|  |  |
| --- | --- |
| **Materi** | **Nilai** |
| Project Restoran Bag.1 (Vid 21-28) | 90 |

**Saya Sudah Belajar dan Mengerti dan Saya BISA**

1. Dapat meng-import bootstrap

2. Memahami penggunaan route untuk mengatur akses web

3. Memahami & dapat mengatur route path untuk akses back end dan front end yaitu

* pada app.js import Front & back
* juga {BrowserRouter as Router, route} from ‘react-router-dom’; untuk digunakan saat routing
* Lalu atur page aksesnya misal <Route path="/home" component={Front} /> artinya saat page /home maka termasuk ke dalam bagian front end
* <Route path="/admin" component={Back} /> makshudnya apabila page /admin maka akan masuk ke bagian back end

4. Dapat mengatur navbar untuk navigasi web app program yaitu :

* Membuat file untuk komponen navbar dalam folder front yaitu side,main dan footer karena akan mmebuat footer berbeda untuk front end dan back end
* Pada file front import semua elemen yaitu Nav,side,main,footer dan react
* Tampilkan elemen dengan menggunakan div Classname

5. Memahami cara mengatur navbar backend untuk link antar page :

* Pada side di backend tambahkan card dari bootstrap lalu isi dengan kategori,menu,pelanggan dsb
* Import Link dan useRouteMatch dari react-router-dom
* Buat const {url} = untuk menggunakan functionuseRouthMatch();
* Tambahkan Link pada tiap opsi navbar dan arahkan ke to={`${url}/Kategori`} hal ini berguna agar kolom backend ter indikasi pada link Admin/Kategori kalau tidak maka akan menjadi /Kategori saja tanpa indikasi apakah ini backend atau front end
* Pada file content import useParams dari react-router-dom lalu but const {isi} = untuk menggunakan function useParams();
* Pada return nya isi untuk menampilkan const isi cnth: <h1>{isi}</h1>
* Pada file main import Content dari file content untuk menampilkan isi content
* Lalu import {Switch,Route,useRouteMatch} dari react-router-dom
* Di dalam const Main buat const {path} = untuk menggunakan useRouteMatchh(); kali ini digunakan untuk menampilkan isi konten berdasarkan link yang di klik
* Di dalam returm isi dengan Switch lalu Route path={`${path}/:isi`} karna menggunakan const isi yg ada pada content
* Jadi saat link navbar di klik maka akan menampilkan konten berdasarkan apa yang di klik tadi namun tetap di halaman tersebut yang berubah hanya tampilan dan isinya saja

6. Memahami fungsi function useRouteMatch untuk mengambi nilai route ketika link navigasi di klik

7. Memahami fungsi function useParams untuk mengisi route guna menampilkan content sesuai route/link nya

8. Memahami bahwa nested route merupakan membuat route di dalam route

9. Memahami & dapat memanggil isi dari suatu file komponen jika button navbar nya di klik:

* Pada content.js membuat variable untuk wadah komponen nantinya
* Masih di folder back, membuat new file untuk Kategori,Menu,Pelanggan lalu isi file dengan rhc dan beri dulu isi sementara h1 sesuai nama
* Pada content sebelum return buat 3 if masing2 untuk kategori,menu,pelanggan kondisinya if (isi === ‘Kategori’ isi nama nya sesuai pada controller dahulu huruf besar kecilnya berpengaruh) lalu {var untuk wadah = <Pelanggan />} untuk memanggil isi nya
* Import new file tadi agar dapat di panggil
* Tampilkan dengan menggunakan return {var untuk wadah}
* Jadi apabila button link di klik maka link akan menyesuaikan dan isi konten akan sesuai pada file nya masing yg sudah di panggil.

10. Memahami & dapat memanggil data dari database untuk ditampilkan menggunakan axios :

* Copy file link dari folder layout lalu paste di folder back pada react-resto
* Import axios karena sudah di install jadi tidak perlu cdn
* Pada file Kategori yg sudah di buat tadi import file link serta useState & useEffect
* Buat const array [isi, setIsi] = useState([]);
* Gunakan useEffect untuk request data isi dengan async function lalu isi function dengan const request = await link.get('/file nya Kategori'); ini untuk memanggil data
* Lalu gunakan setIsi(request.data); (yg sudah di deklarasikan pada array const tadi untuk wadah)
* Lalu di luar function di dalam useEffect jalankan function();
* Pada return isi dengan table untuk menampilkan isi kategori & keterangan sesuai seperti pada database
* Tag <table> isi dengan <tr><th> untuk table-head dan <tr><td> untuk isi table
* Setelah table-head untuk menampilkan isi database maka menggunakan { isi(var yg sudah di deklarasikan di array const tadi).map((val,index) => ( <tr key={index}> <td>{val.kategori}</td> makshudnya adalah menampilkan isi database menggunakan key dan value karena isinya berbentuk array maka dari itu untuk key nya index untuk di sesuaikan pada table database dan val sebagai value nya.
* Jadi isi database di tampilkan menggunakan const array isi dan function map
* Const di deklarasikan dalam bentuk array karena data yg akan di tampilkan berbentuk array.
* Untuk memanggil data ini juga perlu axios karena berperan dalam penyaluran link dan token untuk menyambung ke api-lumen

