|  |  |
| --- | --- |
| **Materi** | **Nilai** |
| Project Restoran Bag 4 (Vid 47 – 54) | 90 |

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

1. Memahami bahwa untuk mendapatkan data untuk detail order yaitu dengan join kan tabel untuk menampilkan data. Tabel2 yang di join kan di antara nya yaitu tabel orders,menus,pelanggans,kategoris

2. Dapat men-join/membentuk relasi kan tabel order dengan berbagai tabel database untuk detail order :

* Menggunakan syntax untuk join table dan isi main table dengan ‘details’
* Menambahkan join untuk tabel apa saja yang akan di relasikan misal: ->join('orders', 'details.idorder', '=', 'orders.idorder')
* Tambahkan select untuk data apa saja yang akan digunakan/ditampilkan ->select('orders.\*', 'details.\*', 'menus.\*', 'pelanggans.\*', 'kategoris.\*') jika menggunakan bintang seperti itu makshudnya adalah data yang di inginkan akan di tampilkan semua nya tanpa terkecuali
* Tambahkan orderBy untuk mengurutkan data berdasarkan apa misal: ->orderBy('orders.status', 'asc') disini data di urutkan berdasarkan status yang diambil dari tabel order
* Tambahkan var $a dan $b pada params function show
* Buat where untuk menampilkan tglorder dari mana dengan var $a hingga mana dgn var $b ->where('tglorder', '>=', $a) ->where('tglorder', '<=', $b)
* Gunakan function ->get() untuk mendapatkan data

3. Dapat menampilkan data detail order ke page back end detail order :

* Pada router menambahkan get untuk detail dunction show
* Pada folder react buat Deatil.js untuk tampilan data nya
* Pada file content import untuk detail.js dan tambahkan if untuk Detail
* Gunakan rfc untuk membuat kerangka const sekaligus import react sebagai prompt tampilan nya
* Import useState from react, useForm from ‘react-hook-from’
* Di dalam const utama tambahkan const [isi] seperti sebelumnya ubah place menjadi Detail karena untuk menampilkan data dari Detail lalu copas juga const awak ahir untuk set tanggal data yang akan di tampilkan.
* Copy juga let today karena untuk mengeset waktu agar sesuai dengan yang ada pada komputer
* Buat let no = 1 untuk looping nomor table tempat menampilkan data nantinya
* Di dalam return buat row2 untuk kerapihan layout page dan isi juga dengantag div utama
* Isi row pertama dengan judul Data yang akan ditampilkan
* Untuk row kedua isi dengan table dimana table thead diisi dengan judul dari masing2 kolom dan kolom tbody isi dengan isi.map((val,index) => ( untuk menampilkan data dari database menggunakna variable val <td>{val.menu}</td> tinggal menyesuaikan dengan data apa saja yang akan ditampilkan
* Untuk menampilkan nomor tinggal <td>{no++}</td>
* Untuk total pembelian buat hargajual dikalikan jumlah pembelian <td>{val.jumlah \* val.hargajual}</td>

4. Dapat menambahkan form untuk mencari/filter order sesuai tanggal :

* Copy form untuk pencarian tanggal seperti pada order
* Copy const [register,handleSubmit] untuk useForm karena diperlukan untuk proses submit untuk menjalankan proses filter data
* Copy function cari yaitu untuk menjalankan setAwal dan setAkhir gunan penetapan tanggal berapa sampai berapa data yang akan ditampilkan
* Function cari nya sudah ter definisi pada form

5. Memahami & dapat melakukan proses pengambilan data user untuk ditampilkan di web page:

* Pada LoginController buat function index untuk menampilkan user yang memiliki level bukan pelanggan $data = User::where('level', '<>', 'pelanggan')->get();
* Atur router get untuk menampilkan datanya
* Pada react buat file User lalu impor pada contet juga add agar tampil di web page
* Pada file user isi dengan judul user admin
* Buat user baru pada postman dengan 3 klasifikasi kelas yaitu koki,kasir,admin

6. Memahami & dapat menampilkan data user yang sudah di ambil sebelumnya pada web page :

* Impor hook useGet karena akan digunakan untuk mengambil data user dengan const [isi] = useGet('/User/');
* Buat div class row dan div untuk judul, tambah data user dan table penampil user
* Isi row judul lalu isi row untuk tambah data dengan button untuk tambah data
* Isi row table dengan map seperti sebelumnya lalu tampilkan data email,level dan status
* Buat variable untuk nomer dan gunakan metode increment untuk looping nomer no++

