

TUGAS PERORANGAN/INDIVIDU

LAPORAN MINGGU-10

Disusun sebagai

MATA KULIAH : Pemrograman Berbasis Framework

Oleh :

Mery Kristiani 1741720104

3B/17



PROGRAM STUDI D-IV TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

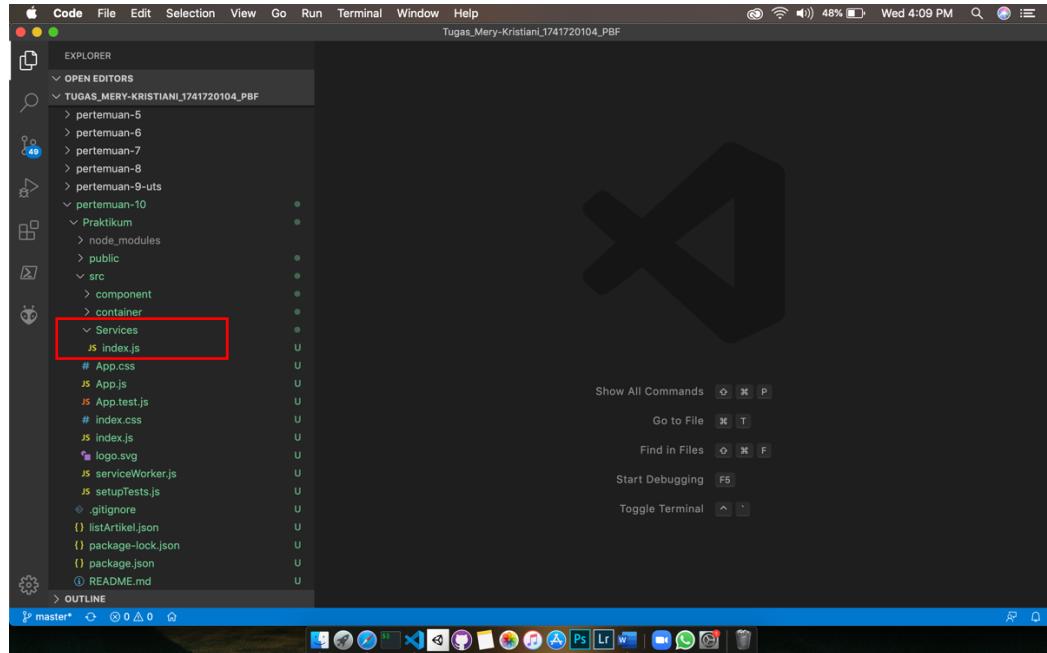
2020

PRAKTIKUM 1

Global API service GET

Langkah Praktikum 1

- Buat folder baru bernama **Services** dalam folder **src**, kemudian buat file **index.js** seperti gambar dibawah ini.



- Buat file **BlogPost.jsx** pada folder container (*statefull component*) dan akan tampil kode seperti gambar dibawah ini.

```
JS index.js  BlogPost.jsx < pertemuan-10 > src > container > BlogPost > BlogPost.jsx > BlogPost
1 import React, {Component} from "react";
2 import './BlogPost.css';
3 import Post from "../../component/BlogSpot/Post";
4
5 class BlogPost extends Component {
6   state = {
7     listArtikel: [], // komponen state dari React untuk statefull component
8     insertArtikel: { // variabel array yang digunakan untuk menyimpan sementara data yang akan di insert
9       userID: 1, // kolom userId, id, title dan body sama, mengikuti kolom yang ada pada listArtikel.json
10      id: 1,
11      title: "",
12      body: ""
13    }
14  };
15
16  ambilDataDariServerAPI = () => { // fungsi untuk mengambil data dari API dengan penambahan sort dan order
17    fetch('http://localhost:3001/posts?sort=id&order=desc') // penambahan sort dan order berdasarkan parameter
18      .then(response => response.json()) // ubah response data dari URL API menjadi sebuah data json
19      .then(jsonHasilAmbilDariAPI => {
20        this.setState({
21          listArtikel: jsonHasilAmbilDariAPI
22        })
23      })
24
25
26  componentDidMount() { // komponen untuk mengecek ketika component telah di mount ing, maka panggil API
27    this.ambilDataDariServerAPI() // ambil data dari server API lokal
28  }
29}

Note that the development build is not optimized.
To create a production build, use npm run build.
```

The code is a React component named 'BlogPost'. It imports React and a local CSS file. It uses a state variable 'listArtikel' to store data from a local API. The component has a constructor that calls the 'ambilDataDariServerAPI' function. This function uses the 'fetch' method to get data from 'http://localhost:3001/posts?sort=id&order=desc'. The response is converted to JSON, and the resulting data is set as the state's 'listArtikel'. The component also includes a 'componentDidMount' lifecycle method that calls 'this.ambilDataDariServerAPI' to fetch data when it mounts.

3. Pada fungsi **ambilDataDariServerApi** (baris 16) terdapat pemanggilan API GET untuk merequest data artikel. Proses dari fungsi inilah yang akan kita manage kedalam satu tempat yaitu **index.js**.
4. Buka file **index.js** pada folder **service**, yang telah kita buat tadi dan tulis kode seperti gambar dibawah ini.

```

Code File Edit Selection View Go Run Terminal Window Help
index.js — Tugas_Mery-Kristiani_1741720104_PBF
pertemuan-10 > src > Services > JS index.js > ...
1 const domainPath = 'http://localhost:3001'; // simpulkan domain server API pada variabel, sehingga bisa dinamis (diganti)
2 const GetAPI = (path) => {
3   const promise = new Promise((resolve, reject) => {
4     fetch(`${domainPath}/${path}`)
5       .then(response => response.json())
6       .then(result) => {
7         resolve(result);
8       }, (err) => {
9         reject(err);
10      })
11    return promise;
12  }
13
14
15 const getNewsBlog = () => GetAPI('posts?_sort=id&_order=desc');
16
17 const API = {
18   getNewsBlog
19 }
20
21 export default API;
22

```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

Note that the development build is not optimized.
To create a production build, use `npm run build`.

Ln 22, Col 1 Spaces: 4 UTF-8 LF JavaScript Prettier

5. Kembali ke file **BlogPost.jsx**. ganti baris 17 sampai 23 menjadi seperti gambar dibawah ini.

```

Code File Edit Selection View Go Run Terminal Window Help
BlogPost.jsx — Tugas_Mery-Kristiani_1741720104_PBF
pertemuan-10 > src > container > BlogPost > BlogPost.jsx > handleHapusArtikel
1 import React, {Component} from "react";
2 import './BlogPost.css';
3 import Post from "../../component/BlogSpot/Post";
4
5 class BlogPost extends Component {
6   state = {
7     listArtikel: [], // komponen state dari React untuk statefull component
8     insertArtikel: { // variabel array yang digunakan untuk menyimpan data API
9       userID: 1, // kolom userid, id, title dan body sama, mengikuti kolom yang ada pada listArtikel.json
10      id: 1,
11      title: "",
12      body: ""
13    };
14  };
15
16  ambilDataDariServerAPI = () => { // fungsi untuk mengambil data dari API dengan penambahan sort dan order
17    API.getNewsBlog().then(result => {
18      this.setState({
19        listArtikel: result
20      })
21    })
22  }
23
24  componentDidMount() { // komponen untuk mengecek ketika component telah di mount ing, maka panggil API
25    this.ambilDataDariServerAPI() // ambil data dari server API lokal
26  }
27
28  handleHapusArtikel = (data) => {
29    fetch(`http://localhost:3001/posts/${data}`, {method: 'DELETE'})

```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

Note that the development build is not optimized.
To create a production build, use `npm run build`.

Ln 33, Col 6 Spaces: 2 UTF-8 LF JavaScript React Prettier

Pertanyaan Praktikum 1

- a. Perhatikan file **index.js**, apa tujuan dibuatnya fungsi **GetAPI** pada baris 2 dan fungsi **getNewsBlog** pada baris 16?

Jawaban :

- Fungsi **GetAPI**

API itu sendiri berfungsi untuk mempercepat proses *development* dengan menyediakan *function* secara terpisah sehingga *developer* tidak perlu membuat fitur yang serupa. Dan **GetAPI** disini berfungsi untuk mengambil data pada json.

