

Laporan Praktikum Minggu-4
Mata Kuliah Pemrograman Berbasis Framework
“MODUL 4: API”



Nama Penyusun:

Khosy Robbin Hood (1941720067) / TI3D

JURUSAN TEKNOLOGI INFORMASI
PROGRAM STUDI D-IV TEKNIK INFORMATIKA
MARET 2022

Praktikum 1

Interaksi dengan API menggunakan method GET

1.3 PRAKTIKUM

1. Buka Project React pada pertemuan sebelumnya dan jalankan “npm start” menggunakan cmd dalam direktori tersebut.
2. Buat folder baru bernama “BlogPost” pada folder container (statefull component).
3. Buat file BlogPost.jsx dan BlogPost.css di dalam folder “BlogPost”.
4. Buka file BlogPost.jsx dan ketikkan kode

```
import React, {Component} from "react";
import './BlogPost.css'

class BlogPost extends Component{
  render() {
    return(
      <p>Blog Artikel</p>
    )
  }
}

export default BlogPost;
```

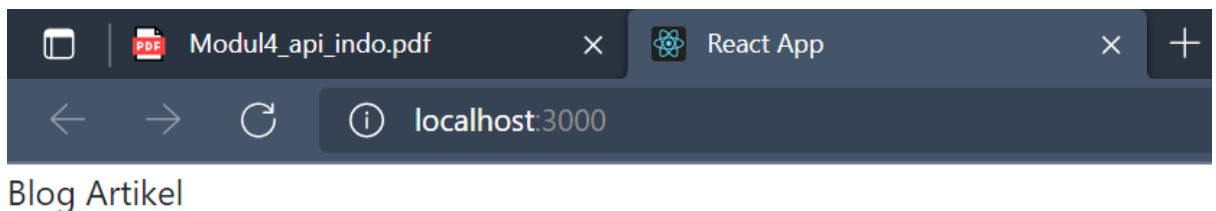
5. Pada file index.js, lakukan import component BlogPost

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
import BlogPost from './container/BlogPost/BlogPost';
import 'bootstrap/dist/css/bootstrap.min.css'

ReactDOM.render(<BlogPost/>, document.getElementById('root'));

reportWebVitals();
```

6. Pada web browser akan tampil seperti ini



7. Import css bootstrap.min.css (css bootstrap yang sudah dikompresi) ke dalam index.js (seperti Gambar 1.6). Jika css tidak ditemukan, install lewat cmd dengan perintah “npm install bootstrap”

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import reportWebVitals from './reportWebVitals';
import BlogPost from './container/BlogPost/BlogPost';
import 'bootstrap/dist/css/bootstrap.min.css'

ReactDOM.render(<BlogPost/>, document.getElementById('root'));

reportWebVitals();
```

8. Modifikasi file index.html pada folder "public" seperti Gambar 1.7. Cermati code program yang ada dalam gambar!.

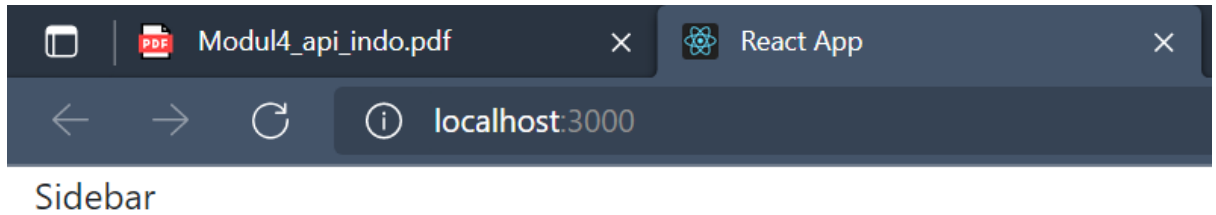
```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8" />
  <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
  <meta name="viewport" content="width=device-width,
initial-scale=1" />
  <meta name="theme-color" content="#000000" />
  <meta name="description" content="Web site created using
create-react-app" />
  <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
  <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
  <title>React App</title>
</head>

<body>
  <noscript>You need to enable JavaScript to run this
app.</noscript>
  <div class="container-fluid">
    <div class="row">
      <div class="col-2" id="sidebar">Sidebar</div>
      <div class="col-10" id="content"></div>
    </div>
  </div>
</body>
```

```
</html>
```

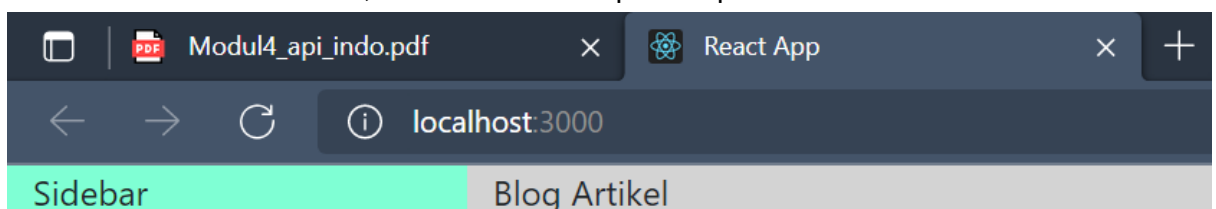
9. Amati tampilan yang ada pada browser



10. Buka file index.css dan tambahkan code css seperti Gambar 1.9, untuk menambah sedikit style pada halaman web

```
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;
}
#sidebar{
  background-color: aquamarine;
}
#content{
  background-color: lightgray;
}
```

