

  }, {

    name: 'Floralis Genérica',

    artist: 'Eduardo Catalano',

    description: 'This enormous (75 ft. or 23m) silver flower is located in Buenos Aires. It is designed to move, closing its petals in the evening or when strong winds blow and opening them in the morning.',

    url: 'https://i.imgur.com/ZF6s192m.jpg',

    alt: 'A gigantic metallic flower sculpture with reflective mirror-like petals and strong stamens.'

  }, {
```

```
    name: 'Eternal Presence',
    artist: 'John Woodrow Wilson',
    description: 'Wilson was known for his preoccupation with equality, social justice, as well as the essential and spiritual qualities of humankind. This massive (7ft. or 2,13m) bronze represents what he described as "a symbolic Black presence infused with a sense of universal humanity."',
    url: 'https://i.imgur.com/aTtVpES.jpg',
    alt: 'The sculpture depicting a human head seems ever-present and solemn. It radiates calm and serenity.'
  }, {
    name: 'Moai',
    artist: 'Unknown Artist',
    description: 'Located on the Easter Island, there are 1,000 moai, or extant monumental statues, created by the early Rapa Nui people, which some believe represented deified ancestors.',
    url: 'https://i.imgur.com/RCwLEoQm.jpg',
    alt: 'Three monumental stone busts with the heads that are disproportionately large with somber faces.'
  }, {
    name: 'Blue Nana',
    artist: 'Niki de Saint Phalle',
```

```
    description: 'The Nanas are triumphant creatures,
symbols of femininity and maternity. Initially, Saint
Phalle used fabric and found objects for the Nanas, and
later on introduced polyester to achieve a more vibrant
effect.',
    url: 'https://i.imgur.com/Sd1AgUOm.jpg',
    alt: 'A large mosaic sculpture of a whimsical dancing
female figure in a colorful costume emanating joy.'
  }];
```

Kemudian kita coba buat komponen baru di src/component/galeri.js

```
import { sculptureList } from "@/data/article"; //ambil
data dari file article.js


export default function Galeri() {
 let index = 0; // indeex data mulai dari 0
  function handleClick() {
    index = index + 1; // counter index bertambah 1, untuk
melihat data selanjutnya
  }
  let sculpture = sculptureList[index]; // membaca data
sesuai dengan index
  return(
    <>
    <button
```

```
    onClick={handleClick}
    className="bg-blue-500 hover:bg-blue-700 text-white
p-2 rounded">Artikel selanjutnya</button>
    <h2>{sculpture.name} <i/> oleh {sculpture.artist}</h2>
    <h3>({index + 1} dari {sculptureList.length})</h3>
    <img src={sculpture.url} alt={sculpture.alt}/>
    <p>
        {sculpture.description}
    </p>
    </>
  )
}
```

Kita panggil komponen tersebut pada page.js

```
"use client";
import Tombol_1, { Tombol_2, Tombol_3 } from
"../components/button";
import Galeri from "../components/galeri";

export default function Home() {
  return (
    <>
      <div className="container mx-auto">
```

```
      <h2>kuis kota</h2>

      <Tombol_1 />

      <hr /><hr /><br/>

      <Tombol_2 isipesan="Ini pesanku"
namaTombol="Pesan"/>

      </div>



      <br /><br />

      <div className="bg-red-300"
onClick={()=>alert("Parent Element : Div")}>

        <Tombol_3 isipesan="Child Element : Tombol-1"
namaTombol="Tombol-1"/>

        <Tombol_3 isipesan="Child Element : Tombol-2"
namaTombol="Tombol-2"/>

      </div>

      <br /><br />

        <Galeri />

    </>

  );