- Fungsi **getNewsBlog**

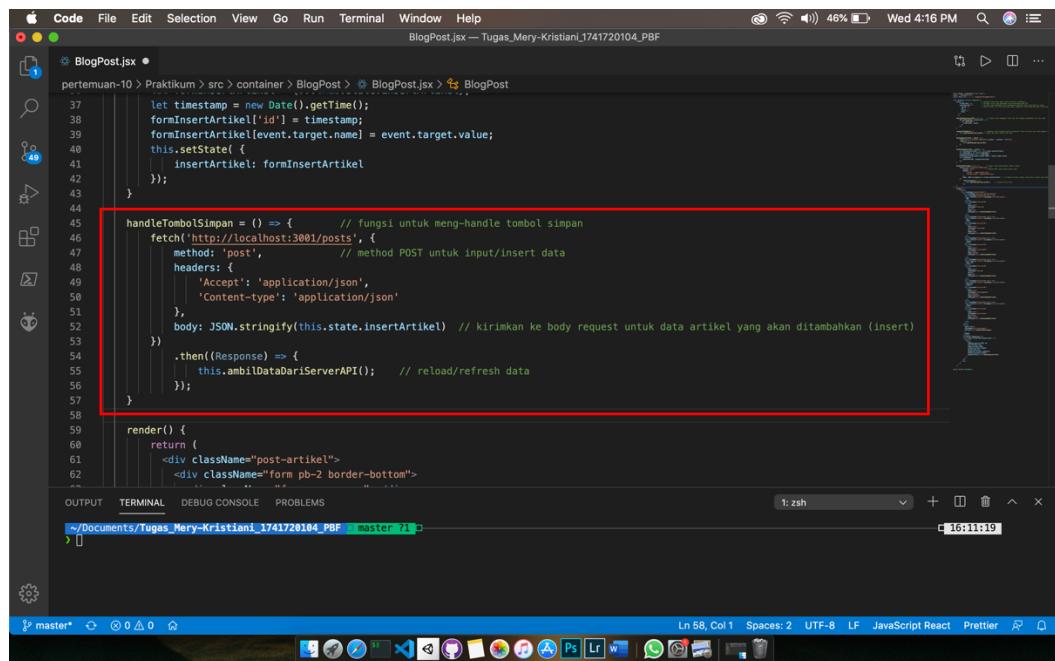
Berfungsi untuk mendapatkan data dari json dan mengurutkan secara descending yaitu dari yang terbesar sampai terkecil berdasarkan id.

PRAKTIKUM 2

Global API service POST

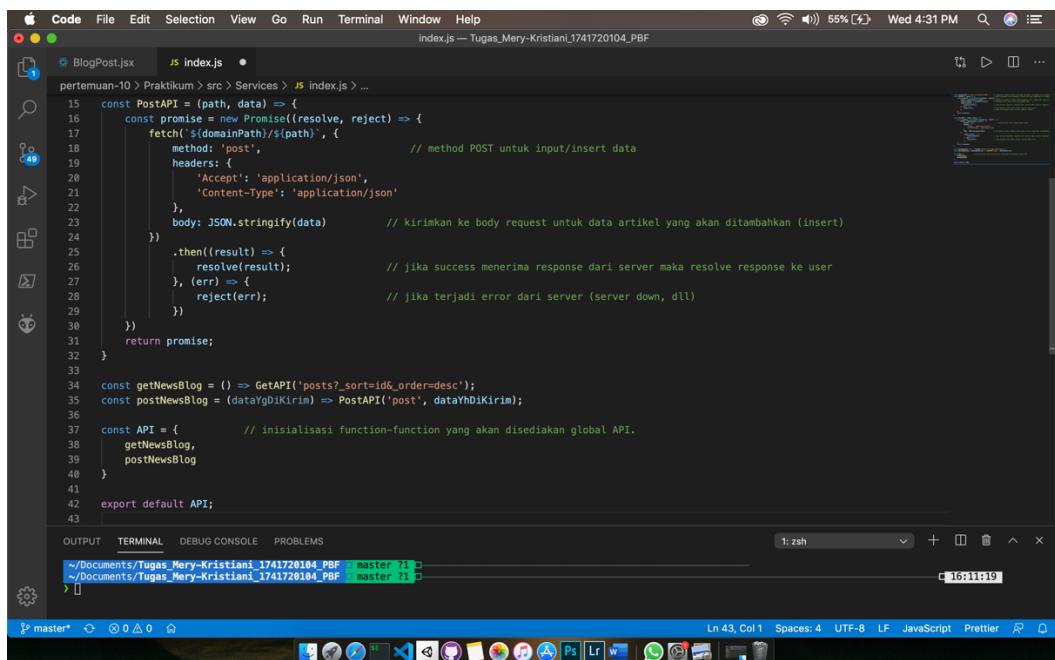
Langkah Praktikum 2

- Perhatikan pada fungsi **handleTombolSimpan** pada file **BlogPost.js**. Bagian inilah yang selanjutnya dikelola agar bisa di manage.



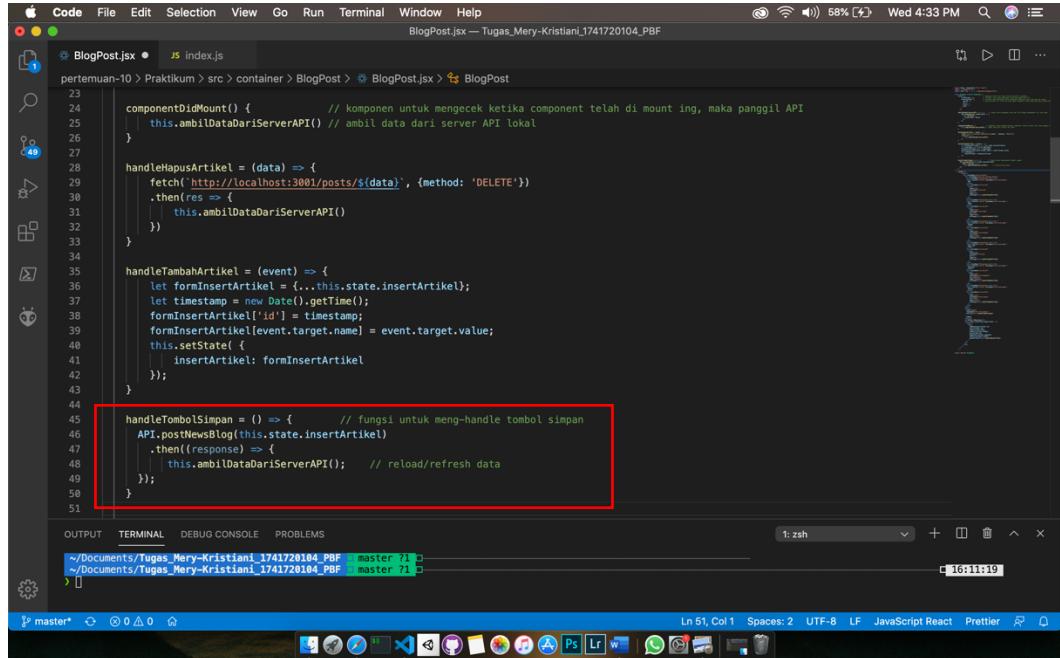
```
BlogPost.jsx • pertemuan-10 > Praktikum > src > container > BlogPost > BlogPost.jsx > BlogPost
37     let timestamp = new Date().getTime();
38     formInsertArtikel['id'] = timestamp;
39     formInsertArtikel[event.target.name] = event.target.value;
40     this.setState({
41       insertArtikel: formInsertArtikel
42     });
43   }
44
45   handleTombolSimpan = () => { // fungsi untuk meng-handle tombol simpan
46     fetch('http://localhost:3001/posts', {
47       method: 'post', // method POST untuk input/insert data
48       headers: {
49         'Accept': 'application/json',
50         'Content-type': 'application/json'
51       },
52       body: JSON.stringify(this.state.insertArtikel) // kirimkan ke body request untuk data artikel yang akan ditambahkan (insert)
53     })
54     .then(Response) => {
55       this.ambilDataDariServerAPI(); // reload/refresh data
56     });
57   }
58
59   render() {
60     return (
61       <div className="post-artikel">
62         <div className="form pb-2 border-bottom">
```

