**PEMROGRAMAN BERBASIS FRAMEWORK**

**Minggu 4**

****

**Raihan Rachmadani**

**NIM 1841720167**

**D4 TEKNIK INFORMATIKA**

**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

**Langkah Praktikum**

Dalam penggunaan API pada suatu website, maka data yang akan kita pakai adalah data dinamis dan memerlukan operasi logic pada ReactJS. Sehingga kita akan menggunakan statefull component ReactJS untuk membuatnya.

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| **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”, seperti pada Gambar di bawah ini |
| **4.** | Buka file BlogPost.jsx dan ketikkan kode seperti Gambar di bawah ini |
| **5.** | Pada file index.js, lakukan import component BlogPost seperti Gambar di bawah |
| **6.** | Pada web browser akan tampil seperti pada Gambar di bawah ini |

Tahapan selanjutnya adalah perbaikan tampilan sebuah website untuk mempercantik halaman website tersebut dengan menggunakan Bootstrap yang umum digunakan.

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| **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” |
| **8.** | Modifikasi file index.html pada folder "public" seperti Gambar 1.7. Cermati code program yang ada dalam gambar!. |
| **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 |
| **11.** | Perhatikan kembali browser, dan lihat hasil tampilan seperti Gambar di bawah |

Kita ingin sebuah website memiliki tampilan seperti pada Gambar 1.1. Dengan minimal ada gambar artikel, judul, dan deskripsi artikel. Maka contoh data dummy yang akan kita pakai bisa menggunakan data dari http://placeimg.com contoh http://placeimg.com/120/120/any. Tahapan edit tampilan post artikel:

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| **12.** | Ubah kode program untuk statefull component BlogPost.jsx menjadi seperti Gambar di bawah |
| **13.** | Tambahkan custom css ke BlogPost.css seperti Gambar di bawah |
| **14.** | Perhatikan tampilan browser. |

**Pemindahan dari statefull component ke stateless component**

Pada component BlogPost (lihat Gambar 1.11), baris 9-17 merupakan daftar artikel yang bisa jadi dalam sebuah website berisi lebih dari 1 (satu) list artikel. Baris 9-17 dapat dipindah ke stateless component untuk dapat digunakan ulang (dipanggil kembali) karena fungsi dari bagian tersebut hanya mengembalikan deskripsi singat artikel (bukan operasi logic).

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| **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 di bawah |
| **17.** | Untuk statefull component BlogPost.jsx pada baris 10, panggil stateless component Post.jsx seperti Gambar di bawah |
| **18.** | Perhatikan hasil tampilan browser, apa yang terjadi?    Hasilnya tetap |

**Muat Data Dinamis.**

Bagaimana caranya untuk dapat membuat data dinamis (lebih dari 1 artikel) dimana data Judul dan Deskripsi pada artikel didapat dari API?

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| **19.** | Pada statefull component BlogPost.jsx, tambahkan parameter yang ingin dilempar ke stateless component untuk ditampilkan. Kode program bisa dilihat pada Gambar |
| **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!. |
| **21.** | Simpan, dan amati apa yang terjadi pada browser kalian!. |

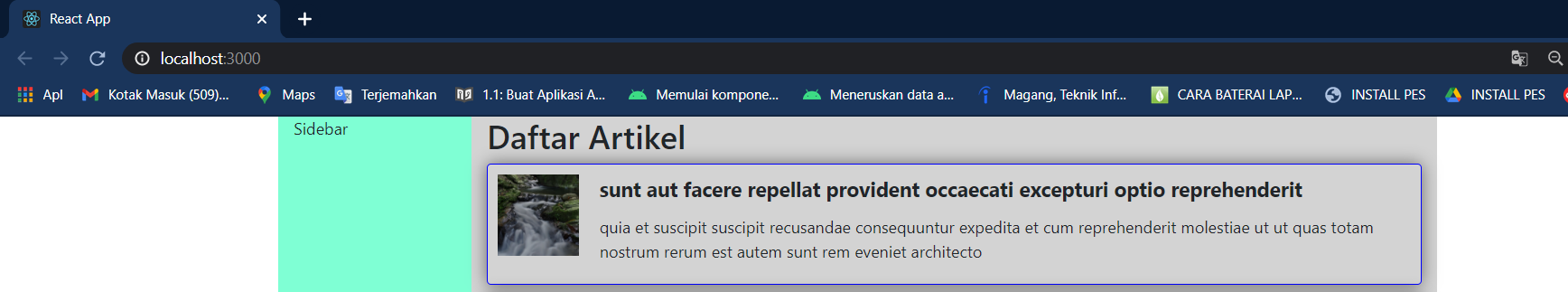
**Mengambil data Post/Artikel dari API.**