7. Memahami & dapat menambahkan opsi untuk tambah data user :

* Impor modal karena akan digunakan untuk menempatkan form tambah data user
* Impor useState karena akan digunakan untuk const mopen,setMopen untuk membuka dan menutup akses modal
* Impor useForm untuk menggunakan const *register* sebagai pendaftaran data,*handleSubmit* untuk submit data,*errors* untuk menampilkan pesan error saat input data dan *reset* untuk mereset proses setelah data berhasil di tambahkan
* Impor juga link axios karena akan berperan untuk proses mengirim data ke api lumen
* Tambahkan const untuk useForm seperti sebelumnya dan const untuk pesan,setpesaan menggunakan useState
* Pada row tambah data arahkan input menggunakan event onClick={()=?(tambah())} untuk mengarahkan ke function tambah.
* Buat function tambah yg berisi untuk membuka modal yaitu setMopen(true);
* Buat modal lalu arahkan untuk membuka dengan isOpen={mopen} dari useState
* Isi dengan event onRequestClose={() => setMopen(false)} ini merupakan perintah jika ingin menutup modal
* Copy style modal seperti pada pembayaran order
* Tambahkan row dan div untuk judul juga untuk form input data
* Pada form input data isi dengan email,password,select option dan submit tidak lupa ref={register} karena untuk menggunakan ketentuan function register yg ada pada LoginController dan disalurkan lewat useForm
* Tambahkan error message menggunakan komponen dari useForm lalu tinggal mengatur pesan nya misal {errors.email && "Email Harus Di Isi!"}
* Pada tag form arahkan dengan event ke function simpan onSubmit={handleSubmit(simpan)} lalu buat function simpan yang akan menyimpan data ke database
* Buat async function simpan dengan params data lalu buat objek dan isi dengan email.data.email untuk password,level dan khusu relasi isi dengan ‘back’ karena ini merupakan bagian dari sistem backend restoran
* Diluar objek buat const res = await link.post('./register', user).then(res => setPesan(res.data.pesan)); artinya menggunakan async function untuk mengirim data ke register setelah data termuat semua. Lalu tambahkan setMopen(false) untuk menutup modal saat proses input data selesai
* Pada LoginController di bagian function register untuk resp nya buat array untuk mengirimkan pesan ‘data berhasil di tambahkan’ dan di tampilkan dengan {pesan} pada webpage //bisa ditampilkan karena sudah di tambahkan dengan const useState lalu pada res setelah proses input data juga.

8. Memahami dan dapat mengubah button active/banned pada user :

* Pada kolom status buat if status === 1 lalu isi dengan input untuk active dan else nya input untuk Banned lalu arahkan ke function status(val.id) mengambil data id dari data yg button nya di klik dengan event onClick={() => status(val.id)}
* Buat async function status dengan params id karena fokusnya dari data id yang diambil dari button yg di klik
* Gunakan filter untuk menyaring hanya id status saja yang akan digunakan const data = isi.filter((val) => val.id === id);
* Buat variable untuk nilai awal dari status let stat = 0;
* Buat if untuk menguji apabila data bernilai 1/aktif maka status akan menjadi 0/banned if (data[0].status === 1) { stat = 0;} else {stat = 1;}
* Buat objek untuk menyandingkan data statu agar sama dengan variable stat let kirim ={status:stat}
* Kirim data untuk melakukan update const res = await link.put('/User/' + id, kirim);
* Pada LoginController buat function untuk update seperti pada kategori dengan menggunakan request dan modal User dan kunci utama id=$id
* Pada router tambahkan akses untuk put yaitu untuk update data

9. Memahami & dapat membuat path login untuk user Bagian 1:

* Pada App.js menambahkan route untuk login menggunakan function Login yang sudah dibuat sebelumnya di LoginController <Route path="/login" component={Login} />
* Buat file untuk page Login
* Import useForm dan link Axios pada Login
* Tambahkan register,handleSubmit,reset dengan useForm
* Buat form untuk Login dengan template form bootstrap dan ubah sesuai kebutuhan login
* Atur juga elemen css nya agar lebih rapi
* Tambahkan ref register di input email dan password agar datanya tersimpan ref={register({ required: true })}
* Arahkan form ke function login untuk login dengan menggunakan event onSubmit={handleSubmit(login)}
* Buat function login dengan params data lalu kirim data menggunakan link const res = await link.post('/login', data); dan tambahkan reset() agar form ter reset setelah berhasil Login
* Pada function login di LoginController tambahkan pengujian agar saat sebuah data tidak terdaftar maka akan tidak bisa login if(isset($usr)){ isi dengan script login}
* Di dalam script login buat pengujian juga untuk data user yang status nya bukan 1/aktif maka tidak akan bisa login if ($user->status === 1)