11. Perhatikan kembali browser, dan lihat hasil tampilan seperti Gambar 1.10.



12. Ubah kode program untuk statefull component BlogPost.jsx menjadi seperti Gambar 1.11

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

```
import './BlogPost.css'

class BlogPost extends Component{
  render() {
    return(
      <div class="post-artikel">
        <h2>Daftar Artikel</h2>
        <div class="artikel">
          <div class="gambar artikel">
            
          </div>
          <div class="konten-artikel">
            <div class="judul-artikel">Judul
Artikel</div>
            <p class="isi-artikel">Isi Artikel</p>
          </div>
        </div>
      </div>
    )
  }
}

export default BlogPost;
```

13. Tambahkan custom css ke BlogPost.css seperti Gambar 1.12

```
.artikel{
  width: 100%;
  padding: 10px;
  border: 1px solid blue;
  border-radius: 4px;
  margin-bottom: 10px;
  box-shadow: 0 0 16px rgba(0, 0, 0, 0.5);
  display: flex;
}

.gambar-artikel{
  height: 80px;
  width: 80px;
  margin-right: 20px;
  vertical-align: top;
}
```

```

.gambar-artikel img{
  width: 100%;
  height: 100%;
  object-fit: cover;
}

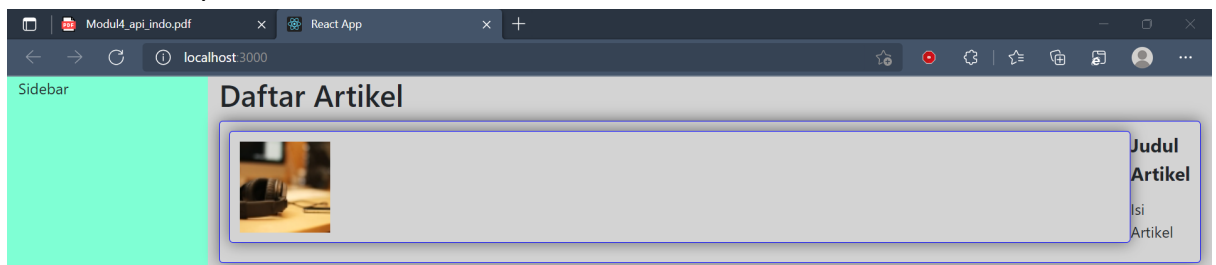
.konten-artikel{
  flex: 1;
}

.konten-artikel div.judul-artikel{
  font-size: 20px;
  font-weight: bold;
  margin-bottom: 10px;
}

.konten-artikel p.isi-artikel{
  font-size: 16px;
  margin-bottom: 10px;
}

```

14. Perhatikan tampilan browser.



15. Buat folder BlogPost pada folder component (stateless component), lalu buat file Post.jsx

16. Potong (cut) baris 9-17 pada statefull component BlogPost.jsx ke stateless component Post.jsx, dan modifikasi Post.jsx seperti Gambar 1.13.

```

import React from "react";

const Post = (props) => {
  return (
    <div class="artikel">
      <div class="gambar artikel">
        
      </div>
      <div class="konten-artikel">
        <div class="judul-artikel">Judul Artikel</div>

```

```

        <p class="isi-artikel">Isi Artikel</p>
      </div>
    </div>

  )
}

export default Post;

```

17. Untuk statefull component BlogPost.jsx pada baris 10, panggil stateless component Post.jsx seperti Gambar 1.14.

```

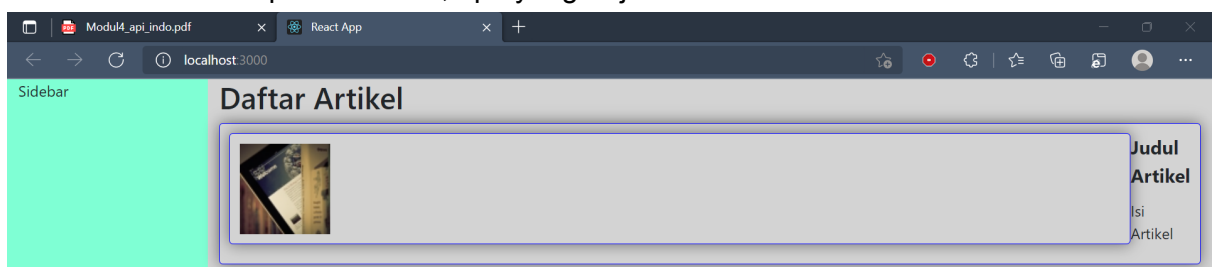
import React, {Component} from "react";
import './BlogPost.css'
import Post from "../../component/BlogPost/Post";

class BlogPost extends Component{
  render() {
    return(
      <div class="post-artikel">
        <h2>Daftar Artikel</h2>
        <Post/>
      </div>
    )
  }
}

export default BlogPost;