Bagaimana caranya untuk mendapatkan list artikel berdasarkan data json dari web API (contohnya: <https://jsonplaceholder.typicode.com/posts> ) ? Kita gunakan life cycle component yaitu componentDidMount() dimana ketika komponen selesai dimount-ing, program akan memanggil API.

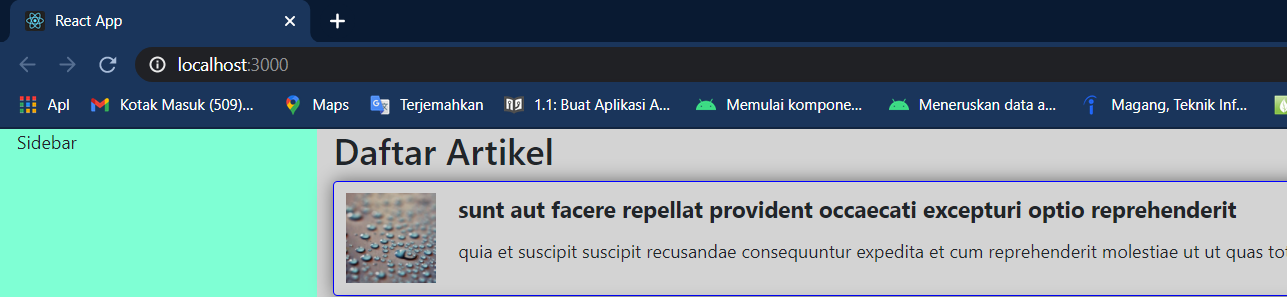
|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| **22.** | Gunakan state untuk menyimpan data hasil request dariAPI |
| **23.** | data API yang akan kita gunakan adalah data dummy dari <https://jsonplaceholder.typicode.com/posts> , dimana memiliki 4 element data yaitu userid, id, title, body (seperti pada Gambar 1.17) |
| **24.** | Edit pada statefull component BlogPost.jsx seperti pada Gambar 1.18 dan perhatikan dengan seksama akan penjelasan dibeberapa baris kode program tersebut. |
| **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 padaprogram 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.** | Simpan dan lihat apa yang terjadi pada console browser (Gambar 1.21). |

**Pertanyaan Praktikum 1**

1. Pada langkah 8, sekarang coba kalian ganti class container dengan container-fluid atau sebaliknya pada file "public/index.html" dan lihat apaperbedaannya
2. **Container**

****

**Container-fluid**

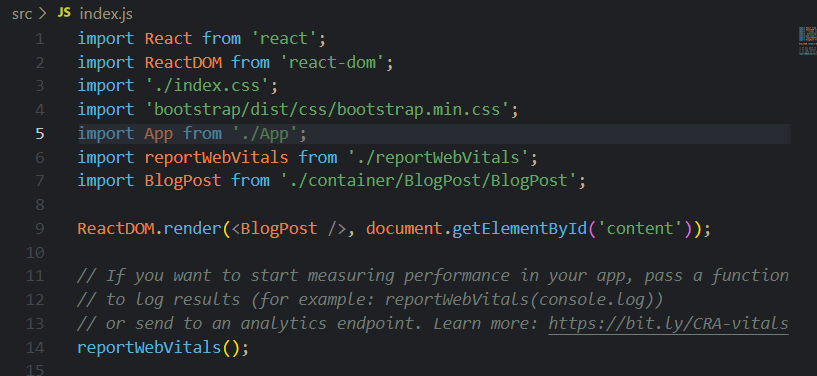
****

1. Apa perbedaan dari container dan container-fluid ?

Jawab : Terdapat jarak/spasi jika menggunakan container saja

1. 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?

