

TUGAS INDIVIDU
Modul 8
Pemrograman Berbasis Framework

Oleh:

Alfalah Anugara

1741720159

TI-3A / 02



PROGRAM STUDI – D4 TEKNIK INFORMATIKA

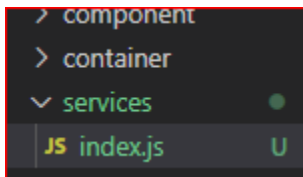
JURUSAN – TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

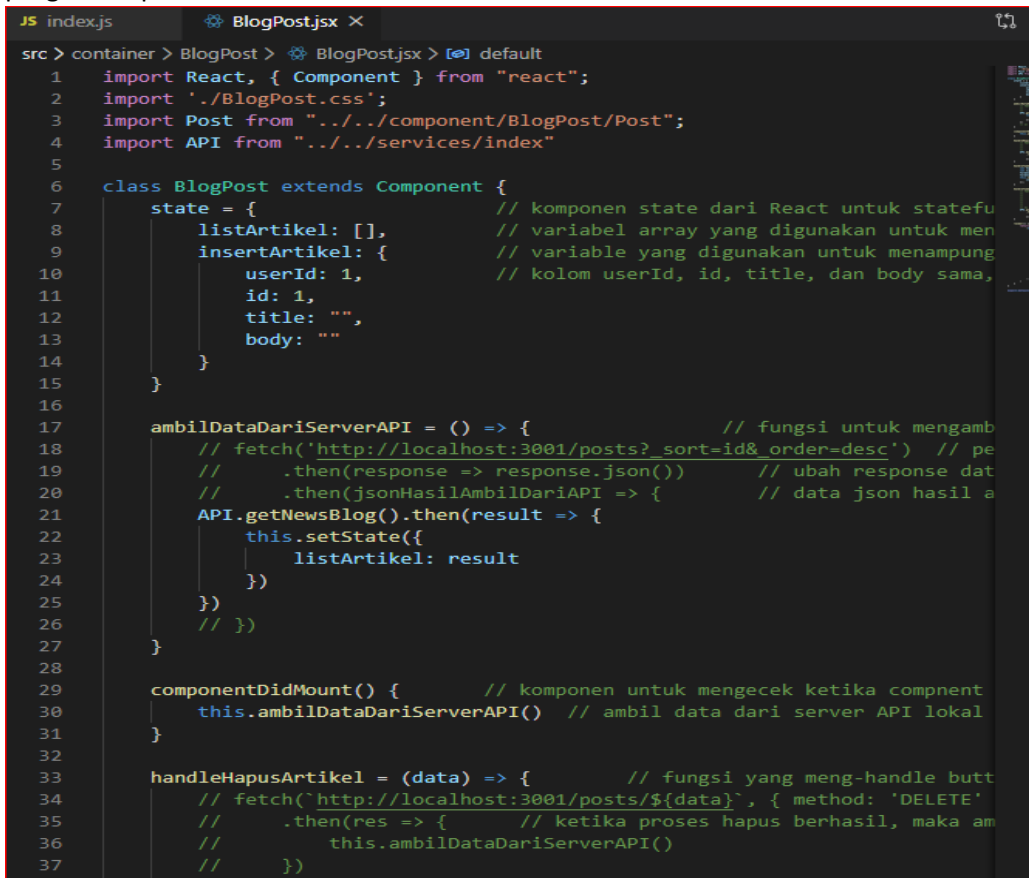
2020/2021

1.2 Langkah Praktikum

1. Buat folder baru bernama "Services" dalam folder **src** , kemudian buat file index.js seperti pada Gambar 1.1



2. Buka file BlogPost.jsx pada folder container (statefull component) dan akan tampil kode program seperti Gambar 1.2



3. Pada fungsi `ambilDataDariServerAPI` (baris16) terdapat pemanggilan API GET untuk merequest data artikel. Proses dari fungsi inilah yang akan kita manage kedalam satu tempat yaitu `index.js`

```
17  ambilDataDariServerAPI = () => {  
18    // fetch('http://localhost:3001/posts?_sort=  
19    // .then(response => response.json())  
20    // .then(jsonHasilAmbilDariAPI => {  
21    API.getNewsBlog().then(result => {  
22    this.setState({  
23      listArtikel: result  
24    })  
25    })  
26    // })  
27  }  
28
```