```

18. Perhatikan hasil tampilan browser, apa yang terjadi?



19. Pada statefull component BlogPost.jsx, tambahkan parameter yang ingin dilempar ke stateless component untuk ditampilkan. Kode program bisa dilihat pada Gambar 1.15.

```

import React, {Component} from "react";
import './BlogPost.css'
import Post from "../../component/BlogPost/Post";

class BlogPost extends Component{

```

```

render() {
  return(
    <div class="post-artikel">
      <h2>Daftar Artikel</h2>
      <Post judul="JTI POLINEMA" isi="Jurusan Teknologi
Informasi - Politeknik Negeri Malang"/>
    </div>
  )
}
}

export default BlogPost;

```

20. Setelah itu pada stateless component Post.jsx tangkap parameter yang dilempar oleh statefull component seperti pada Gambar 1.16 dan lihat pada browser apa yang terjadi!.

```

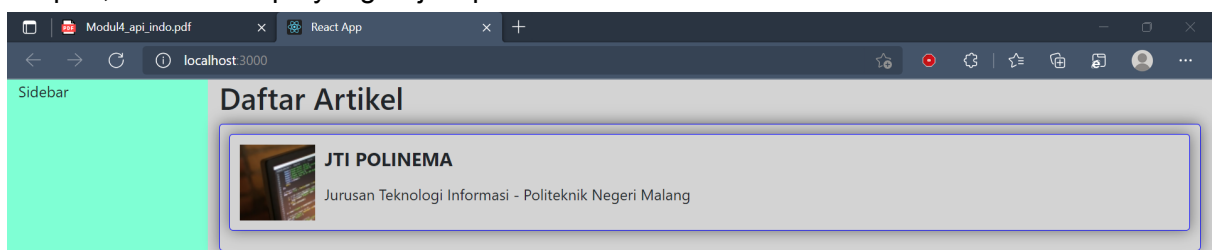
import React from "react";

const Post = (props) => {
  return (
    <div class="artikel">
      <div class="gambar artikel">
        
        <div class="konten-artikel">
          <div
class="judul-artikel">{props.judul}</div>
          <p class="isi-artikel">{props.isi}</p>
        </div>
      </div>
    </div>
  )
}

export default Post;

```

21. Simpan, dan amati apa yang terjadi pada browser kalian!.



22. Gunakan state untuk menyimpan data hasil request dari API
23. data API yang akan kita gunakan adalah data dummy dari <https://jsonplaceholder.typicode.com/posts>, dimana memiliki 4 element data yaitu userid, id, title, body (seperti pada Gambar 1.17)
24. Edit pada statefull component BlogPost.jsx seperti pada Gambar 1.18 dan perhatikan dengan seksama akan penjelasan di beberapa baris kode program tersebut.

```
import React, {Component} from "react";
import './BlogPost.css'
import Post from "../../component/BlogPost/Post";

class BlogPost extends Component{
  state = { //Komponen state dari React untuk
statefull component
    listArtikel: [] //variabel array untuk menyimpan data
API
  }

  componentDidMount(){ //komponen yg dicek ketika telah diambil
di-mount-ing
    fetch('https://jsonplaceholder.typicode.com/posts')
//alamat API
    .then(response => response.json())
    .then(jsonHasilAmbilDariAPI => {
      this.setState({
        listArtikel: jsonHasilAmbilDariAPI
      })
    })
  }

  render() {
    return(
      <div class="post-artikel">
        <h2>Daftar Artikel</h2>
        {
          this.state.listArtikel.map(artikel => {
//lopping dan masukan data ke variabel artikel
            return <Post judul={artikel.title}
isi={artikel.body}/> //mapping data JSON sesuai kategori
          })
        }
        { /* <Post judul="JTI POLINEMA" isi="Jurusan
Teknologi Informasi - Politeknik Negeri Malang"/> */}
      </div>
    )
  }
}
```

```

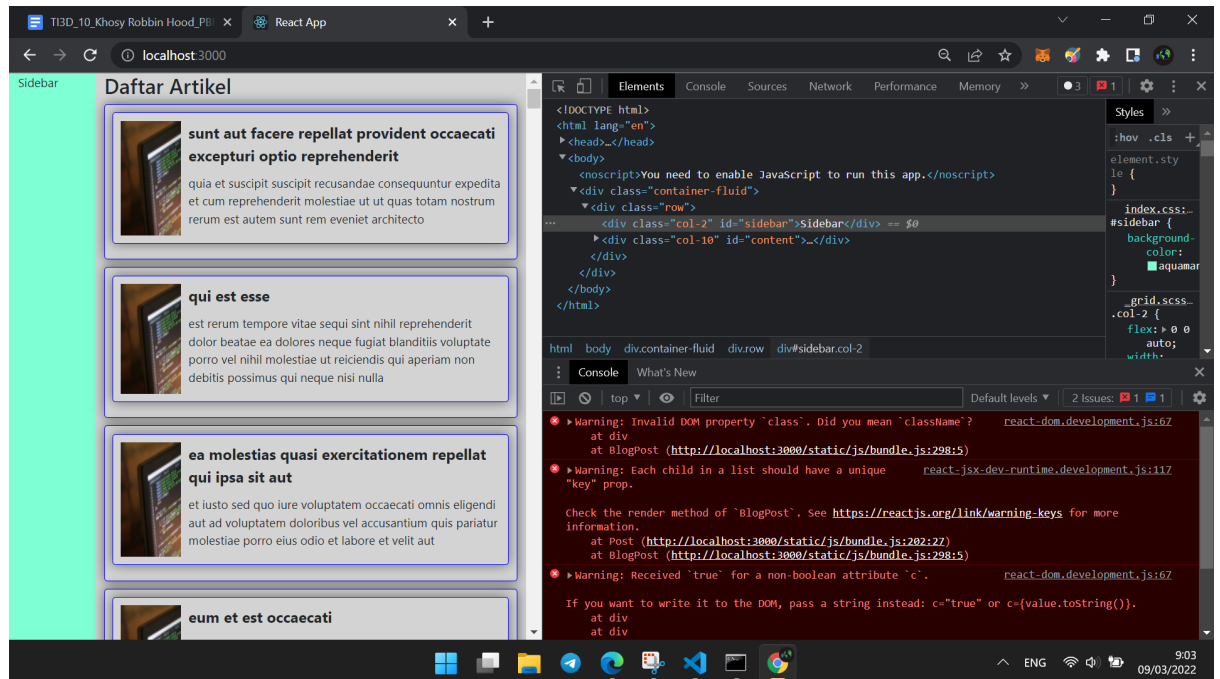
}

}

export default BlogPost;