}
```

Sekarang coba di browser dan klik tombol "Artikel Selanjutnya" dan perhatikan apa yang terjadi?

Ya, tidak terjadi apa-apa 😀.

Event handler handleClick memperbarui nilai variabel index. Namun dua hal mencegah pembaruan tersebut ditampilkan ke pengguna:

1. Variabel lokal tidak dipertahankan antar-render. Saat React me-render komponen ini untuk kedua kalinya, react membuat ulang dari awal, sehingga index tetap bernilai 0 dan react tidak memperhatikan adanya perubahan ke variabel index tersebut.
2. Perubahan terhadap variabel lokal tidak memicu *render*. React tidak menyadari kalau dia perlu melakukan *render* ulang dengan data yang baru.

Untuk memperbarui komponen dengan data baru, dua hal perlu terjadi:

1. Mempertahankan data antar-*render*.
2. Memicu React untuk merender ulang komponennya dengan data baru.

Dua hal tersebut bisa dicapai dengan HookuseState:

1. Sebuah variabel state untuk mempertahankan data antar-*render*.
2. Sebuah fungsi *state* setter untuk memperbarui variabel dan memicu React untuk merender ulang komponen.

Langkah 2: Menambahkan variabel *state*

Untuk menambahkan variabel state, import useState dari React di paling atas file src/components/galeri.js

Untuk menambahkan variabel state, import useState dari React di paling atas file src/components/galeri.js.

```
import { useState } from 'react';
```

Lalu, ubah baris berikut:

```
let index = 0;
```

Menjadi

```
const [index, setIndex] = useState(0);
```

index merupakan variabel state dan setIndex adalah fungsi setter.

Ubah fungsi dalam handleClick menjadi seperti ini

```
  function handleNextClick() {
    // Cek apakah index belum mencapai batas akhir
    if (index < sculptureList.length - 1) {
      setIndex(index + 1);
    }
  }
```

Maka kode pada galeri.js seperti berikut

```
import { useState } from 'react';
import { sculptureList } from "@/data/article";

export default function Galeri() {
  const [index, setIndex] = useState(0);

  function handleNextClick() {
    // Cek apakah index belum mencapai batas akhir
    if (index < sculptureList.length - 1) {
      setIndex(index + 1);
    }
  }
```

```
  function handlePrevClick() {
    // Cek apakah index lebih dari 0
    if (index > 0) {
      setIndex(index - 1);
    }
  }

  const sculpture = sculptureList[index];

  return (
    <>
      <div className="space-x-2 mb-4">
        <button
          onClick={handlePrevClick}
          className="bg-gray-500 hover:bg-gray-700
text-white p-2 rounded"
          disabled={index === 0}
        >
          Artikel Sebelumnya
        </button>
        <button
```

```
          onClick={handleNextClick}
          className="bg-blue-500 hover:bg-blue-700
text-white p-2 rounded"
          disabled={index === sculptureList.length - 1}
        >
          Artikel Selanjutnya
        </button>
      </div>

      <h2>{sculpture.name} <i /> oleh
{sculpture.artist}</h2>
      <h3>({index + 1} dari {sculptureList.length})</h3>
      <img src={sculpture.url} alt={sculpture.alt} />
      <p>{sculpture.description}</p>
    </>
  );
}
```

Jalankan pada browser

Soal

1. Jika kita menekan tombol "Artikel Selanjutnya" sebanyak 5x (atau melebihi halaman total artikel), apa yang akan terjadi?
2. Modifikasilah galeri.js agar bisa meng-*handle* permasalahan tersebut.

3. Tambahkan tombol "Artikel Sebelumnya", untuk menampilkan artikel secara mundur.

Praktikum 5

Langkah 1

Kita buat komponen baru src/components/form.js

```
import React, { useState } from "react";



export default function Form() {

  const [jawaban, setJawaban] = useState('');

  const [error, setError] = useState(null);

  const [status, setStatus] = useState('typing');



  if (status === 'success') {

    return React.createElement('h1', null, 'Yay... Jawaban
Benar!');

  }



  function handleTextareaChange(e) {

    setJawaban(e.target.value);

  }



  async function handleSubmit(e) {

    e.preventDefault();

    setStatus('submitting');
```

```
    try {
      await submitForm(jawaban);
      setStatus('success');
    } catch (err) {
      setStatus('typing');
      setError(err);
    }
  }

  return React.createElement(
    'div',
    { className: 'w-full max-w-xs' },
    React.createElement('h2', null, 'Tebak Nama Hewan'),
    React.createElement('p', null, 'Hewan apa yang
ditakuti oleh doraemon?'),
    React.createElement(
      'form',
      {
        className: 'shadow-md rounded px-8 pt-6 pb-8 mb-4
text-black border-gray-400',
        onSubmit: handleSubmit,
      },
```