Jawab : install lewat cmd “npm install bootstrap”. Setelah selesai import komponen bootstrap nya

****

**Praktikum 2**

**Interaksi dengan API menggunakan Fake API**

**Install Fake API (JSON Server)**

Fake API/JSON Server bisa kita dapatkan di halaman https://github.com/typicode/jsonserver. Tahapan install dan membuat data json sendiri

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| **1.** | Install pada direktori project reactjs kita dengan perintah npm install -gjson-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 |
| **5.** | Kita cek url resource yang ada padaFake APIserver ke browser apakah bisa diakses. Ketik url <http://localhost:3001/posts> pada browser. |
| **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.    Tidak berubah, masih sama |

**Pertanyaan Praktikum 2**

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

**Jawab** : menurut saya, Karena port 3000 sudah dipakai pada projek sebelumnya. Dan jika m emakai port yang sama (pakai port 3000) akan tertimpa, dan projek sebelumnya tidak bisa keluar

1. Bagaimana jadinya kalua kita ganti port json-server menjadi 3000?

Jawab : Tidak akan keluar projek pada halaman web

**Praktikum 3**

**Interaksi dengan API menggunakan method DELETE**

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| **1.** | Buka stateless component Post. Tambahkan 1 baris kode program pada baris 10 seperti pada Gambar di bawah |
| **2.** | Kemudian pada statefull component BlogPost, modifikasi kode program sebelumnya sesuai dengan Gambar di bawah |
| **3.** | Vvdv    Nb: Jika ingin mengeluarkan tampilan seperti diatas, kedua port (3001 & 3000) harus sama sama dijalankan |

**Pertanyaan praktikum 3**

1. Apa yang terjadi setelah kalian klik tombol hapus?

**Jawab** : 1 konten dalam tampilan website akan terhapus

1. Perhatikan file listArtikel.json, apa yang terjadi pada file tersebut? Kenapa demikian?

**Jawab** : 1 data dalam listArtikel.json juga terhapus, karena listArtikel.json masuk ke dalam content

1. Fungsi handleHapusArtikel itu untuk apa?

**Jawab** : merupakan sebuah method/fungsi untuk menghapus 1 konten yang ingin dihapus

1. Jelaskan perbedaan fungsi componentDidMount() pada Gambar 1.18 dengan fungsi componentDidMount() pada Gambar 3.2 ?

**Jawab : Gambar 3.2** -> terdapat fungsi yang lebih simple untuk mengambil data dari server API local

**Gambar 1**.18 -> menurut saya, masih terdapat beberapa fungsi yang bisa dibilang masih banyak, dikarenakan fungsinya masih terpisah pisah untuk memanggil dan mengubah data nya masih manual

**Praktikum 4**

**Interaksi dengan API menggunakan method POST**

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| **1.** | Buka statefull component BlogPost, dan modifikasi pada fungsi render() untuk menampilkan form input artikel yang berisi judul dan isi berita. seperti pada Gambar 4.1 |
| **2.** | Kemudian modifikasi BlogPost untuk bagian state dan request API dari server, seperti Gambar 4.2 |
| **3.** | Tambahkan untuk handle form tambah data artikel seperti Gambar 4.3 |
| **4.** | Langkah terakhir tambahkan fungsi untuk handle tombol simpan artikel, seperti pada Gambar 4.4 |
| **5.** | Simpan, lakukan percobaan penambahan data, dan amati perubahannya. |

**Pertanyaan Praktikum 4**

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

**Jawab :** kalau sebelum ditambah, datanya masih lengkap. Jika sudah dilakukan penambahan data, makan dalam file listArtikel.json juga ikut bertambah juga

1. 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?