```

25. Lihat hasilnya pada browser. Kemudian klik kanan pada browser pilih "inspect element" kemudian pilih tab "console". Refresh browser dan amati apa yang terjadi.



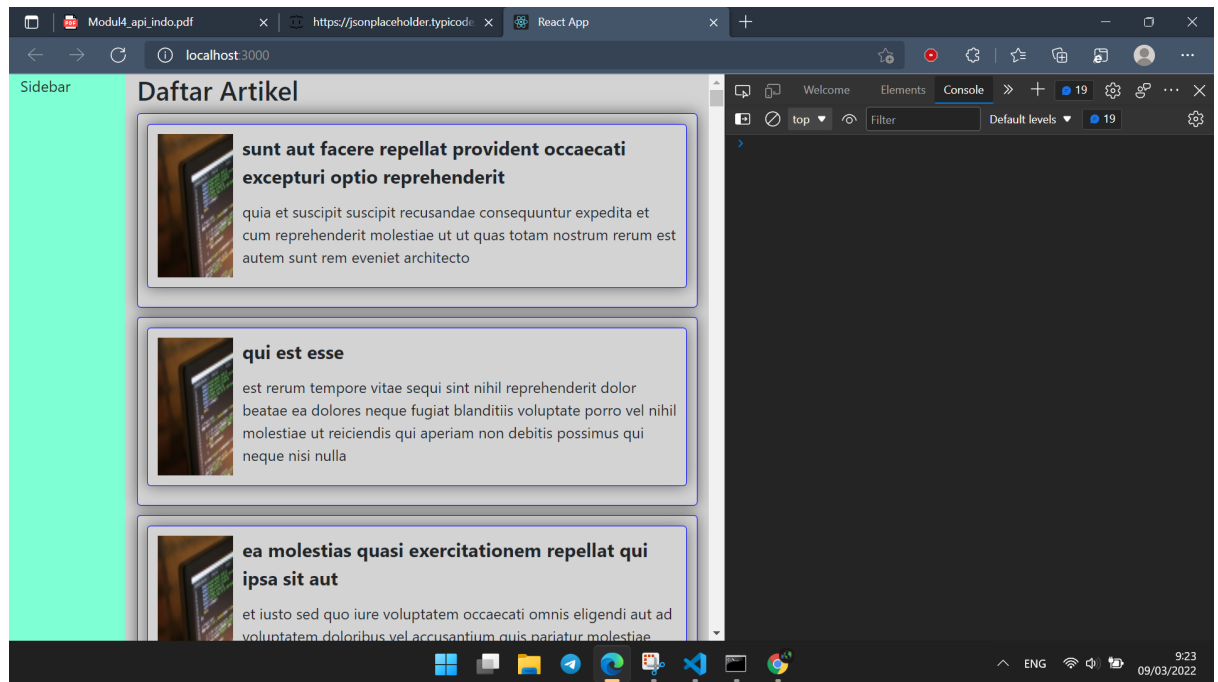
26. Jika terlihat seperti pada Gambar 1.19, maka terjadi kesalahan pada program yang kita buat.
27. Jika terjadi hal demikian, hal ini terjadi karena dalam react "class" dalam tag html harus ditulis menjadi "className". selain itu, pada statefull component yang dinamis, harus ada "UNIQUE KEY" pada tiap komponen yang diproses sehingga komponen perlu diberi UNIQUE KEY.
28. UNIQUE KEY dapat diambil dari element yang ada pada data API yang sudah kita ambil (contoh saat ini adalah element id pada data API (userid, id, title, body) yang akan kita gunakan untuk UNIQUE KEY.

```

render() {
  return(
    <div className="post-artikel">
      <h2>Daftar Artikel</h2>
      {
        this.state.listArtikel.map(artikel => {
//lopping dan masukan data ke variabel artikel
          return <Post key={artikel.id}
judul={artikel.title} isi={artikel.body}/> //mapping data JSON
sesuai kategori
        })
      }
    )
  }
}

```

29. Simpan dan lihat apa yang terjadi pada console browser (Gambar 1.21).



1.4 Pertanyaan Praktikum 1

a. Pada langkah 8, sekarang coba kalian ganti class container dengan container-fluid atau sebaliknya pada file "public/index.html" dan lihat apa perbedaannya.