- Buatlah fungsi untuk menampung action POST dari file **BlogPost.js** pada file **services/index.js** dan inisialisasi fungsi tersebut seperti gambar dibawah ini.



```
index.js • pertemuan-10 > Praktikum > src > Services > index.js ...
15 const PostAPI = (path, data) => {
16   const promise = new Promise((resolve, reject) => {
17     fetch(`${domainPath}/${path}`, {
18       method: 'post', // method POST untuk input/insert data
19       headers: {
20         'Accept': 'application/json',
21         'Content-Type': 'application/json'
22       },
23       body: JSON.stringify(data) // kirimkan ke body request untuk data artikel yang akan ditambahkan (insert)
24     })
25     .then(result) => {
26       resolve(result);
27     }, (err) => {
28       reject(err);
29     }
30   })
31   return promise;
32 }
33
34 const getNewsBlog = () => GetAPI('posts?_sort=id&_order=desc');
35 const postNewsBlog = (dataYgDiKirim) => PostAPI('post', dataYgDiKirim);
36
37 const API = { // inisialisasi function-function yang akan disediakan global API.
38   getNewsBlog,
39   postNewsBlog
40 }
41
42 export default API;
43
```

3. Selanjutnya pindah ke file **Blogpost.js** dan ganti isi dari fungsi **handleTombolSimpan** menjadi seperti gambar dibawah ini.



```

 23   componentDidMount() { // komponen untuk mengecek ketika component telah di mount ing, maka panggil API
 24     | this.ambilDataDariServerAPI() // ambil data dari server API lokal
 25   }
 26
 27
 28   handleHapusArtikel = (data) => {
 29     fetch(`http://localhost:3001/posts/${data}`, {method: 'DELETE'})
 30     .then(res => {
 31       | this.ambilDataDariServerAPI()
 32     })
 33   }
 34
 35   handleTambahArtikel = (event) => {
 36     let formInsertArtikel = {...this.state.insertArtikel};
 37     let timestamp = new Date().getTime();
 38     formInsertArtikel['id'] = timestamp;
 39     formInsertArtikel[event.target.name] = event.target.value;
 40     this.setState({
 41       insertArtikel: formInsertArtikel
 42     });
 43   }
 44
 45   handleTombolSimpan = () => { // fungsi untuk meng-handle tombol simpan
 46     API.postNewsBlog(this.state.insertArtikel)
 47     .then((response) => {
 48       | this.ambilDataDariServerAPI(); // reload/refresh data
 49     });
 50   }
 51

```

Pertanyaan Praktikum 2

- a. Perhatikan file **index.js**, apa tujuan dibuatnya fungsi **PostAPI** dan fungsi **postNewBlog**?

Jawaban :

- Fungsi **PostAPI**

Berfungsi untuk meng-inputkan atau meng-insertkan data.

- Fungsi **postNewBlog**

Berfungsi untuk mengirim data yang selanjutnya dilanjutkan ke fungsi **PostAPI**.

- b. Pada fungsi **postNewBlog**, terdapat variable **dataYangDikirim**. Apa tujuan dari dibuatnya variable tersebut?

Jawaban :

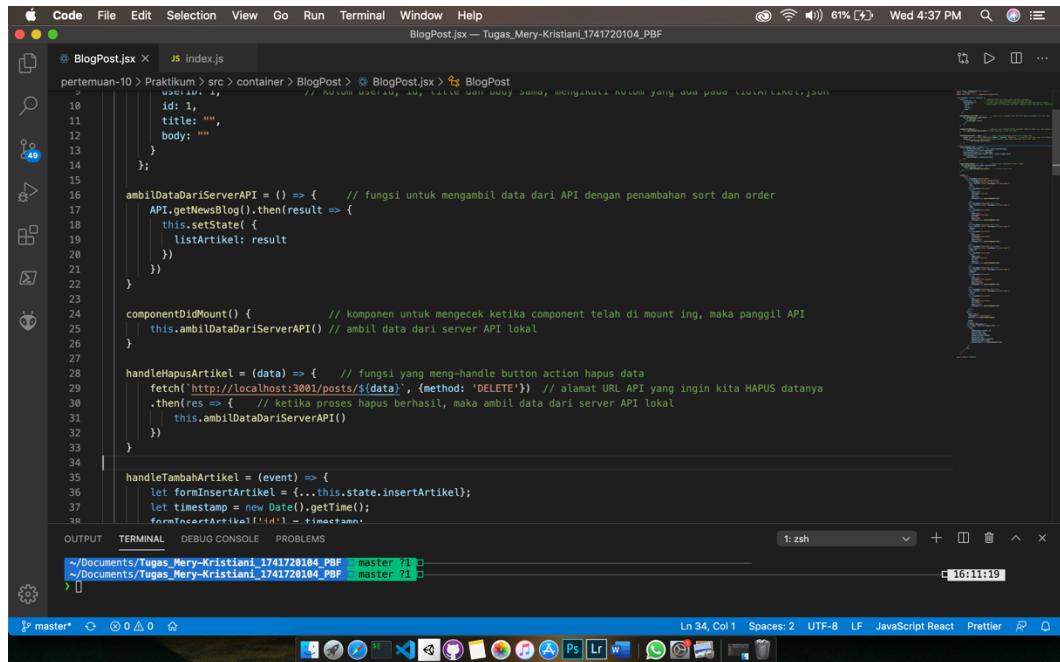
Tujuan dibuat variable **dataYangDikirim** yaitu menerima data sementara yang dikirim sebelum dilanjutkan ke fungsi **PostAPI**.

