BAB V  
IMPLEMENTASI

Implementasi akan merelealisasikan desain sistem yang telah kita buat ke dalam wujud nyata. Pada bab ini akan banyak menjelaskan mengenai segmen program yang dijalankan pada masing – masing fitur. Selain itu beberapa segmen program yang mungkin tidak masuk ke dalam fitur karena sifatnya yang terlalu umum namun penting akan dijelaskan. Bagian ini akan menjelaskan bagaimana sekiranya code yang telah dibuat di dalam sistem bekerja. Semua detail terkait program yang dibuat akan dijelaskan.

1. General

General akan menjelaskan beberapa hal yang sekiranya perlu diatur terlebih dahulu. Disini banyak hal terkait settingan program ditulis. Program ini akan berjalan pada hampir di setiap halaman program. Namun perlu diingat program ini tidak akan jalan ketika *code* dari program yang dibuat tidak dipanggil.

* + 1. Koneksi Database

Koneksi diperlukan untuk membuat program yang dibuat dalam melakukan akses ke database. Hal ini harus dilakukan dikarenakan program database dengan yang dibuat terpisah. Ketika kita sudah membuat koneksi, beberapa segmen yang lainnya dapat dengan mudah melakukan *insert*,*update*,dan delete data ke database. Bisa dikatakan koneksi database ini adalah sebuah helper yang membantu programmer dalam memudahkan hal – hal yang berhubungan dengan database.

Segmen Program 5.1 Koneksi

1. 'connections' => [
2. 'mongodb' => [
3. 'driver' => 'mongodb',
4. 'host' => env('DB\_HOST', '127.0.0.1'),
5. 'port' => env('DB\_PORT', 27017),
6. 'database' => env('DB\_DATABASE', 'homestead'),
7. 'username' => env('DB\_USERNAME', 'homestead'),

Segmen Program 5.1 (Lanjutan)

1. 'password' => env('DB\_PASSWORD', 'secret'),
2. 'options' => [
3. 'database'=>env('DB\_AUTHENTICATION\_DATABASE' ,'admin'),
4. ],
5. ],
6. ],

Line 1 hingga ke 13 adalah segmen code untuk sebuah koneksi. Untuk mendefinisikan sebuah koneksi line 2 hinnga 12 diperlukan. Nantinya koneksi ke dua akan diletakkan di dalam kurung ‘*connection*’. Line 2 adalah nama koneksinya. Line 3 adalah driver yang digunakan bisa mysql, sqlserver, atau mongodb dalam hal ini. Line 4 adalah untuk memberikan koneksi ini di host di mana, jika tidak di setting di *environment* maka defaultnya adalah nembak ke *localhost*. Line 5 untuk memberikan port yang digunakan. Line 6 adalah nama database yang digunakan. Line 7 hingga 8 adalah untuk memberikan username dan password saat mau mengakses database. Line 10 adalah untuk autentikasi databasenya.

* + 1. Authentikasi dan persist data login

Autentikasi dipastikan digunakan untuk memastikan user yang login saat ini. Data user yang berhasil login harus disimpan ke dalam sebuah cookie sehingga user ketika berpindah halaman atau tidak sengaja menutup browsernya tetap login. Hal ini digunakan untuk memberikan *user experience* yang lebih baik kepada user. Untuk memastikan tidak terjadi perulangan ke menu login. Penyimpanan pada program ini menggunakan library Pinia, untuk melakukan hal yang telah disebutkan diatas.

Segmen Program 5.2 Auth Store Pinia

1. export const useAuth = defineStore("auth", {
2. state: () => ({
3. authuser: null,
4. authErrors: []
5. }),
6. getters: {
7. user: (state) => state.authuser,
8. errors: (state) => state.authErrors
9. },
10. actions: {

Segmen Program 5.2 (Lanjutan)

1. async getUser() {
2. this.authuser = this.authuser
3. },
4. async handleLogin(data) {
5. this.authErrors = [];
6. try {
7. await axios.post('/api/login', data).then((response) => {
8. if (response.data[0] == null) {
9. this.authuser = null
10. }
11. else {
12. this.authuser = response.data[0].user
13. }
14. });
15. } catch (error) {}
16. },
17. async handleLogout() {
18. try {
19. this.authuser = null;
20. } catch (error) {
21. }
22. },
23. },
24. persist: true
25. })