1. Tampilan seperti apa yang kalian temukan setelah mencoba mengganti nama class tersebut?

→ akan memenuhi ukuran layar sesuai perangkat

2. Apa perbedaan dari container dan container-fluid ?

→ container-fluid memiliki ukuran lebar kontainer memenuhi lebar layar (full width). Dengan menggunakan class ini maka kontainer yang anda buat akan memenuhi ukuran layar dari semua perangkat yang anda gunakan.

b. Jika kita ingin meng-import suatu component contoh component bootstrap, akan tetapi component dalam tersebut belum terdapat pada module ReactJS. Apa yang akan dilakukan untuk dapat menggunakan component tersebut? Bagaimana caranya?

→ dengan import mandiri dengan cara "import 'bootstrap/dist/css/bootstrap.min.css';"

PRAKTIKUM 2

Interaksi dengan API menggunakan Fake API

2.1 Install Fake API (JSON Server)

1. Install pada direktori project reactjs kita dengan perintah `npm install -g json-server`
2. Copy-kan file json `listArtikel.json` yang sudah ada pada direktori project reactjs kita.
3. Buka cmd baru pada direktori project, lalu ketik perintah `json-server --watch listArtikel.json --port 3001`.
4. Apabila pada cmd tampil seperti Gambar 2.1, maka server Fake API local kita telah siap

```
E:\SMSTR 6\Pemogramran Berbasis Framework\pertemuan 4\blog-post>json-server -p 3001 -w listArtikel.json

\{^_^}/ hi!

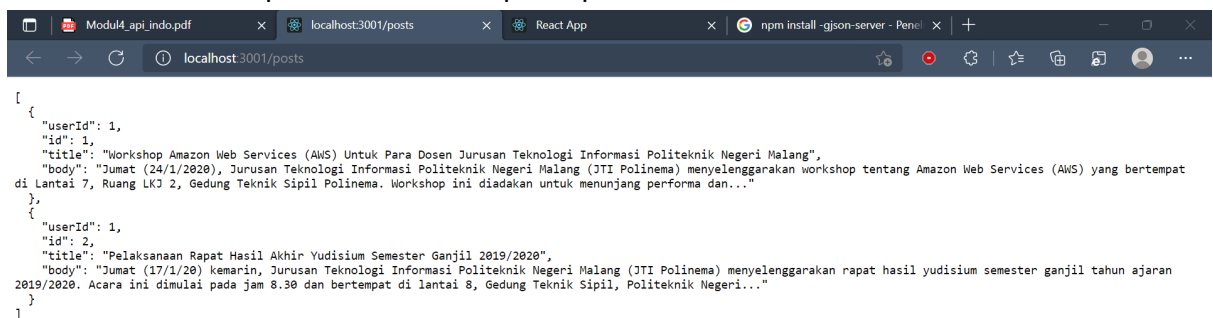
Loading listArtikel.json
Done

Resources
http://localhost:3001/posts

Home
http://localhost:3001

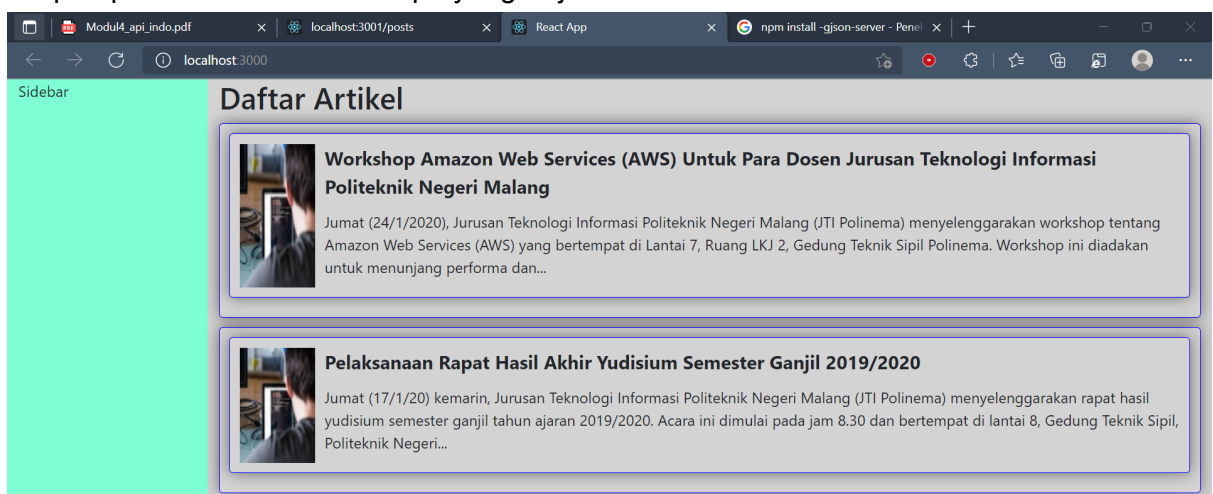
Type s + enter at any time to create a snapshot of the database
Watching...
```