11. Dapat membuat form menggunakan react hook form :

* Pada file kategori import {useForm} from ‘react-hook-form’;
* Buat const {register,handleSubmit,reset,errors} = gunakan function useForm(); seperti pada dokumentasi
* Buat function simpan(data) untuk memanggil data yang sudah di ambil {console.log(data) lalu gunakan function reset(); agar data tidak selalu ada dan hanya ada ketika di panggil saja.
* Pada return buat 3 row untuk judul,form dan tabel data
* Pada row form isi dengan div class col-4 lalu tag form menggunakan onSubmit={function handleSubmit(simpan)}> disini makshudnya adalah untuk menggunakan function handleSubmit ketika button submit di klik(onSubmit) untuk menjalankan function simpan
* Lalu isi form dengan template card pada bootstrap sesuaikan untuk kategori & keterangan lalu tambahkan ref={register({required:true})} hal ini berguna untuk pengecualian/ketentuan yang apabila button submit di klik tanpa ada form yang di isi maka akan menampilkan pesan error
* Menampilkan pesan error menggunakan {errors.elemen inputan && “message error”}

12. Memahami & dapat meng-insert data dengan metode axios:

* Pada api-lumen KategoriController pada function create tambahkan if yang apabila data berhasil di tambahkan maka akan return dan isi respon json nya dengan array berisi [‘pesan’=>’pesan yg akan ditampilkan’] karena akan di kirimkan ke page front
* Buat const [pesan,setPesan] = useState([‘’]); ini untuk menampilkan pesan ketika data berhasil di inputkan dan harus menggunakan use state karena
* Pada function simpan isi dengan link.post(‘./Kolom yg akan di insert data’,data).then(res=> setPesan(res.data.pesan));
* Agar respon data langsung ber-effect pada tabel kategori di page front saat proses input data berhasil maka perlu mengeluarkan async function fetchData dari dalam useEffect lalu sisakan menggunakan fetchData() saja
* Kemudian buat row baru untuk menampilkan pesan dari KategoriController yang sudah di handle oleh useState tampilkan menggunakan misal <h2>{pesan}</h2>
* Tambahkan variable untuk nomer tabel dengan metode looping lalu tampilkan pada table <td>{no++></td>

13. Memahami & dapat men-delete data dengan metode axios:

* Menambahkan kolom untuk hapus pada tabel juga button nya dengan onClick menjalankan function hapus(val.idkategori) karena menggunakan idkategori sebagai penanda data yang akan di hapus lalu val merupakan key data yang digunakan (karna data berbentuk array jadi untuk mempengaruhi nya memerlukan key juga value).
* Buat function hapus dengan params id karena id yang digunakan sebagai acuan data mana yang akan di hapus
* Lalu isi dengan if (window.confirm(‘Yakin akan menghapus data’)) ini digunakan sebagai opsi pop up konfirmasi proses penghapusan sebelum menghapus data
* Di dalam if buat const res = await link.delete(‘/Database tempat data/’+id); ini ber arti menggunakan link dan function delete dari axios untuk menghapus data ber-id yg di pilih dari database mana yang akan di hapus
* Pada controller bagian function delete untuk resp json nya isi dengan array yang memuat var pesan berisi message apabila data berhasil di hapus metode nya sama seperti pada input data
* Lalu kembali pada function hapus setelah const set pesan penghapusan dengan const setPesan(res.data.pesan) menggunakan elemen const untuk menampilkan pesan yang sudah di deklarasikan sebelumnya.

14. Memahami mengapa proses mempengaruhi data baik delete/input/ubah menggunakan async dan await karena agar semua data yang dibutuhkan untuk proses tersebut termuat semuanya terlebih dahulu baru proses dari function dijalankan.

15. Memahami & dapat melakukan proses select/ambil data untuk nantinya akan di ubah :

* Menambahkan button ubah pada tabel dengan onClick menjalankan function showData()
* Buat async function showData dengan params(id)
* Buat const res = await link.get(‘/Database/’+id); karena menggunakan async maka menunggu data termuat semua terlebih dahulu baru proses akan di jalankan

16. Memahami & dapat melakukan proses peng-input an data yg akan di ubah ke ke form :

* Pada const untuk useForm tambahkan setValue sebuah function untuk mengisikan value/nilai
* Lalu pada async function showData gunakan setValue untuk mengisi form dari data yang di pilih untuk di hapus tadi setValue(‘Tabel database’, res.data[0].kolom nya);
* Lalu gunakan setIdkategori(res.data[0].idkategori); untuk menge-set/menyimpan sementara datanya

17. Memahami & dapat melakukan proses update data :

* Tambahkan const state pilihan dan setPilihan untuk useState bernilai true
* Tambahkan setPilihan bernilai false pada showData agar data yang di select bernilai netral sebelum dilakukan pengubahan oleh function
* Pada function simpan buat if kondisi pilihan lalu pindahkan link.post untuk input data ke dalam if
* Sedangkan untuk else buat lin.put seperti pada axios untuk proses update data juga atur untuk setPesan lalu tambahkan setPilihan(true) agar proses update data tidak tertukar dengan input data
* Pada controller update data buat pesan update datanya dalam bentuk array seperti sebelum2 nya.
* Proses update dan delete data menjadi lebih efisien karena hanya perlu satu form namun dengan 2 kondisi yang berbeda.