PRAKTIKUM 3

Global API service DELETE

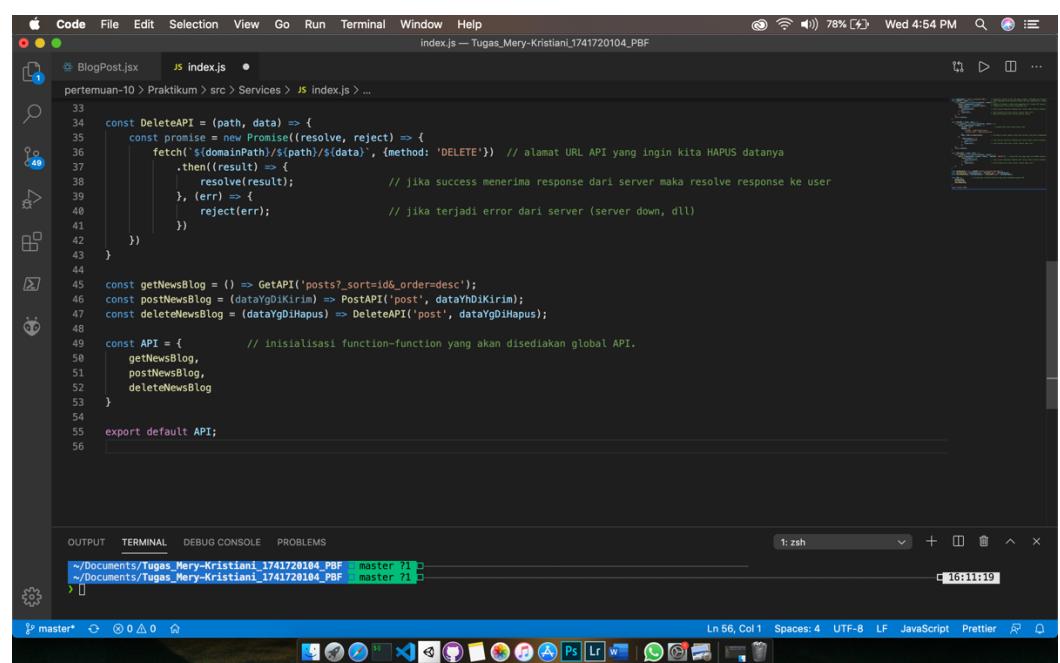
Langkah Praktikum 3

- Perhatikan pada fungsi **handleHapusArtikel** pada file **BlogPost.jsx**. Bagian inilah yang selanjutnya dikelola agar bisa di manage.



```
BlogPost.jsx x js index.js
pertemuan-10 > Praktikum > src > container > BlogPost > BlogPost.jsx - Tugas_Mery-Kristiani_1741720104_PBF
10  user_id, ...
11  id: 1,
12  title: '',
13  body: ''
14  };
15
16  ambilDataDariServerAPI = () => { // fungsi untuk mengambil data dari API dengan penambahan sort dan order
17    API.getNewsBlog().then(result => {
18      this.setState({
19        listArtikel: result
20      })
21    })
22  }
23
24  componentDidMount() { // komponen untuk mengecek ketika component telah di mount ing, maka panggil API
25    this.ambilDataDariServerAPI() // ambil data dari server API lokal
26  }
27
28  handleHapusArtikel = (data) => { // fungsi yang meng-handle button action hapus data
29    fetch(`http://localhost:3001/posts/${data}`, {method: 'DELETE'}) // alamat URL API yang ingin kita HAPUS datanya
30    .then(res => { // ketika proses hapus berhasil, maka ambil data dari server API lokal
31      this.ambilDataDariServerAPI()
32    })
33  }
34
35  handleTambahArtikel = (event) => {
36    let formInsertArtikel = {...this.state.insertArtikel};
37    let timestamp = new Date().getTime();
38    formInsertArtikel['id'] = timestamp;
39
40    const API = {
41      getNewsBlog,
42      postNewsBlog,
43      deleteNewsBlog
44    }
45
46    const getNewsBlog = () => GetAPI('posts?_sort=id&_order=desc');
47    const postNewsBlog = (dataYgDiKirim) => PostAPI('post', dataYgDiKirim);
48    const deleteNewsBlog = (dataYgDiHapus) => DeleteAPI('post', dataYgDiHapus);
49
50    const API = { // inisialisasi function-function yang akan disediakan global API.
51      getNewsBlog,
52      postNewsBlog,
53      deleteNewsBlog
54    }
55
56    export default API;
```

- Buatlah fungsi untuk menampung action DELETE dari file **BlogPost.jsx** pada file **services/index.js** dan inisialisasi fungsi tersebut seperti gambar dibawah ini.



```
BlogPost.jsx x js index.js
index.js - Tugas_Mery-Kristiani_1741720104_PBF
pertemuan-10 > Praktikum > src > Services > index.js > ...
33
34  const DeleteAPI = (path, data) => {
35    const promise = new Promise((resolve, reject) => {
36      fetch(`${domainPath}/${path}/${data}`, {method: 'DELETE'}) // alamat URL API yang ingin kita HAPUS datanya
37      .then((result) => {
38        resolve(result); // jika success menerima response dari server maka resolve response ke user
39      }, (err) => {
40        reject(err); // jika terjadi error dari server (server down, dll)
41      })
42    })
43  }
44
45  const getNewsBlog = () => GetAPI('posts?_sort=id&_order=desc');
46  const postNewsBlog = (dataYgDiKirim) => PostAPI('post', dataYgDiKirim);
47  const deleteNewsBlog = (dataYgDiHapus) => DeleteAPI('post', dataYgDiHapus);
48
49  const API = { // inisialisasi function-function yang akan disediakan global API.
50    getNewsBlog,
51    postNewsBlog,
52    deleteNewsBlog
53  }
54
55  export default API;
```

3. Selanjutnya pindah file **BlogPost.jsx** dan ganti isi dari fungsi **handleHapusArtikel** menjadi seperti gambar dibawah ini.

```

 24         // komponen untuk mengecek ketika component telah di mount ing, maka panggil API
 25         this.ambilDataDariServerAPI() // ambil data dari server API lokal
 26     }
 27
 28     handleHapusArtikel = (data) => { // fungsi yang meng-handle button action hapus data
 29         fetch(`http://localhost:3001/posts/${data}`, {method: 'DELETE'}) // alamat URL API yang ingin kita HAPUS datanya
 30         .then(res => { // ketika proses hapus berhasil, maka ambil data dari server API lokal
 31             this.ambilDataDariServerAPI()
 32         })
 33     }
 34
 35     handleTambahArtikel = (event) => {
 36         let formInsertArtikel = {...this.state.insertArtikel};
 37         let timestamp = new Date().getTime();
 38         formInsertArtikel['id'] = timestamp;
 39         formInsertArtikel[event.target.name] = event.target.value;
 40         this.setState({
 41             insertArtikel: formInsertArtikel
 42         });
 43     }
 44
 45     handleTombolSimpan = () => { // fungsi untuk meng-handle tombol simpan
 46         API.postNewsBlog(this.state.insertArtikel)
 47         .then((response) => {
 48             this.ambilDataDariServerAPI(); // reload/refresh data
 49         });
 50     };
 51 }

```

Pertanyaan Praktikum 3

- a. Perhatikan file **index.js**, apa tujuan dibuatnya fungsi **DeleteAPI** dan fungsi **deleteNewsBlog**?

Jawaban :

- Fungsi **DeleteAPI**

Berfungsi untuk menghapus data yang ada pada file json.

- Fungsi **deleteNewsBlog**

Berfungsi untuk menghapus data yang selanjutnya dilanjutkan ke fungsi **PostAPI**.

- b. Pada fungsi **deleteNewsBlog**, terdapat variable **dataYangDihapus**. Apa tujuan dari dibuatnya variable tersebut?

Jawaban :

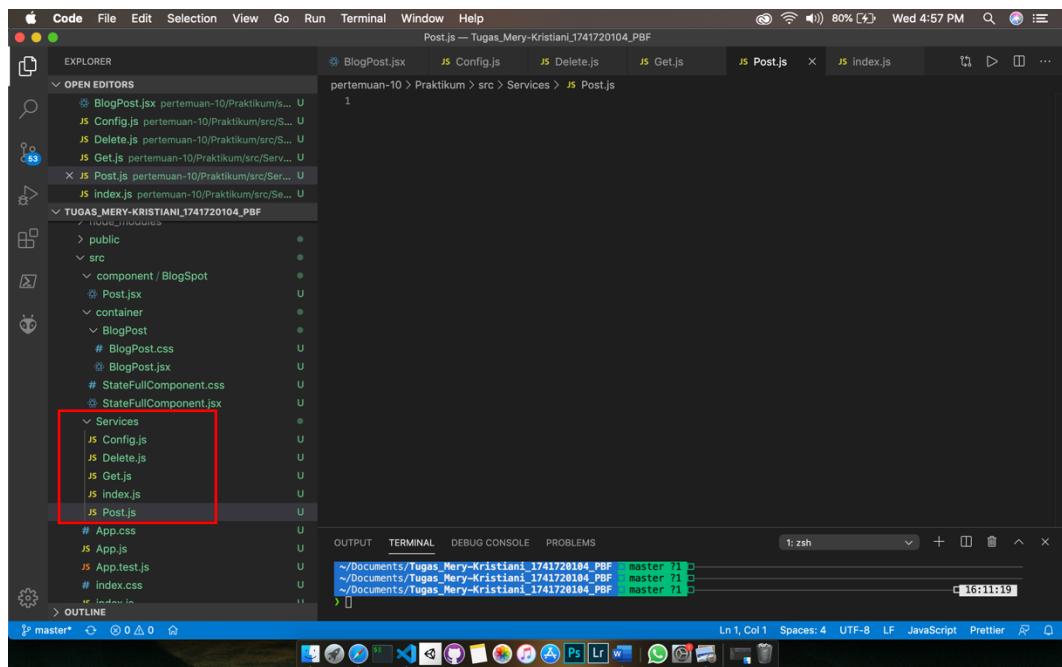
Tujuan dibuat variabel **dataYangDihapus** yaitu tempat menghapus data sebelum dilanjutkan ke fungsi **PostAPI**.

PRAKTIKUM 4

Manage Global API service

Langkah Praktikum 4

- Buatlah file **Config.js**, **Get.js**, **Post.js**, dan **Delete.js** dalam folder **services** seperti gambar dibawah ini.