5. Kita cek url resource yang ada pada Fake API server ke browser apakah bisa diakses. Ketik url `http://localhost:3001/posts` pada browser



```
[
  {
    "userId": 1,
    "id": 1,
    "title": "Workshop Amazon Web Services (AWS) Untuk Para Dosen Jurusan Teknologi Informasi Politeknik Negeri Malang",
    "body": "Jumat (24/1/2020), Jurusan Teknologi Informasi Politeknik Negeri Malang (JTI Polinema) menyelenggarakan workshop tentang Amazon Web Services (AWS) yang bertempat di Lantai 7, Ruang LKJ 2, Gedung Teknik Sipil Polinema. Workshop ini diadakan untuk menunjang performa dan..."
  },
  {
    "userId": 1,
    "id": 2,
    "title": "Pelaksanaan Rapat Hasil Akhir Yudisium Semester Ganjil 2019/2020",
    "body": "Jumat (17/1/20) kemarin, Jurusan Teknologi Informasi Politeknik Negeri Malang (JTI Polinema) menyelenggarakan rapat hasil yudisium semester ganjil tahun ajaran 2019/2020. Acara ini dimulai pada jam 8.30 dan bertempat di lantai 8, Gedung Teknik Sipil, Politeknik Negeri..."
  }
]
```

6. Untuk memastikan lagi, kita edit statefull component `BlogPost` (Gambar 1.18) pada baris 11. Kita ganti url API dari `https://jsonplaceholder.typicode.com/posts` menjadi `http://localhost:3001/posts`
7. Simpan perubahan dan amati apa yang terjadi.



2.2 Pertanyaan Praktikum 2

a. Kenapa json-server dijalankan pada port 3001? Kenapa tidak sama-sama dijalankan pada port 3000 seperti project react yang sudah kita buat?

→ Karena penggunaan json-server untuk melihat file json sesuai dengan perintah yg kita tuliskan, sedangkan penggunaan pada port 3001 agar tidak mengganggu port default pada react js.

b. Bagaimana jadinya kalau kita ganti port json-server menjadi 3000?

→ akan terjadi error karena terdapat 2 penggunaan pada port yang sama.

Praktikum 3

Interaksi dengan API menggunakan method DELETE

3.1 Langkah Praktikum 3

1. Buka stateless component Post. Tambahkan 1 baris kode program pada baris 10 seperti pada Gambar 3.1

```
import React from "react";

const Post = (props) => {
  return (
    <div className="artikel">
      <div className="gambar artikel">
        
        <div className="konten-artikel">
          <div
className="judul-artikel">{props.judul}</div>
          <p className="isi-artikel">{props.isi}</p>
          <button className="btn btn-sm btn-warning"
onClick={ () =>
props.hapusArtikel(props.idArtikel) }>Hapus</button>
        </div>
      </div>
    </div>
  )
}

export default Post;
```

2. Kemudian pada statefull component BlogPost, modifikasi kode program sebelumnya sesuai dengan Gambar 3.2

```
import React, {Component} from "react";
import './BlogPost.css'
import Post from "../../component/BlogPost/Post";

class BlogPost extends Component{
  state = { //Komponen state dari React untuk
statefull component
    listArtikel: [] //variabel array untuk menyimpan data
API
  }

  ambilDataDariServer = () => {
```

```

        fetch('http://localhost:3001/posts') //alamat API
        .then(response => response.json())
        .then(jsonHasilAmbilDariAPI => {
            this.setState({
                listArtikel: jsonHasilAmbilDariAPI
            })
        })
    }

    componentDidMount(){ //komponen yg dicek ketika telah diambil
di-mount-ing
        this.ambilDataDariServer() //ambil data API lokal
    }

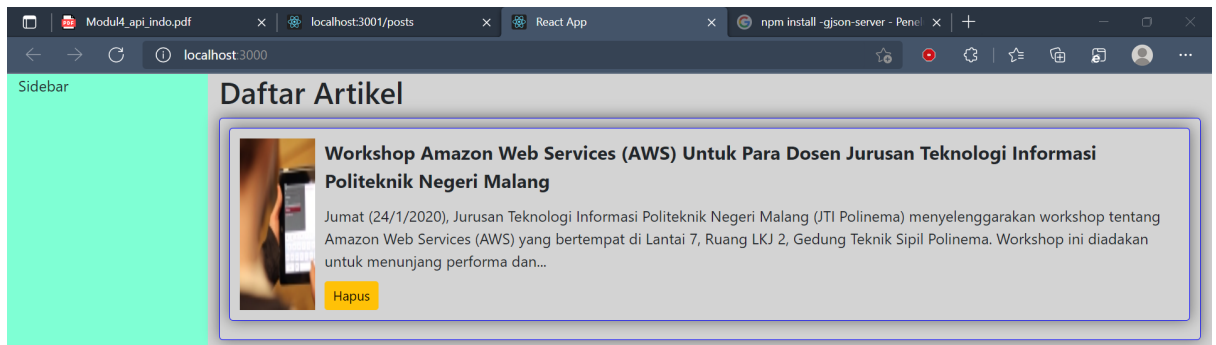
    handleHapusArtikel = (data) => {
        fetch(`http://localhost:3001/posts/${data}`, {method:
'DELETE'})
        .then(res => {
            this.ambilDataDariServer()
        })
    }

    render() {
        return(
            <div className="post-artikel">
                <h2>Daftar Artikel</h2>
                {
                    this.state.listArtikel.map(artikel => {
//lopping dan masukan data ke variabel artikel
                        return <Post key={artikel.id}
judul={artikel.title} isi={artikel.body} idArtikel={artikel.id}
hapusArtikel={this.handleHapusArtikel}/> //mapping data JSON
sesuai kategori
                    })
                }
                { /* <Post judul="JTI POLINEMA" isi="Jurusan
Teknologi Informasi - Politeknik Negeri Malang"/> */}
            </div>
        )
    }
}

export default BlogPost;

