|  |  |
| --- | --- |
| Materi | Nilai |
| Pembelajaran React-JS Video ke 39 - 46  (Tugas semester 4 #11) | 95 |

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

1. Membuat tabel order dan detail menggunakan terminal dan perintah php artisan make:model –mc - -resource. Yaitu membuat model untuk tabel order dan tabel detail serta membuat file migration dan file controller untuk tabel order dan tabel detail. - -resource digunakan agar file model, migration, dan controller yang sudah dibuat otomatis diisi function functionnya.
2. Melakukan join table pada table pelanggans dan orders.

Cth :

use Illuminate\Support\Facades\DB;

$data = DB::table('orders')

            ->join('pelanggans', 'pelanggans.idpelanggan', '=', 'orders.idpelanggan')

            ->select('orders.\*', 'pelanggans.\*')

            ->orderBy('orders.status', 'asc')

            ->get();

*return* response()->json($data);

Artinya, mendaftarkan file DB ke dalam file OrderController. Setelah itu, di dalam $data berisi fungsi DB::Table dengan memilih tabel orders, terdapat perintah join yang digunakan untuk memilih kolom mana yang akan dihubungkan, perintah join memilih tabel pelanggans dengan memilih kolom idpelanggan pada tabel pelanggans dan menghubungkannya dengan kolom idpelanggan pada tabel orders.

Lalu, setelah menentukan kolom yang akan dihubungkan, terdapat perintah select yang digunakan untuk memilih data yang akan ditampilkan. Data yang akan ditampilkan adalah semua data yang ada pada tabel orders dan semua data yang ada pada tabel pelanggans.

Kemudian, mengurutkan penampilan data yaitu sesuai dengan kolom status pada tabel orders dan menggunakan perintah urutan secara asc / dari urutan kecil ke yang besar.

Dan, data yang sudah dipilig akan ditampilkan menggunakan get().

1. Melakukan update data pada tabel order

Cth :

$data = [

            'bayar' => $request->input('bayar'),

            'kembali' => $request->input('kembali'),

            'status' => $request->input('status')

        ];

        $orders = Order::where('idorder', $id)->update($data);

Artinya, di dalam $data berisi array yang di dalamnya terdapat key bayar dan value mengambil nilai sesuai dengan $request dari form bayar yang diinputkan, key kembali mengambil nilai sesuai dengan $request dari form kembali yang diinputkan, dan key status mengambil nilai sesuai dengan $request dari form status yang diinputkan. Setelah itu, $orders berisi model Order yang memilih data sesuai dengan idorder yang sama pada $id, lalu melakukan perintah update dan menjalankan $data.

1. Memilih dan menampilkan data sesuai dengan tanggal yang dipilih.

Cth :

public function show($a, $b)

    {

        $orders = Order::where('tglorder', '>=', $a)

            ->where('tglorder', '<=', $b)

            ->get();

*return* response()->json($orders);

    }

$router->get('order/{a}/{b}', ['uses' => 'OrderController@show']);

Artinya, pada parameter function show berisi $a dan $b, lalu di dalam function show terdapat $orders yang berisi model Order dengan memilih data menggunakan where, dan melakukan pemilihan sesuai dengan kolom tanggal order jika lebih besar sama dengan dari $a dan melakukan pemilihan sesuai dengan kolom tanggal order jika lebih kecil sama dengan $b. Lalu, data akan ditampilkan menggunakan perintah get(). Jadi, data yang ditampilkan adalah data yang ada di antara tanggal yang dipillih.

Setelah itu, nilai akan dikeluarkan menggunakan return dan response() akan mengirimkan $orders berbentuk json.

Untuk pemanggilan, di dalam $router menjalankan perintah get() apabila diketikkan order/a/b, maka akan menjalankan function show pada OrderController.