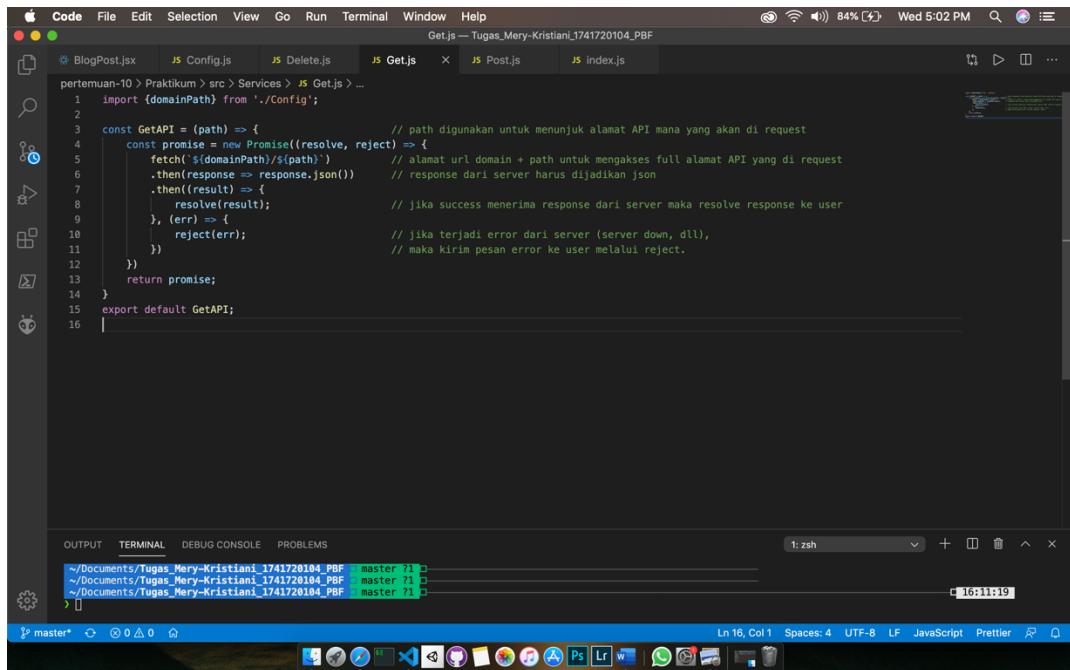
- Buka **Config.js** dan isikan kode seperti gambar dibawah ini.

```
// file config ini bisa berisi variabel-variabel lain yang dibutuhkan untuk proses API
export const doaminPath = 'http://localhost:3001'; // simpan url domain server API pada variabel, sehingga bisa dinamis (dиганті)
```

The screenshot shows the VS Code interface with the following details:

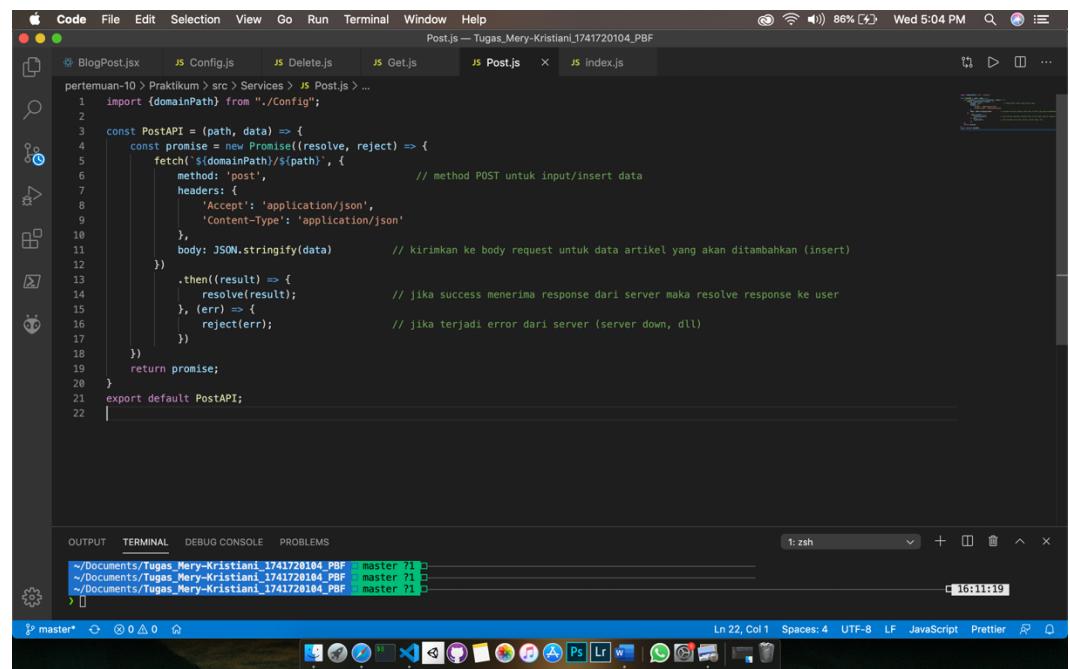
- File Explorer:** Shows the project structure under 'TUGAS_MERY-KRISTIANI_1741720104_PBF'. A red box highlights the 'Config.js' file.
- Editors:** The 'Config.js' file is open in the editor.
- Terminal:** The terminal shows three entries: ~/Documents/Tugas_Mery-Kristiani_1741720104_PBF master ?1, ~/Documents/Tugas_Mery-Kristiani_1741720104_PBF master ?1, and ~/Documents/Tugas_Mery-Kristiani_1741720104_PBF master ?1.
- Status Bar:** Shows the file path as 'pertemuan-10 > Praktikum > src > Services > JS Config.js', the line number '1: zsh', and the time '16:11:19'.

3. Buka **Get.js** dan isikan kode seperti gambar dibawah ini.



```
Code File Edit Selection View Go Run Terminal Window Help
Get.js — Tugas_Mery-Kristiani_1741720104_PBF
BlogPost.jsx JS Config.js JS Delete.js JS Get.js JS Post.js JS index.js
pertemuan-10 > Praktikum > src > Services > JS Get.js > ...
1 import {domainPath} from './Config';
2
3 const GetAPI = (path) => {
4   const promise = new Promise((resolve, reject) => {
5     fetch(`${domainPath}/${path}`)
6       .then(response => response.json())
7       .then(result => {
8         resolve(result);
9       }, (err) => {
10      reject(err);
11    })
12  })
13  return promise;
14}
15 export default GetAPI;
16
```

4. Buka **Post.js** dan isikan kode seperti gambar dibawah ini.



```
Code File Edit Selection View Go Run Terminal Window Help
Post.js — Tugas_Mery-Kristiani_1741720104_PBF
BlogPost.jsx JS Config.js JS Delete.js JS Get.js JS Post.js JS index.js
pertemuan-10 > Praktikum > src > Services > JS Post.js > ...
1 import {domainPath} from "./Config";
2
3 const PostAPI = (path, data) => {
4   const promise = new Promise((resolve, reject) => {
5     fetch(`${domainPath}/${path}`, {
6       method: 'post', // method POST untuk input/insert data
7       headers: {
8         'Accept': 'application/json',
9         'Content-Type': 'application/json'
10      },
11      body: JSON.stringify(data) // kirimkan ke body request untuk data artikel yang akan ditambahkan (insert)
12    })
13      .then(result => {
14        resolve(result);
15      }, (err) => {
16        reject(err);
17      })
18  })
19  return promise;
20}
21 export default PostAPI;
22
```

5. Buka **Delete.js** dan isikan kode seperti gambar dibawah ini.

```
Code File Edit Selection View Go Run Terminal Window Help
Delete.js — Tugas_Mery-Kristiani_1741720104_PBF
pertemuan-10 > Praktikum > src > Services > Delete.js > ...
1 import {domainPath} from "./Config";
2
3 const DeleteAPI = (path, data) => {
4     const promise = new Promise((resolve, reject) => {
5         fetch(`${domainPath}/${path}/data`, {method: 'DELETE'}) // alamat URL API yang ingin kita HAPUS datanya
6             .then(result) => {
7                 resolve(result); // jika success menerima response dari server maka resolve response ke user
8             }, (err) => {
9                 reject(err); // jika terjadi error dari server (server down, dll)
10            })
11    })
12 }
13 export default DeleteAPI;
14
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

```
~/Documents/Tugas_Mery-Kristiani_1741720104_PBF master 71
~/Documents/Tugas_Mery-Kristiani_1741720104_PBF master 71
~/Documents/Tugas_Mery-Kristiani_1741720104_PBF master 71
```

Ln 14, Col 1 Spaces: 4 UTF-8 LF JavaScript Prettier

6. Pada **index.js** edit menjadi seperti gambar dibawah ini.

```
Code File Edit Selection View Go Run Terminal Window Help
index.js — Tugas_Mery-Kristiani_1741720104_PBF
pertemuan-10 > Praktikum > src > Services > index.js > ...
1 import GetAPI from "./Get";
2 import PostAPI from "./Post";
3 import DeleteAPI from "./Delete";
4
5 // Daftar API - GET
6 const getNewsBlog = () => GetAPI('posts?_sort=id&_order=desc');
7
8 // Daftar API - POST
9 const postNewsBlog = (dataYgDiKirim) => PostAPI('post', dataYgDiKirim);
10
11 // Daftar API - DELETE
12 const deleteNewsBlog = (dataYgDihapus) => DeleteAPI('post', dataYgDihapus);
13
14 const API = { // inisialisasi function-function yang akan disediakan global API.
15     getNewsBlog,
16     postNewsBlog,
17     deleteNewsBlog
18 }
19
20 export default API;
21
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

```
~/Documents/Tugas_Mery-Kristiani_1741720104_PBF master 71
~/Documents/Tugas_Mery-Kristiani_1741720104_PBF master 71
~/Documents/Tugas_Mery-Kristiani_1741720104_PBF master 71
```

Ln 21, Col 1 Spaces: 4 UTF-8 LF JavaScript Prettier

Pertanyaan Praktikum 4

- Bagaimana caranya untuk menambahkan daftar API baru baik untuk method GET, POST, DELETE pada Global API?