**Jawab :**

**TUGAS**

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| **1.** | Buat projek baru untuk tugasnya |
| **2.** | Buat file MahasiswaBlogPost.jsx dan MahasiswaBlogPost.css di dalam folder “BlogPost” |
| **3.** | Buka file MahasiswaBlogPost.jsx dan ketikkan kode seperti di bawah |
| **4.** | Pada file index.js, lakukan import component MahasiswaBlogPost seperti ini |
| **5.** | Pada web browser akan tampil seperti pada Gambar |
| **6.** | 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” (Disini saya sudah menginstall di praktikum atas) |
| **7.** | Modifikasi file index.html pada folder "public" seperti Gambar 1.7. Cermati code program yang ada dalam gambar! |
| **8.** | Amati tampilan yang ada pada browser |
| **9.** | Buka file index.css dan tambahkan code css seperti Gambar 1.9, untuk menambah sedikit style pada halaman web |
| **10.** | Perhatikan kembali browser, dan lihat hasil tampilan seperti Gambar |
| **11.** | Ubah kode program MahasiswaBlogPost.jsx menjadi seperti Gambar |
| **12.** | Tambahkan custom css ke BlogPost.css seperti Gambar |
| **13.** | Perhatikan tampilan browser. |
| **14.** | Buat folder BlogPost pada folder component, lalu buat file Post.jsx |
| **15.** | Potong (cut) baris 9-17 pada statefull component MahasiswaBlogPost.jsx ke stateless component MahasiswaPost.jsx, dan modifikasi MahasiswaPost.jsx seperti Gambar |
| **16.** | Untuk statefull component MahasiswaBlogPost.jsx pada baris 10, panggil stateless component MahasiswaPost.jsx seperti Gambar |
| **17.** | Perhatikan hasil tampilan browser, apa yang terjadi? |
| **18.** | Pada statefull component MahasiswaBlogPost.jsx, tambahkan parameter yang ingin dilempar ke stateless component untuk ditampilkan. Kode program bisa dilihat pada Gambar |
| **19.** | 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! |
| **20.** | Simpan, dan amati apa yang terjadi pada browser kalian! |
| **21.** | Menggunakan state untuk menyimpan data hasil request dari API |
| **22.** | data API yang akan kita gunakan adalah data dummy dari <http://localhost:3002/mahasiswa> dimana kita memiliki 6 element yaitu *nim, nama, alamat, hp, angkatan, status* |
| **23.** | Edit pada statefull component MahasiswaBlogPost.jsx seperti pada Gambar 1.18 dan perhatikan dengan seksama akan penjelasan dibeberapa baris kode program tersebut |
| **24.** | Simpan dan lihat hasilnya |
| **25.** | Install Fake API (JSON Server)  Install pada direktori projek kita. Buka cmd baru dan ketik perintah “*json-server --watch mahasiswal.json –-port 3002”* |
| **26.** | Apabila pada cmd tampil seperti Gambar 2.1, maka server Fake API local kita telah siap. |
| **27.** | Kita cek url resource yang ada padaFake APIserver ke browser apakah bisa diakses. Ketik url <http://localhost:3002/mahasiswa> pada browser. |
| **28.** | Untuk memastikan lagi, kita edit statefull component MahasiswaBlogPost (Gambar 1.18) pada baris 11. Kita ganti url API menjasi <http://localhost:3001/posts> |
| **29.** | Simpan dan refresh browser |
| **30.** | Buka stateless component MahasiswaPost. Tambahkan 1 baris kode program pada baris 12 seperti pada Gambar |
| **31.** | Kemudian pada statefull component MahasiswaBlogPost, modifikasi kode program sebelumnya sesuai dengan Gambar |
| **32.** | Buka browser lalu refresh. Awal mula data json ada 2 nama Rachmadani dan Raihan. Ketika tombol hapus di klik maka data akan berkurang 1 |
| **33.** | Buka statefull component MahasiswaBlogPost, dan modifikasi pada fungsi render() untuk menampilkan form input artikel yang berisi judul dan isi berita. seperti pada Gambar |
| **34.** | Kemudian modifikasi MahasiswaBlogPost untuk bagian seperti pada Gambar |
| **35.** | Tambahkan untuk handle form tambah data artikel seperti Gambar |
| **36.** | Langkah terakhir tambahkan fungsi untuk handle tombol simpan artikel, seperti pada Gambar |
| **37.** | Simpan, lakukan percobaan penambahan data, dan amati perubahannya.  Sebelum tambah data    Sesudah tambah data |

**LINK GITHUB :** [**https://github.com/raihanrach/Pemrograman-Berbasis-Framework**](https://github.com/raihanrach/Pemrograman-Berbasis-Framework)