```
      React.createElement('textarea', {
        value: jawaban,
        onChange: handleTextareaChange,
        disabled: status === 'submitting',
        className: 'w-full p-2 border border-gray-300
rounded mb-4',
      }),
      React.createElement('br'),
      React.createElement(
        'button',
        {
          type: 'submit',
          className: 'bg-blue-400 p-2 m-2 rounded text-sm
text-white',
          disabled: jawaban.length === 0 || status ===
'submitting',
        },
        'Submit'
      ),
      error &&
        React.createElement(
          'p',
          { className: 'text-red-500 text-sm' },
```

```
          error.message
        )
    )
  );
}


// Fungsi submitForm: simulasi request
function submitForm(jawaban) {
  return new Promise((resolve, reject) => {
    setTimeout(() => {
      const shouldError = jawaban.toLowerCase() !==
'tikus';
      if (shouldError) {
        reject(new Error('Tebakan yang bagus tetapi
jawaban salah. Silahkan coba lagi!'));
      } else {
        resolve();
      }
    }, 500); // set timeout selama 0,5 detik
  });
}
```

Kemudian kita tambahkan kode pada file page.js

```
"use client";

import Tombol_1, { Tombol_2, Tombol_3 } from
"../components/button";

import Galeri from "../components/galeri";

import Form from "../components/form";



export default function Home() {

  return (

    <>

      <div className="container mx-auto">

      <h2>kuis kota</h2>

      <Tombol_1 />

      <hr /><hr /><br/>

      <Tombol_2 isipesan="Ini pesanku"
namaTombol="Pesan"/>

      </div>



      <br /><br />

      <div className="bg-red-300"
onClick={()=>alert("Parent Element : Div")}>
```

```
        <Tombol_3 isipesan="Child Element : Tombol-1"
namaTombol="Tombol-1"/>

        <Tombol_3 isipesan="Child Element : Tombol-2"
namaTombol="Tombol-2"/>

      </div>


      <br /><br />


        <Galeri />


      <br /><br />


        <Form />

    </>

  );

}
```

Jalankan pada browser, amati dan laporkan apa yang terjadi?

Praktikum 6

Langkah 1

Kita buat file komponen pada src/components/accordion.js

```
import { useState } from 'react';



export default function Accordion() {

    const [activeIndex, setActiveIndex] = useState(0);

    return (

        <>

            <h2>Almaty, Kazakhstan</h2>

            <Panel

                title="About"

                isActive={activeIndex === 0}

                onShow={() => setActiveIndex(0)}

            >

                Dengan populasi sekitar 2 juta orang, Almaty adalah kota terbesar di Kazakhstan. Dari tahun 1929 hingga 1997, kota ini menjadi ibu kota Kazakhstan.

            </Panel>

            <Panel

                title="Etymology"

                isActive={activeIndex === 1}
```

```
                onShow={() => setActiveIndex(1)}
            >
                Nama "Almaty" berasal dari kata <span
lang="kk-KZ">alma</span>, dalam bahasa Kazakh yang berarti
"apel" dan sering diterjemahkan sebagai "penuh dengan
apel". Sebenarnya, wilayah sekitar Almaty dipercaya
sebagai asal usul apel, dan <i lang="la">Malus
sieversii</i> liar dianggap sebagai kandidat yang mungkin
menjadi nenek moyang apel domestik modern.
            </Panel>
        </>
    );
}

function Panel({ title, children, isActive, onShow }) {
    return (
        <section className="panel border border-gray-700
p-2">
            <h3>{title}</h3>
            {isActive ? (
                <p>{children}</p>
            ) : (
                <button className="bg-blue-400 text-xs
text-white p-1 rounded m-2" onClick={onShow}>
```

```
                    Tampilkan
                </button>
            )}
        </section>
    );
}
```

Lalu kita tambahkan component Accordion ke file page.js