4. Buka file `index.js` pada folder `service` , yang telah kita buat tadi dan tuliskan baris kode seperti gambar 1.3 berikut

```
JS index.js  X  BlogPost.jsx  
  
src > services > JS index.js > [0] default  
1  import React from 'react';  
2  import getAPI from './Get';  
3  import PostAPI from './Post';  
4  import DeleteAPI from './Delete';  
5  
6  const getNewsBlog = () => getAPI("posts?_sort=id&_order=asc")  
7  const postNewsBlog = (dataYangDikirim) => PostAPI("posts", dataYangDikirim)  
8  const deleteNewBlog = (dataYgDihapus) => DeleteAPI("posts", dataYgDihapus)  
9  
10 const API = {  
11   getNewsBlog,  
12   postNewsBlog,  
13   deleteNewBlog  
14 }  
15  
16 export default API;
```

5. Kembali ke `BlogPost.js` Ganti baris 17 sampai baris 23 menjadi seperti Gambar 1.4

```
17  ambilDataDariServerAPI = () => {  
18    // fetch('http://localhost:3001/posts?_sort=  
19    // .then(response => response.json())  
20    // .then(jsonHasilAmbilDariAPI => {  
21    API.getNewsBlog().then(result => {  
22    this.setState({  
23      listArtikel: result  
24    })  
25    })  
26    // })  
27  }  
28
```

1.3 Pertanyaan Praktikum 1

1. Perhatikan file index.js apa tujuan dibuatnya fungsi GetAPI pada baris 2 dan fungsi getNewsBlog pada baris 16?

Untuk request data artikel yang akan dimanage dalam satu file yaitu di index.js

2.1 Langkah Praktikum 2

1. Kita perhatikan pada fungsi handleTombolSimpan pada file BlogPost.js. Bagian inilah yang selanjutnya kita kelola agar bisa di-manage.

```
handleTombolSimpan = () => { // fungsi untuk meng-handle tombol simpan
  // fetch('http://localhost:3001/posts', { // method POST untuk input/insert data
  //   method: 'post',
  //   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)
  // })
  API.postNewsBlog(this.state.insertArtikel)
    .then((response) => {
      this.ambilDataDariServerAPI(); // reload / refresh data
    });
}
```

2. Buatlah fungsi untuk menampung action POST dari file BlogPost.js pada file services/index.js dan inialisasi fungsi tersebut seperti pada Gambar 2.2

```
src > services > JS index.js > [?] getNewsBlog
1  import React from 'react';
2  import getAPI from './Get';
3  import PostAPI from './Post';
4  import DeleteAPI from './Delete';
5
6  const getNewsBlog = () => getAPI(["posts?_sort=id&_order=asc"])
7  const postNewsBlog = (dataYangDikirim) => PostAPI("posts", dataYangDikirim)
8  const deleteNewBlog = (dataYgDihapus) => DeleteAPI("posts", dataYgDihapus)
9
10 const API = {
11   getNewsBlog,
12   postNewsBlog,
13   deleteNewBlog
14 }
15
16 export default API;
```

3. Selanjutnya pindah ke file BlogPost.js dang anti isi dari fungsi handleTombolSimpan menjadi seperti Gambar 2.3

```
handleTombolSimpan = () => { // fungsi untuk meng-handle tombol simpan
  // fetch('http://localhost:3001/posts', {
  //   method: 'post', // method POST untuk input/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)
  // })
  API.postNewsBlog(this.state.insertArtikel)
  .then((response) => {
    this.ambilDataDariServerAPI(); // reload / refresh data
  });
}
```