Line 1 adalah nama storenya. Line 2 hingga line 5 menjelaskan model data dari store yang dibuat. Line 6 hingga 8 adalah *getter* seperti pada java yang ditujukan untuk mengambil data. Line 11 hingga 33 adalah kumpulan aksi atau perintah yang dapat dilakukan pada store ini. Line 11 hingga 13 berfungsi untuk mendapatkan user sekarang. Line 14 hingga 26 berfungsi adalah fungsi untuk melakukan login ke sistem. Line 27 hingga 33 berfungsi untuk proses logout user. Line 34 adalah untuk membuat data user yang login sekarang tetap tersimpan selama user belum menekan tombol logout.

* + 1. Tarik Data Database Perusahaan (SQL Server)

Perbedaan database mongodb dengan yang perusahaan miliki sekarang, mengharuskan dilakukan penarikan data. Saat ini perusahaan menggunakan database berbasis SQL yang bernama Microsoft SQL Server. Oleh karena itu penarikan data ini tidak bisa langsung semerta merta dipakai. Pada tahapan ini terjadi beberapa tahapan yaitu proses menarik semua data menggunakan SQL query kemudian melakukan konversi ke json untuk disimpan ke mongodb.

Segmen Program 5.3 Tarik Data SPK

1. $datas=DB::connection('sqlsrv')-> table('SURATPERINTAHKERJA')
2. ->join('SPECIFICATION', 'SPECIFICATION.SPK Number', '=', 'SURATPERINTAHKERJA.SPK Number')
3. ->get();
4. foreach ($datas as $data) {
5. $newdata = SPK::create([
6. "NOSPK" => trim($data->{'SPK Number'}),
7. "Tipe"=>trim($data->{'SPK Type'}),
8. "AirSuspensi"=>trim($data->{'Air Suspensi'}),
9. "Semi\_Monocoque"=>trim($data->{'Semi Monocoque'}),
10. "No\_Rangka"=>trim($data->{'No Rangka'}),
11. "No\_Mesin"=>trim($data->{'No Mesin'}),
12. "parameter" => [
13. "ModelMobil" => trim($data->{'Merk'}),
14. "TipeMobil" => trim($data->{'Type'}),
15. "TinggiMobil" => "",
16. "newparameter" => [
17. [
18. "Newparam" => trim($data->{'User Defined'}),
19. "Component" => [trim($data->{'User Defined Desc'})],
20. ],
21. ]
22. ]
23. ]);
24. }

Line 1 hingga 3 adalah perintah untuk mendapatkan semua data SPK milik perusahaan. Baris 4 hingga 24 adalah untuk proses melakukan looping baris data yang didapat. Setelah didatpatkan dibuatlah Model dalam hal ini SPK untuk proses insert data dimulai dari line 5 hingga 23. Segmen program ini memiliki cara yang sama dengan pengecekkan data yang sudah ada di database master milik mongodb. Perbedaannya hanya terletak pada jumlah banyak if untuk melakukan proses pengecekkan data.

* + 1. Routing

Routing adalah kumpulan rute pada website yang mana akan menjadi tempat path website didefinisikan. Karena penggunaan vue sebagai frontendnya maka route juga harus menggunakan cara vue. Disini diperlukan sebuah library bernama vue-router untuk melakukan routing. Pada bagian ini juga middleware akan didefinisikan untuk membatasi akses web user supaya tidak dapat mengakses path yang bukan haknya.

Segmen Program 5.4 Routing

1. import Vue from 'vue';
2. import VueRouter from 'vue-router'
3. import { useAuth } from '../../Stores/Auth'
4. Vue.use(VueRouter)
5. const routes = [{
6. name: 'Home',
7. path: '/',
8. component:()=> import("../components/Pages/Home/Home.vue"),
9. meta: {
10. guestPageAccess: true,
11. }
12. },
13. {
14. name: 'Register', // register user baru
15. path: '/user/create',
16. component:()=>import("../components/Pages/Register/Register.vue"),
17. meta: {
18. guestPageAccess: false,
19. levelAccess: 'Super Admin Role'
20. }
21. },
22. ]

Line 1 hingga 4 adalah inisialisasi untuk mengimpor library. Line 5 hingga 22 adalah tempat *route* ditaruh. Line 5 hingga 12 adalah contoh pembuatan sebuah route begitu juga line 13 hingga 21. Baris 6 adalah untuk memberikan alias path sehingga dapat dengan mudah dipanggil. Baris 7 adalah *pathnya*. Baris 8 adalah tempat untuk mengimpor komponen yang akan diload pada halaman *path*.baris 11 hingga 11 adalah akses level yang akan berhubungan dengan middleware pada segmen program 5.5. Meta adalah fungsi yang sifatnya opsional ketika tidak di declare maka semua user yang belum login bisa menagkses path yang bersangkutan.