1. Mengambil data sesuai dengan tanggal yang dipilih menggunakan form.

Cth :

const [awal, setAwal] = useState("2021-03-01");

const [akhir, setAKhir] = useState("2021-03-31");

  function cari(data) {

    setAwal(data.tawal);

    setAKhir(data.takhir);

  }

const [isi] = useGet(`/order/${awal}/${akhir}`);

Artinya, di dalam const pertama terdapat variabel awal dengan function pengisi yaitu setAwal dan isi useState adalah 2021-03-01, dan pada const kedua terdapat variabel akhir dengan function pengisi yaitu setAkhir dan isi useState adalah 2021-03-31.

Lalu, di dalam function cari dengan parameter data berisi, mengatur function setAwal dengan mengambil nilai dari data.tawal yang dikirimkan oleh form dan mengatur function setAkhir dengan mengambil nilai dari data.takhir yang dikirimkan oleh form.

Kemudian, const isi menjalankan useGet dengan url yang memanggil /order/$awal/$akhir.

Jadi, ketika form di submit, maka handlesubmit akan mengarahkan form tersebut ke dalam function cari dan data yang dikirimkan disimpan ke dalam variabel data. Lalu, melakukan pengisian function dengan mengambil nilai dari tawal untuk mengisi setAwal dan takhir untuk mengisi nilai dari setAkhir. Setelah function pengisi mengambil data dari form yang dikirimkan, maka variabel awal dan akhir akan dijalankan untuk mengambil data dan menampilkan data yang sesuai dengan tanggal yang dipilih.

1. new Date() adalah objek yang digunakan untuk mengambil tanggal dan waktu sekarang ini.
2. toISOString() digunakan untuk membuat object date dalam format standar ISO.
3. Method slice() menampilkan sebuah perintah yang dipilih, yaitu perintah start sampai end. Dimana start dan end merupakan index / penentu dari data yang akan ditampilkan dan tidak merubah array asli.
4. Menginstall react-modal yang digunakan untuk menampilkan modal.
5. Mengimportkan react-modal ke dalam file yang di dalamnya akan menggunakan modal.

Cth :

*import* Modal *from* "react-modal";

1. Untuk menggunakan react-modal, maka di dalam file diatur setAppelement untuk modal.

Cth :

Modal.setAppElement("#root"); //#root sama seperti (‘root’) di dalam file index.js

1. isOpen={} adalah even yang memuat perintah ketika modal dibuka.
2. onRequestClose={} adalah even yang berisi perintah untuk menutup modal ketika mengklik bagian luar modal. Di dalam even ini harus menggunakan function.
3. Untuk memepercantik tampilan, di dalam <Modal /> bisa menggunakan style={}.
4. Contoh penggunaan modal :

*import* Modal *from* "react-modal";

Modal.setAppElement("#root");

const [mopen, setMopen] = useState(false);

<Modal

        isOpen={mopen}

        onRequestClose={() => setMopen(false)}

        style={{

          overlay: {

            background: "grey",

          },

          content: {

            color: "navy",

            borderRadius: "6px",

          },

        }}

      >

        <h4>Pembayaran</h4>

        <p>Isi Pembayaran</p>

<button className="btn btn-info btn-md" onClick={() => setMopen(false)}>

          Close

        </button>

      </Modal>

        <button className="btn btn-info btn-md" onClick={() => setMopen(false)}>

          Close

        </button>

      </Modal>

<button

           className="btn btn-danger btn-md"

            onClick={() => setMopen(true)}

             >

             Belum Bayar

             </button>

Artinya, mengimportkan Modal dari react-modal, lalu mengatur setAppElement dengan (“#root”). Const dengan variabel mopen dan function pengisi setMopen dengan nilai awal adalah false.

Di code bawah, terdapat button yang menjalankan event onClick, jadi ketika button Belum Bayar di click, maka function setMopen menjadi bernilai true.

Di dalam <Modal />, terdapat even yang menjalankan mopen dengan pengaturan yang sudah dilakukan pada button Belum Bayar. Jadi ketika button belum bayar di klik, maka akan modal dengan function setMopen bernilai true sehingga modal bisa ditampilkan.