4. Selesai. Kode Program pada BlogPost.js sedikit lebih simple sebelumnya
5. Silahkan kalian jalankan proses input artikel melalui browser dan amati apa yang terjadi

2.2 Pertanyaan Praktikum 2

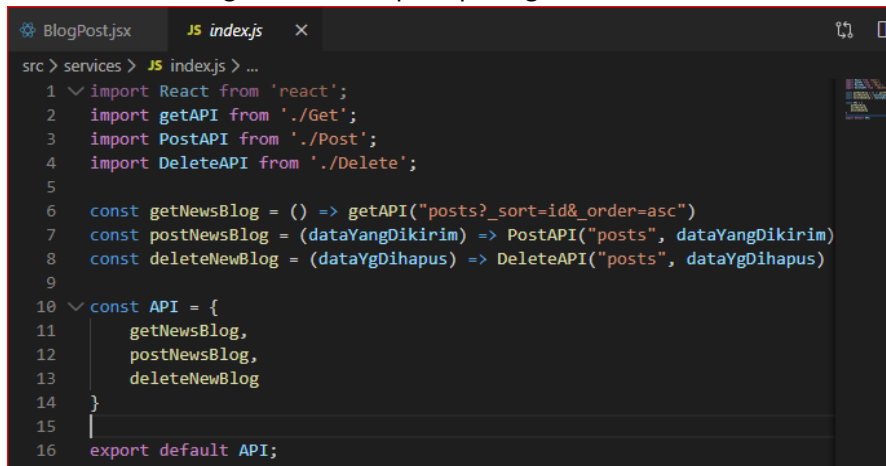
1. Perhatikan file index.js apa tujuan dibuatnya fungsi postApi dan fungsi PostNewsBlog?
untuk mengirimkan data kepada server API.
2. Pada fungsi postNewsBlog , terdapat variable dataYangDiKirim. Apa tujuan dari dibuatnya variable tersebut?
Tujuannya adalah untuk menampung data sementara yang akan dikirim , kemudian ditujukan ke PostAPI untuk mengirimkan ke pada server API

3.1 Langkah Praktikum 3

1. Kita perhatikan pada fungsi handleHapusArtikel pada file Blogpost.jsx. Bagian inilah yang selanjutnya kita kelola agar bisa di-manage

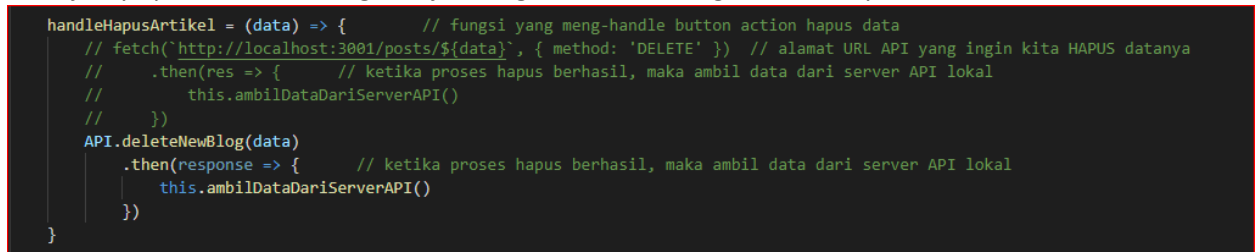
```
handleHapusArtikel = (data) => { // fungsi yang meng-handle button action hapus data
  // fetch('http://localhost:3001/posts/${data}', { method: 'DELETE' }) // alamat URL API yang ingin kita HAPUS datanya
  // .then(res => { // ketika proses hapus berhasil, maka ambil data dari server API lokal
  //   this.ambilDataDariServerAPI()
  // })
  API.deleteNewBlog(data)
  .then(response => { // ketika proses hapus berhasil, maka ambil data dari server API lokal
    this.ambilDataDariServerAPI()
  })
}
```