Jawaban :

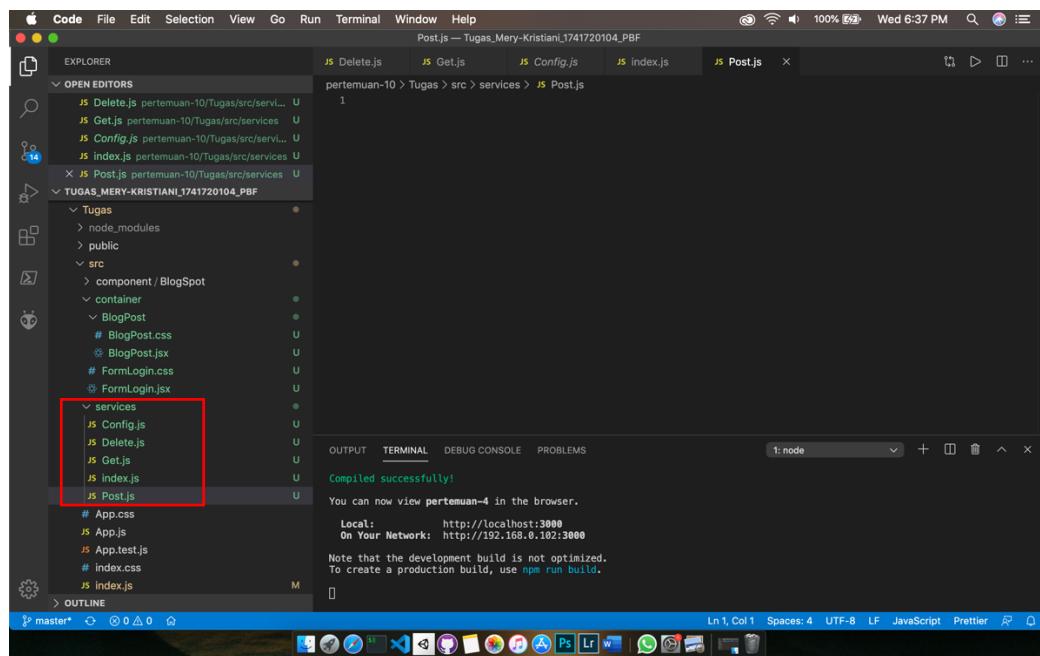
Menambahkan di API baru (method GET, POST, DELETE) di folder **services** dan import masing-masing file tersebut ke dalam file **index.js**

TUGAS PRAKTIKUM

- Ubahlah **Tugas pada Modul 4** yang sudah kalian buat dengan memanfaatkan Global API seperti praktikum yang kita lakukan pada Modul 8 ini.

Jawaban :

- Buatlah file **Config.js, Get.js, Post.js, index.js** dan **Delete.js** dalam folder **services** seperti gambar dibawah ini.



- Buka **Config.js** dan isikan kode seperti gambar dibawah ini.

```
1 // file config ini bisa berisi variabel-variabel lain yang dibutuhkan untuk proses API
2 export const doaminPath = 'http://localhost:3001'; // simpan url domain server API pada variabel, sehingga bisa dinamis (diganti)
```

The screenshot shows a code editor with the 'Config.js' file open. The code contains two lines of JavaScript: '1 // file config ini bisa berisi variabel-variabel lain yang dibutuhkan untuk proses API' and '2 export const doaminPath = 'http://localhost:3001'; // simpan url domain server API pada variabel, sehingga bisa dinamis (diganti)'. The status bar at the bottom indicates 'Ln 2, Col 131' and various file formats like 'JavaScript' and 'Prettier'.

c. Buka **Get.js** dan isikan kode seperti gambar dibawah ini.

```
Code File Edit Selection View Go Run Terminal Window Help
Get.js — Tugas_Mery-Kristiani_1741720104_PBF
pertemuan-10 > Tugas > src > services > JS Get.js ...
1 import {domainPath} from './Config';
2
3 const GetAPI = (path) => {
4     const promise = new Promise((resolve, reject) => {
5         fetch(`${domainPath}/${path}`)
6             .then(response => response.json())
7             .then(result => {
8                 resolve(result);
9             }, (err) => {
10                 reject(err);
11             })
12     })
13     return promise;
14 }
15 export default GetAPI;
16 |
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

Compiled successfully!

You can now view pertemuan-4 in the browser.

Local: http://localhost:3000
On Your Network: http://192.168.0.102:3000

Note that the development build is not optimized.
To create a production build, use `npm run build`.

Ln 16, Col 1 Spaces: 4 UTF-8 LF JavaScript Prettier ⌂ ⌂

d. Buka **Post.js** dan isikan kode seperti gambar dibawah ini.

```
Code File Edit Selection View Go Run Terminal Window Help
Post.js — Tugas_Mery-Kristiani_1741720104_PBF
pertemuan-10 > Tugas > src > services > JS Post.js ...
1 import {domainPath} from './Config';
2
3 const PostAPI = (path, data) => {
4     const promise = new Promise((resolve, reject) => {
5         fetch(`${domainPath}/${path}`, {
6             method: 'post', // method POST untuk input/insert data
7             headers: {
8                 'Accept': 'application/json',
9                 'Content-Type': 'application/json'
10            },
11            body: JSON.stringify(data) // kirimkan ke body request untuk data artikel yang akan ditambahkan (insert)
12        })
13        .then(result => {
14            resolve(result);
15        }, (err) => {
16            reject(err);
17        })
18    })
19    return promise;
20 }
21 export default PostAPI;
22 |
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

Compiled successfully!

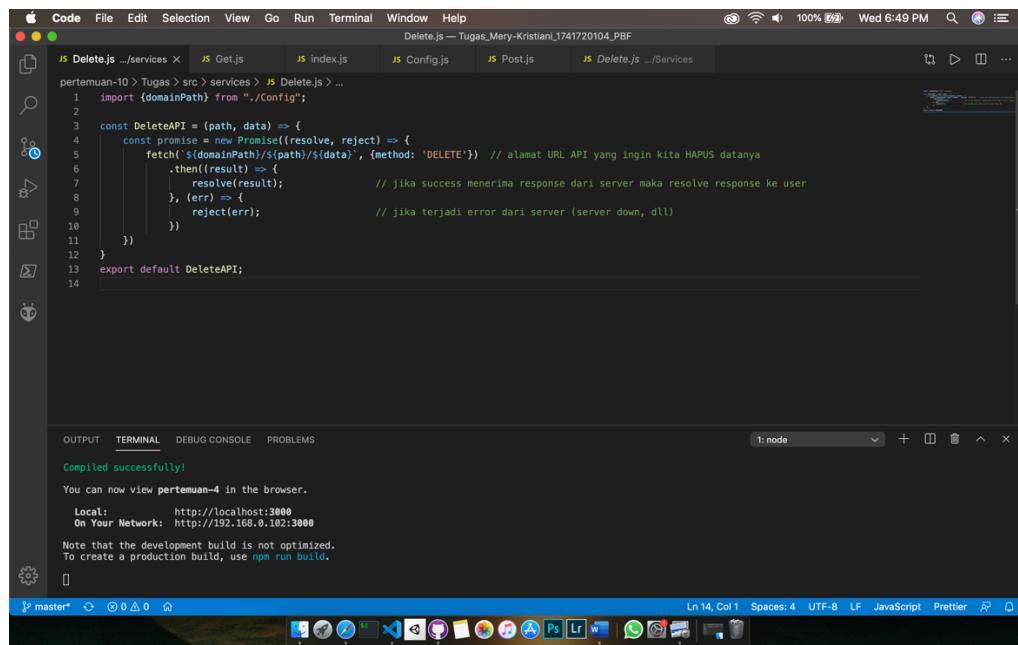
You can now view pertemuan-4 in the browser.