Lalu, setelah modal ditampilkan, terdapat event onRequestClose dengan function di dalamnya yaitu mengatur setMopen menjadi bernilai false, jadi ketika bagian luar di klik, maka nilai dari setMopen akan menjadi false, sehingga tampilan modal akan tertutup.

Terdapat style={} yang digunakan untuk mengatur tampilan, bagian overlay, adalah bagian yang terdapat di luar modal, sementara bagian content adalah bagian dalam modal.

Terakhir, ada button Close yang jika di klik, maka akan menjalankan event onClick dimana di dalamnya terdapat function setMopen menjadi false. Jadi, ketika button close di klik, maka nilai dari setMopen akan menjadi false dan modal akan ditutup.

1. onAfterOpen{} adalah event yang digunakan untuk menjalankan suatu function ketika <Modal /> dibuka.
2. Menggunakan form di dalam modal :

const [total, setTotal] = useState(0);

const [pelanggan, setPelanggan] = useState("");

  function filterData(id) {

    const data = isi.filter((val) => val.idorder === id);

    setPelanggan(data[0].pelanggan);

    setTotal(data[0].total);

    setMopen(true);

  }

  function isiForm() {

    setValue("total", total);

  }

<Modal />

onAfterOpen={isiForm}

Artinya, terdapat const dengan variabel total dan function pengisi variabel adalah setTotal dan terdapat const dengan variabel pelanggan dan function pengisi variabel adalah setPelanggan. Karena setelah button di klik, maka akan diarahkan untuk menjalankan function filterData dimana di dalam nya sudah diatur parameter dari filterData ini diisi dengan mengambil nilai dari val.idorder. Dan di dalam function filterData, berisi parameter id, lalu function nya berisi const data yang mengambil nilai dari const isi dengan memfilter data yang diambil yaitu dengan variabel penampung val, dan mengambil nilai dari val.idorder apakah sama dengan id.

Setelah itu, mengatur function pengisi setPelanggan dengan mengambil data pelanggan dari variabel data pada index ke 0. Lalu, mengatur function pengisi setTotal dengan mengambil nilai data total dari variabel data pada index ke 0. Dan mengatur function pengisi setMopen yang nilainya menjadi true.

Function isiForm, digunakan untuk mengisi value dari form, karena di Modal tidak bisa langsung dipanggil, tetapi harus diatur terlebih dahulu di dalam function. Di dalam function isiForm terdapat function pengisi setValue dengan mengambil name dari form yang akan diisi yaitu total, dan mengisi nya dengan mengambil nilai dari variabel total.

Lalu, di dalam Modal, terdapat event onAfterOpen yang menjalankan function isiForm.

1. Ketika ingin mengupdate data tanpa ada file di dalamnya, maka update menggunakan perintah put. Tetapi, ketika terdapat file di dalamnya, maka update dilakukan menggunakan perintah post dengan mengirimkan data menggunakan formData().
2. Mengirimkan data pembayaran dan melakukan update ketika pembayaran yang dilakukan sudah lunas.

Cth :

async function simpan(data) {

    let hasil = {

      bayar: data.bayar,

      kembali: data.bayar - data.total,

      status: 1,

    };

    const res = *await* link.put("/order/" + idorder, hasil);

    setMopen(false);

  }

Artinya, terdapat async function simpan yang di dalam nya terdapat variabel data yang diisi dari data – data yang dikirimkan dari dalam form. Lalu, terdapat variabel hasil yang berisi objek, bayar dengan mengambil data dari bayar yang ada pada variabel data, kembali mengambil nilai pengurangan dari bayar pada variabel data dengan total pada variabel data, dan status dirubah menjadi 1. Lalu const res berisi link yang disambungkan dengan put untuk perintah update dengan menjalankan /order/idorder dan mengirimkan variabel hasil.

Setelah berhasil untuk update, maka function pengisi setMopen akan diatur menjadi false, jadi otomatis modal akan ditutup.

**Saya Belum Mengerti**