2. Buatlah fungsi untuk menampung action DELETE dari file BlogPost.jsx pada file services/index.js dan inialisasi fungsi tersebut seperti pada gambar 3.2



```
src > services > JS index.js > ...
1  import React from 'react';
2  import getAPI from './Get';
3  import PostAPI from './Post';
4  import DeleteAPI from './Delete';
5
6  const getNewsBlog = () => getAPI("posts?_sort=id&_order=asc")
7  const postNewsBlog = (dataYangDikirim) => PostAPI("posts", dataYangDikirim)
8  const deleteNewBlog = (dataVgDihapus) => DeleteAPI("posts", dataVgDihapus)
9
10 const API = {
11   getNewsBlog,
12   postNewsBlog,
13   deleteNewBlog
14 }
15
16 export default API;
```

3. Selanjutnya pindah ke file BlogPost.jsx dang anti isi dari fungsi handleHapusArtikel



```
handleHapusArtikel = (data) => { // fungsi yang meng-handle button action hapus data
  // fetch('http://localhost:3001/posts/${data}', { method: 'DELETE' }) // alamat URL API yang ingin kita HAPUS datanya
  // .then(res => { // ketika proses hapus berhasil, maka ambil data dari server API lokal
  //   this.ambilDataDariServerAPI()
  // })
  API.deleteNewBlog(data)
  .then(response => { // ketika proses hapus berhasil, maka ambil data dari server API lokal
    this.ambilDataDariServerAPI()
  })
}
```

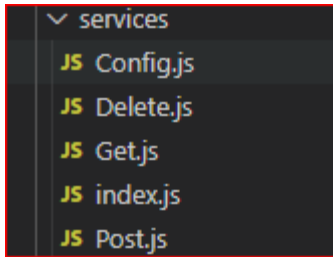
4. Kode program pada BlogPost.jsx lebih sedikit lebih simple darii sebelumnya
5. Silakan kalian jalankan proses hapus artikel melalui browser dan amatii apa yang terjadi

3.2 Pertanyaan Praktikum 3

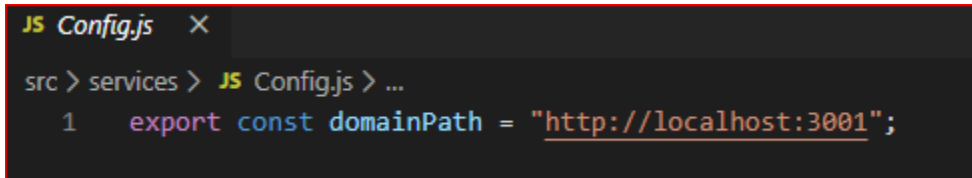
1. Perhatikan file index.js , apa tujuan dibuatnya fungsi DeleteAPI dan fungsi deleteNewsBlog?
Tujuannya adalah untuk melakukan request hapus data pada server API
2. Pada fungsi deleteNewsBlog , terdapat variable dataYangDiHapus. Apa Tujuan dari skill dibuatnya variable tersebut?
Untuk menyimpan sementara data yang akan dihapus, kemudian diteruskan kepada deleteAPI yang nantinya akan melakukan penghapusan pada data di server API.