```

3. Klik tombol hapus pada list artikel di browser. Amati apa yang terjadi.



3.2 Pertanyaan Praktikum 3

- a. Apa yang terjadi setelah kalian klik tombol hapus?
→ terdapat artikel yang terhapus.
- b. Perhatikan file listArtikel.json, apa yang terjadi pada file tersebut? Kenapa demikian?
→ terdapat data yang terhapus.
- c. Fungsi handleHapusArtikel itu untuk apa?
→ dilakukan penghapusan listArtikel.json pada port 3001

Praktikum 4

Interaksi dengan API menggunakan method POST

4.1 Langkah Praktikum 4

1. Buka statefull component BlogPost, dan modifikasi pada fungsi render() untuk menampilkan form input artikel yang berisi judul dan isi berita

```
render() {
  return (
    <div className="post-artikel">
      <div className="form pb-2 border-bottom">
        <div className="form-group row">
          <label htmlFor="title"
className="col-sm-2 col-form-label">Judul</label>
          <div className="col-sm-10">
            <input type="text"
className="form-control" id="title" name="title"
onChange={this.handleTambahArtikel}
            />
          </div>
        </div>
        <div className="form-group row">
          <label htmlFor="body" className="col-sm-2
col-form-label">Isi</label>
          <div className="col-sm-10">
            <textarea
              name="body" id="body" rows="3"
className="form-control" onChange={this.handleTambahArtikel}
            ></textarea>
          </div>
        </div>
        <button type="submit" className="btn
btn-primary" onClick={this.handleTombolSimpan}>Simpan</button>
      </div>
      <h2>Daftar Artikel</h2>
      {this.state.listArtikel.map((artikel) => {
        // looping dan masukkan untuk setiap data
yang ada di listArtikel ke variabel artikel
        return (
          <Post
            key={artikel.id}
            judul={artikel.title}
            isi={artikel.body}
            idArtikel={artikel.id}
```

```

hapusArtikel={this.handleHapusArtikel}
        />
    ); // mappingkan data json dari API sesuai
    dengan kategorinya
    )))
  </div>
);
}

```

2. Kemudian modifikasi BlogPost untuk bagian state dan request API dari server, seperti Gambar 4.2

```

    state = { //Komponen state dari React untuk
statefull component
      listArtikel: [], //variabel array untuk menyimpan
data API
      insertArtikel: {
        userId: 1,
        id: 1,
        title: "",
        body: ""
      }
    }
  }

```

3. Tambahkan untuk handle form tambah data artikel

```

    handleTambahArtikel = (event) => { // fungsi untuk
meng-handle form tambah data artikel
      let formInsertArtikel = { ...this.state.insertArtikel };
// cloning data state insertArtikel ke dalam variabel
formInsertArtikel
      let timestamp = new Date().getTime(); // digunakan untuk
menyimpan waktu (sebagai id artikel)
      formInsertArtikel["id"] = timestamp;
      formInsertArtikel[event.target.name] =
event.target.value; // menyimpan data onChange ke
formInsertArtikel sesuai dengan target yang diisi
      this.setState({
        insertArtikel: formInsertArtikel,
      });
    };
  };