Local: http://localhost:3000
On Your Network: http://192.168.0.102:3000

Note that the development build is not optimized.
To create a production build, use `npm run build`.

Ln 22, Col 1 Spaces: 4 UTF-8 LF JavaScript Prettier ⌂ ⌂

e. Buka **Delete.js** dan isikan kode seperti gambar dibawah ini.



```
JS Delete.js .../services < JS Get.js JS index.js JS Config.js JS Post.js JS Delete.js .../Services
pertemuan-10 > Tugas > src > services > JS Delete.js ...
1 import {domainPath} from './Config';
2
3 const DeleteAPI = (path, data) => {
4     const promise = new Promise((resolve, reject) => {
5         fetch(`${domainPath}/${path}/${data}`, {method: 'DELETE'}) // alamat URL API yang ingin kita HAPUS datanya
6             .then(result) => {
7                 resolve(result); // jika success menerima response dari server maka resolve response ke user
8             }, (err) => {
9                 reject(err); // jika terjadi error dari server (server down, dll)
10            })
11        }
12    }
13
14 export default DeleteAPI;
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

Compiled successfully!

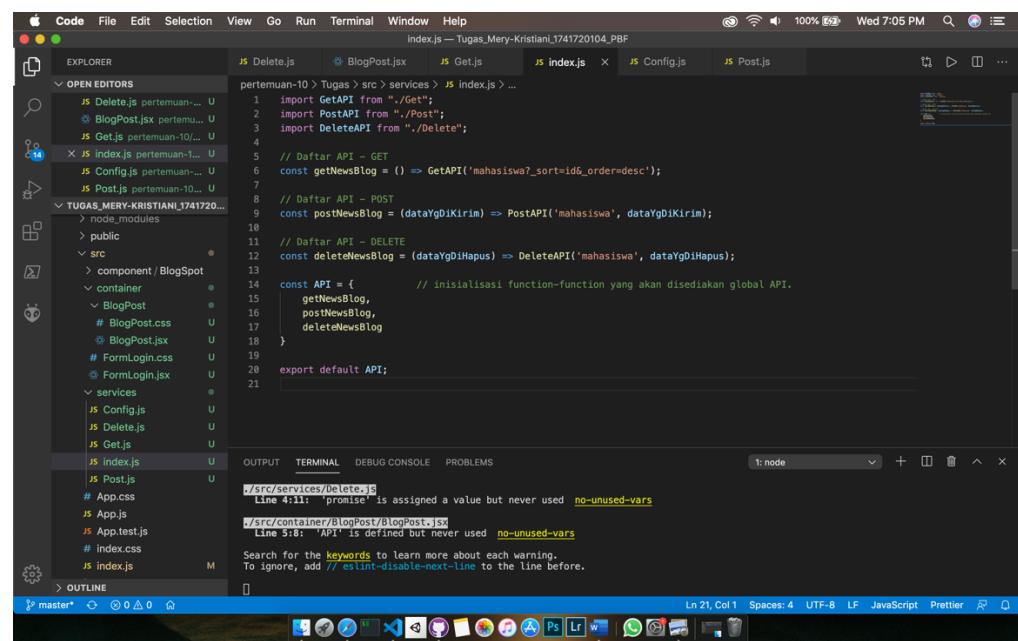
You can now view pertemuan-4 in the browser.

Local: http://localhost:3000
On Your Network: http://192.168.0.102:3000

Note that the development build is not optimized.
To create a production build, use `npm run build`.

Ln 14, Col 1 Spaces: 4 UTF-8 LF JavaScript Prettier ⌂ ⌂

f. Buka **index.js** dan isikan kode seperti gambar dibawah ini.



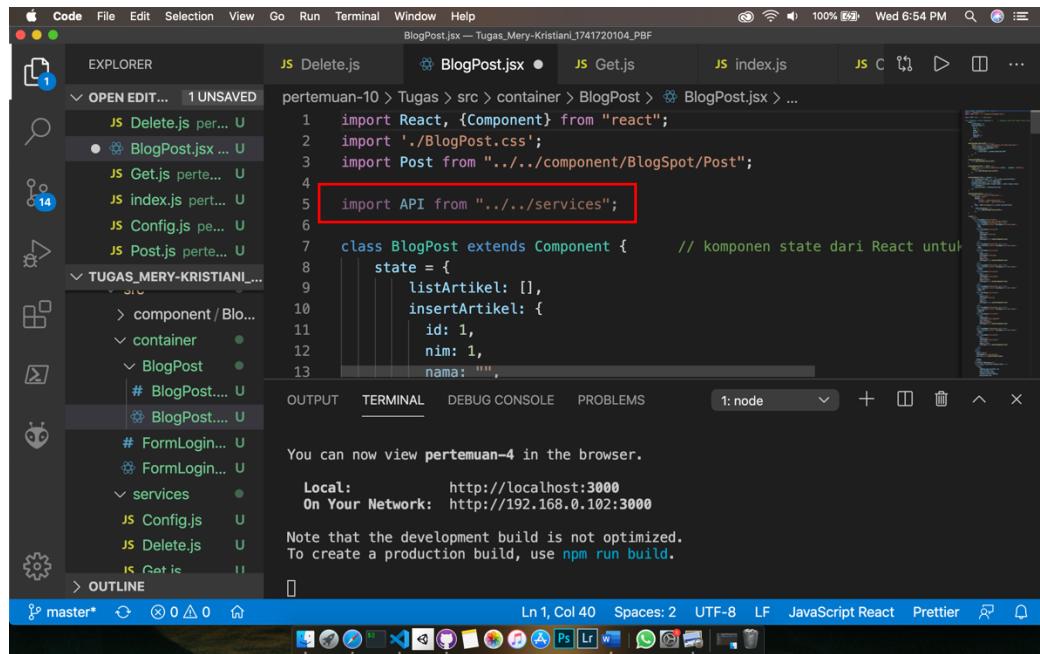
```
JS Delete.js JS BlogPost.jsx JS index.js JS Config.js JS Post.js
pertemuan-10 > Tugas > src > services > JS index.js ...
1 import GetAPI from './Get';
2 import PostAPI from './Post';
3 import DeleteAPI from './Delete';
4
5 // Daftar API - GET
6 const getNewsBlog = () => GetAPI('mahasiswa?_sort=id&_order=desc');
7
8 // Daftar API - POST
9 const postNewsBlog = (dataYgDiKirim) => PostAPI('mahasiswa', dataYgDiKirim);
10
11 // Daftar API - DELETE
12 const deleteNewsBlog = (dataYgHapus) => DeleteAPI('mahasiswa', dataYgHapus);
13
14 const API = { // inisialisasi function-function yang akan disediakan global API.
15     getNewsBlog,
16     postNewsBlog,
17     deleteNewsBlog
18 }
19
20 export default API;
```

./src/services/Delete.js
Line 4:11: 'promise' is assigned a value but never used no-unused-vars
./src/container/BlogPost/BlogPost.jsx
Line 5:8: 'API' is defined but never used no-unused-vars

Search for the **keywords** to learn more about each warning.
To ignore, add `// eslint-disable-next-line` to the line before.

Ln 21, Col 1 Spaces: 4 UTF-8 LF JavaScript Prettier ⌂ ⌂

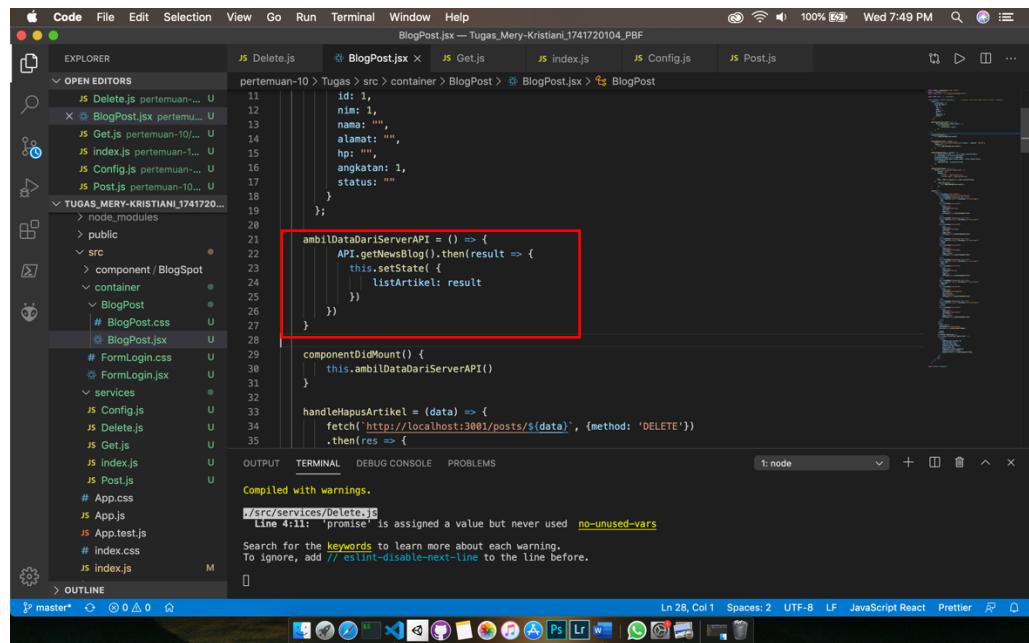
g. Mengimport API dari folder services.



The screenshot shows the VS Code interface with the file `BlogPost.jsx` open in the editor. The code imports several components and modules from other files. A red box highlights the line `import API from "../../services";`, which imports the `API` module from the `services` directory. The code also includes a class definition for `BlogPost` and a note about viewing the application in the browser.