4.2 Langkah Praktikum 4

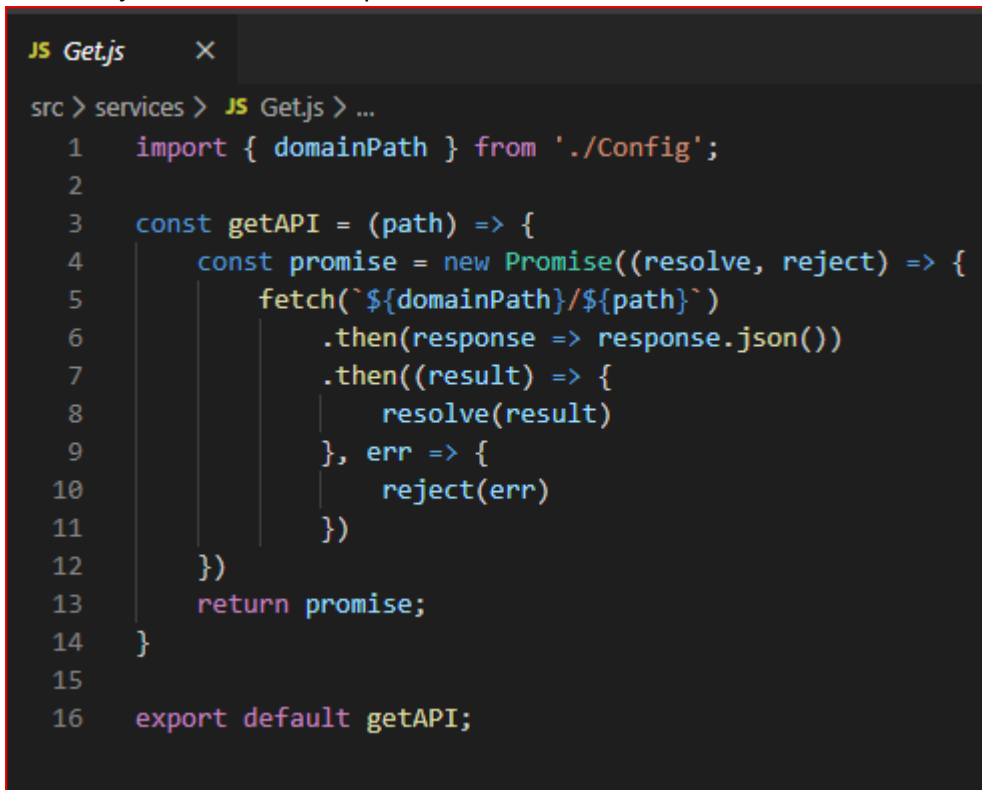
1. Buatlah file Config.js , Get.js , Post.js , dan Delete.js dalam folder services seperti pada Gambar



2. Buka config.js dan isikan kode seperti Gambar 4.2



3. Buka Get.js dan isikan kode seperti Gambar 4.3



4. Buka Post.js dan isikan kode seperti Gambar 4.4



```
JS Post.js X
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",
7              headers: {
8                  'Accept': 'application/json',
9                  'Content-Type': 'application/json'
10             },
11             body: JSON.stringify(data)
12         })
13         .then((result) => {
14             resolve(result)
15         }, err => {
16             reject(err)
17         })
18     })
19     return promise;
20 }
21
22 export default PostAPI;
```


5. Buka delete.js dan isikan kode seperti Gambar 4.5

```
JS Delete.js X
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}`, {
6              method: "DELETE"
7          })
8              .then((result) => {
9                  resolve(result)
10             }, err => {
11                 reject(err)
12             })
13         })
14         return promise;
15     }
16
17     export default DeleteAPI;
```

6. Pada Index.js kita edit menjadi seperti Gambar 4.6

```
JS index.js X
src > services > JS index.js > [⌕] default
1  import React from 'react';
2  import getAPI from './Get';
3  import PostAPI from './Post';
4  import DeleteAPI from './Delete';
5
6  const getNewsBlog = () => getAPI("posts?_sort=id&_order=asc")
7  const postNewsBlog = (dataYangDikirim) => PostAPI("posts", dataYangDikirim)
8  const deleteNewBlog = (dataYgDihapus) => DeleteAPI("posts", dataYgDihapus)
9
10  const API = {
11      getNewsBlog,
12      postNewsBlog,
13      deleteNewBlog
14  }
15
16  export default API;
```

4.3 Pertanyaan Praktikum 4

1. Bagaimana caranya untuk menambahkan daftar API baru baik untuk method GET, POST , DELETE pada Global API?
dengan cara membuat file baru dengan nama Get post delete dengan ekstensi js