```

4. Langkah terakhir tambahkan fungsi untuk handle tombol simpan artikel,

```

    handleTombolSimpan = () => { // fungsi untuk meng-handle
tombol simpan

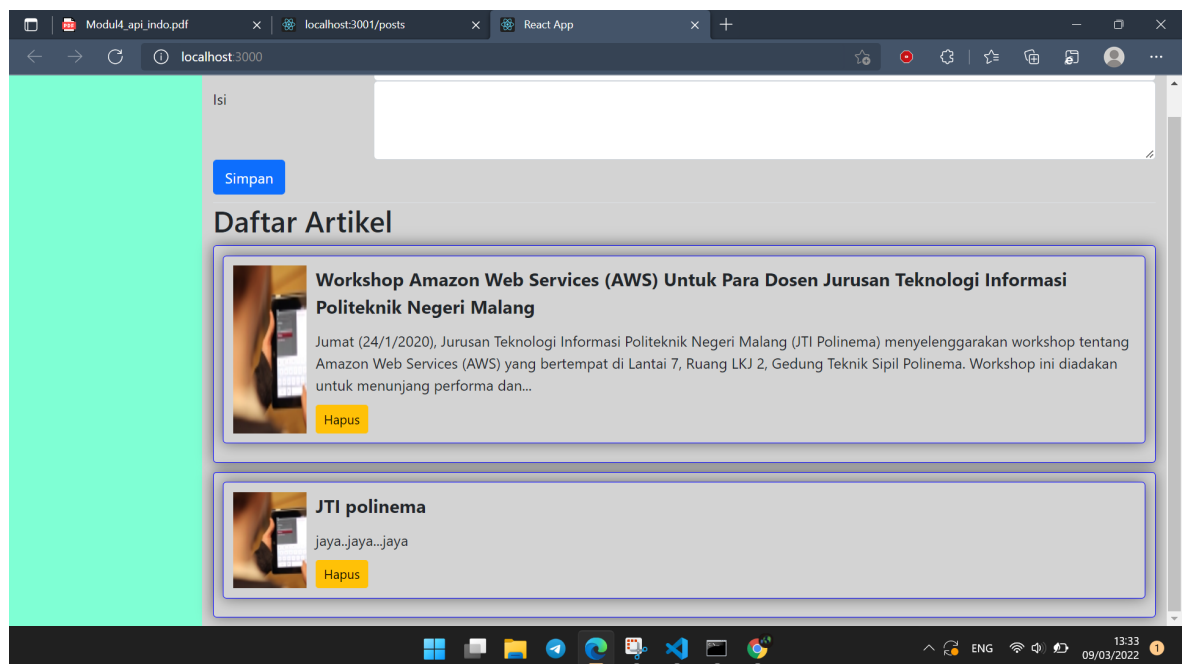
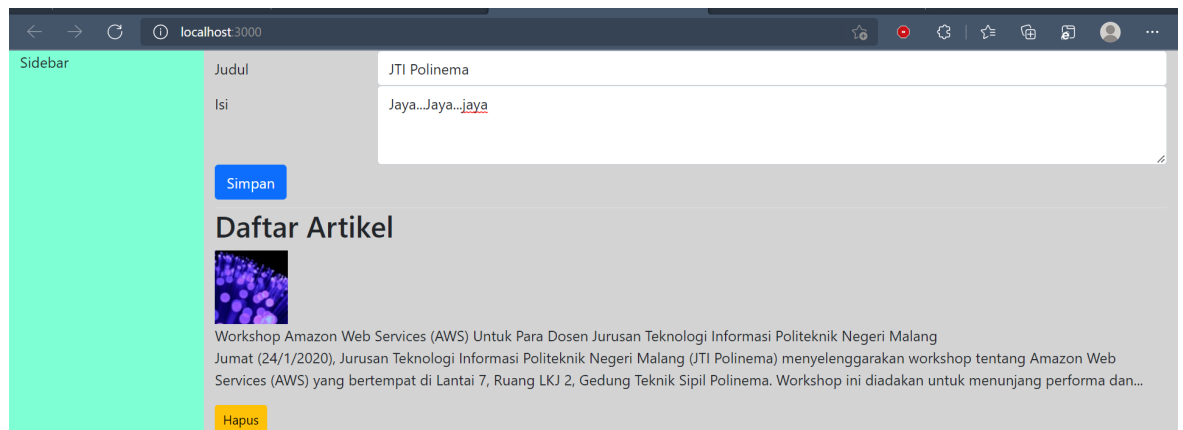
```

```

fetch("http://localhost:3001/posts", {
  method: "POST", // method POST untuk input atau insert
data
  headers: {
    Accept: "application/json",
    "Content-Type": "application/json",
  },
  body: JSON.stringify(this.state.insertArtikel), //
  kirimkan ke body request untuk data artikel yang akan ditambahkan
(insert)
}).then((Response) => {
  this.ambilDataDariSeverAPI(); // reload / refresh data
});
};

```

5. Simpan, lakukan percobaan penambahan data, dan amati perubahannya.



4.2 Pertanyaan Praktikum 4

a. Jelaskan apa yang terjadi pada file listArtikel.json sebelum dan setelah melakukan penambahan data?

→ setelah dilakukan penambahan data maka otomatis file .json juga akan bertambah sesuai data yg kita inputkan.

b. Data yang ditampilkan di browser adalah data terbaru berada di posisi atas dan data lama berada di bawah, sedangkan pada file listArtikel.json data terbaru malah berada di bawah. Jelaskan mengapa demikian?

→ karna untuk pengambilan data di browser dilakukan pengambilan data dari waktu yang terbaru. sedangkan pada file.json data yang diinputkan akan berada dibawah data sebelumnya.