```
1 import React, {Component} from "react";
2 import './BlogPost.css';
3 import Post from "../../component/BlogSpot/Post";
4
5 import API from "../../services";
6
7 class BlogPost extends Component {
8     state = {
9         listArtikel: [],
10        insertArtikel: {
11            id: 1,
12            nim: 1,
13            nama: ""
14        }
15    }
16}
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
```

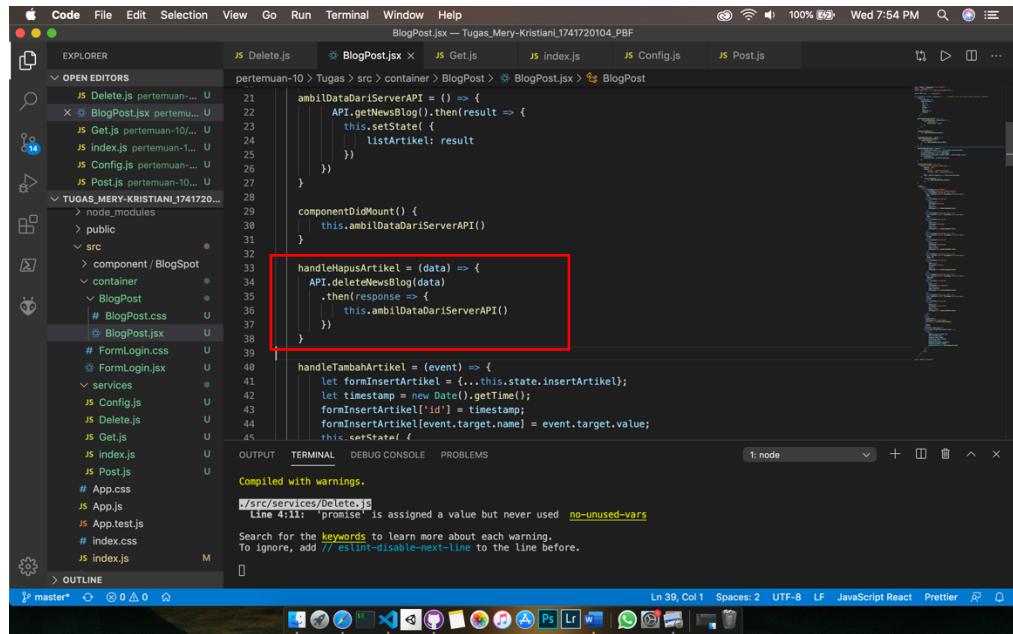
h. Buka `BlogPost.jsx` pada folder `BlogPost` dan ubah kode pada bagian `ambilDataDariServerAPI` seperti gambar dibawah ini.



The screenshot shows the VS Code interface with the file `BlogPost.jsx` open in the editor. The code defines a function `ambilDataDariServerAPI` that makes a call to the `API.getNewsBlog()` method and updates the component state with the result. A red box highlights this function definition. The code also includes a `componentDidMount` lifecycle method that calls this function.

```
11     id: 1,
12     nim: 1,
13     nama: "",
14     alamat: "",
15     hp: "",
16     angkatan: 1,
17     status: ""
18 };
19
20
21 ambilDataDariServerAPI = () => {
22     API.getNewsBlog().then(result => {
23         this.setState({
24             listArtikel: result
25         })
26     })
27 }
28
29
30
31
32
33
34
35
```

- i. Buka **BlogPost.jsx** pada folder **BlogPost** dan ubah kode pada bagian **handleHapusArtikel** seperti gambar dibawah ini.



```

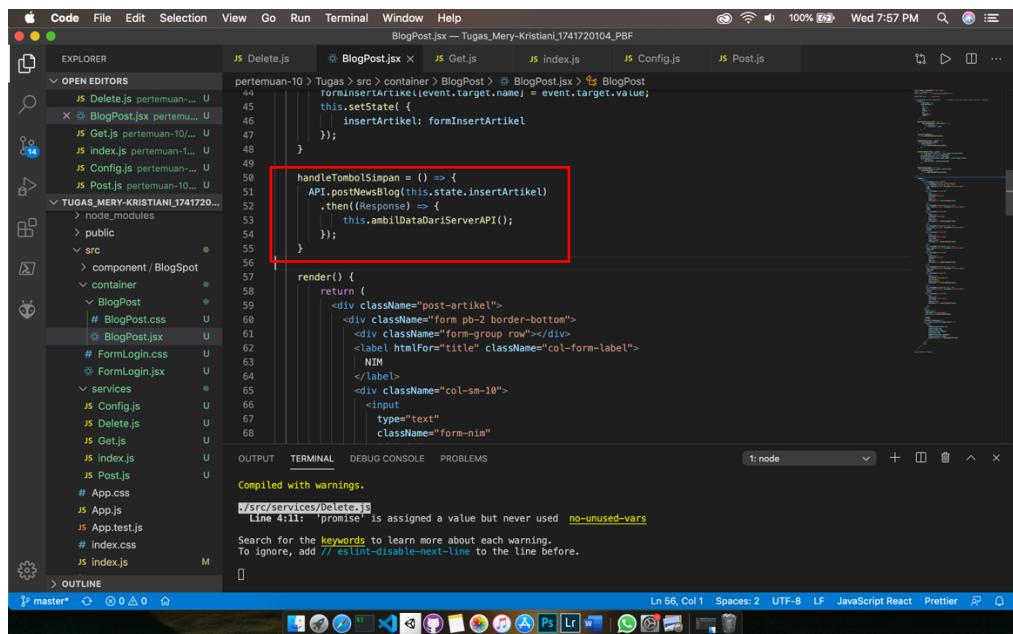
    ambilDataDariServerAPI = () => {
      API.getNewsBlog().then(result => {
        this.setState({
          listArtikel: result
        })
      })
    }

    componentDidMount() {
      this.ambilDataDariServerAPI()
    }

    handleHapusArtikel = (data) => {
      API.deleteNewsBlog(data)
      .then(response => {
        this.ambilDataDariServerAPI()
      })
    }

    handleTambahArtikel = (event) => {
      let formInsertArtikel = {...this.state.insertArtikel};
      let timestamp = new Date().getTime();
      formInsertArtikel['id'] = timestamp;
      formInsertArtikel[event.target.name] = event.target.value;
      this.setState({
        insertArtikel: formInsertArtikel
      })
    }
  
```

- j. Buka **BlogPost.jsx** pada folder **BlogPost** dan ubah kode pada bagian **handleTombolSimpan** seperti gambar dibawah ini.



```

    formInsertArtikel[event.target.name] = event.target.value;
    this.setState({
      insertArtikel: formInsertArtikel
    });

    handleTombolSimpan = () => {
      API.postNewsBlog(this.state.insertArtikel)
      .then(Response => {
        this.ambilDataDariServerAPI();
      });
    }

    render() {
      return (
        <div className="post-artikel">
          <div className="form pb-2 border-bottom">
            <div className="form-group row"></div>
            <label htmlFor="title" className="col-form-label">
              NIM
            </label>
            <div className="col-sm-10">
              <input
                type="text"
                className="form-nim"
            
```