Segmen Program 5.5 Middleware

1. router.beforeEach((to, from, next) => {
2. const store = useAuth();

Segmen Program 5.5 (Lanjutan)

1. if (!to.meta.guestPageAccess) {
2. store.getUser();
3. const routes = [
4. if (store.user) {
5. if (store.user.account\_privileges.title == "Super Admin Role") {
6. next()
7. }
8. else if (store.user.account\_privileges.title == "Admin Role") {
9. if (to.meta.levelAccess == "Admin Role" || to.meta.levelAccess == "Staff Role") {
10. next()
11. }
12. }
13. else if (store.user.account\_privileges.title == "Staff Role") {
14. if (to.meta.levelAccess == "Staff Role") {
15. next()
16. }
17. }else{
18. next({
19. name: 'NotFound'
20. })
21. }
22. }
23. else {
24. next({
25. name: 'Login'
26. })}
27. }
28. else {
29. next()
30. }
31. })

Line 1 adalah mendefinisikan fungsi middleware. Line 2 adalah untuk mengakses store yang ada di pinia. Line 3 hingga 33 adalah untuk mendeteksi apakah user sedang login atau tidak. Line 4 berfungsi untuk mendapatkan role dari user. Line 5 berfungsi untuk memberikan rute yang dapat dipilih oleh user berdasarkan rolenya. Line 6 hingga 9 adalah fungsi untuk mengatur *path* user dengan role super admin. Line 10 sampai 14 adalah fungsi untuk mengatur *path* user yang memiliki role admin. Line 15 hingga 24 adalah untuk mengatur path user dengan role staff. Line 25 hingga 29 adalah untuk fungsi untuk melakukan *redirect* ke path login ketika user mengakses path yang bukan rolenya. Dan line 30 hingga 32 adalah untuk mengatur path user yang belum login.

* + 1. Helper Konversi Waktu

Konversi waktu diperlukan untuk membuat format tanggal dan waktu sesuai dengan keinginan. Di mongodb secara default penyimpanan yang berhubungan dengan datetime ditulis dalam bentuk model data UTC miliseconds sejak epoch. Untuk menghindari proses melakukan penulisan code yang berulang dalam melakukan konversi datetime dibuatlah sebuah fungsi global yang membantu.

Segmen Program 5.6 Konversi Format Waktu

1. import { format, parseISO } from "date-fns"
2. import { id } from 'date-fns/locale';
3. export default {
4. methods: {
5. converttime(date) {
6. const str = format(
7. new Date(date),
8. 'dd-MMMM-yyyy HH:mm:ss', { locale: id }
9. );
10. return str;
11. },
12. }
13. };

Line 1 dan 2 adalah proses mengimpor library yang diperlukan. Line 3 adalah proses mendaklarasikan agar segmen program ini dapat dipakai secara global. Selanjutnya line 4 adalah tempat meletakkan methodnya. Line 5 sampai 11 adalah satu kesatuan fungsi. Dimana pada line 6 hingga 9 dilakukan proses konversi data ke datetime local indonesia.

1. Master

Untuk melakukan fitur konversi nantinya, fitur master harus dilakukan terlebih dahulu. Disini master sesuai yang telah di sebutkan pada pembagian role sebelumnya, pengirisan master hanya dapat dilakukan oleh dua role yaitu Super admin dan admin. Proses penambahan master disini memiliki 2 macam form yang terbagi di sebelah kiri dan kanan untuk itu dalam melakukan proses input kedua form harus diisi. Database master dapat di tambah , edit, dan dihapus (*soft delete*) seperti data pada umumnya. Untuk edit dan hapus tidak dibahas pada segmen program nantinya dikarenakan fungsi edit memiliki kesamaan yang cukup banyak dengan fungsi tambah, perbedaannya hanya terletak pada prosedur langkah awal, yang mana data yang mau di update harus dicari terlebih dahulu kemudian di tampilkan. Lalu data akan diubah seperti pengecekkan validasi master sebelum memasukkan data. Untuk fungsi delete juga tidak dibahas karena programnya yang sangat singkat, yaitu mencari id dari data yang mau dihapus kemudian menulis sebaris program untuk delete

Segmen Program 5.7 Format Input

1. $param = $request->dataparam;
2. $kit = $request->datakit;
3. $i = 0;
4. foreach ($param['TipeMobil'] as $item) {
5. $param['TipeMobil'][$i] = ucwords($item);
6. $i++;
7. }
8. $i = 0;
9. foreach ($param['ModelMobil'] as $item) {
10. $param['ModelMobil'][$i] = ucwords($item);
11. $i++;
12. }
13. $i = 0;
14. foreach ($param['TinggiMobil'] as $item) {
15. $param['TinggiMobil'][$i] = ucwords($item);
16. $i++;
17. }

Segmen ini berfungsi untuk mengubah format input yang dilakukan oleh user. Fungsinya adalah membuat setiap kata pada huruf pertama selalu berhuruf besar diikuti oleh huruf kecil pada kata tersebut.pada baris 1 dan 2 berfungsi untuk mengambil inputan user. Pada baris 3 hingga 7 adalah proses melakukan looping untuk mengganti setiap field inputan tipe mobil. Baris 8 hingga 12 berperan untuk mengganti format di field model mobil secara berulang . dan pada baris 13 hingga 17 berperan untuk megganti format pada field tinggi mobil.

Segmen Program 5.8 Cek Inputan Kosong

1. function FungsicekKosong(array $cek)
2. {
3. if (count($cek) == 0) {
4. return true;
5. }
6. }

Segmen Program 5.8 (Lanjutan)

1. $paramkosong = false;
2. if (!$paramkosong) {
3. $paramkosong = FungsicekKosong($param['TipeMobil']);
4. }
5. if (!$paramkosong) {
6. $paramkosong = FungsicekKosong($param['ModelMobil']);
7. }
8. if (!$paramkosong) {
9. $paramkosong = FungsicekKosong($param['TinggiMobil']);
10. }
11. if (!$paramkosong) {
12. $paramkosong = FungsicekKosong($param['Departemen']);
13. }
14. if (!$paramkosong) {
15. $paramkosong = FungsicekKosong($param['Stall']);
16. }
17. if ($paramkosong) {
18. return response()->json([
19. "success" => true,
20. "statuscode" => 401,
21. ]);
22. }
23. $paramtambahankosong = false;
24. foreach ($param['NewParameter'] as $newparam) {
25. if ($newparam['Newparam'] == "" || $newparam['Newparam'] == null) {
26. $paramtambahankosong = true;
27. break;
28. }
29. foreach ($newparam['Component'] as $komponen) {
30. if ($komponen == "" || $komponen == null) {
31. $paramtambahankosong = true;
32. break;
33. }
34. }
35. if ($paramtambahankosong) {
36. break;
37. }
38. }
39. if ($paramtambahankosong) {
40. return response()->json([
41. "success" => true,
42. "statuscode" => 403,
43. ]);

Line 1 hingga 6 berfungsi untuk membuat fungsi kecil dalam melakukan proses untuk mengecek apakah terdapat inputan yang kosong. Nantinya hal ini akan digunakan secara berulang untuk mengecek beberapa parameter. Line 18 adalah penanda untuk memberikan tanda apakah nantinya inputan yang diberikan oleh user ada yang kosong atau terisi semua. Line 19 hingga 21 mengecek parameter tipe mobil terisi lengkap atau tidak. Line 22 hingga 24 mengecek parameter model mobil terisi lengkap atau tidak. Line 25 hingga 27 mengecek parameter tinggi mobil terisi lengkap atau tidak. Line 28 hingga 30 mengecek parameter departemen terisi lengkap atau tidak. Line 31 hingga 33 mengecek parameter stall terisi lengkap atau tidak. Line 34 hingga 39 berfungsi untuk memberikan response gagal jika ternyata terdapat parameter yang kosong.

Untuk pengisian pada parameter baru pengecekkan dilakukan dimulai dari line 40 hingga 60. Line 42 berfungsi untuk melakukan looping pada area inoputan new parameter. Line 42 hingga 45 adalah mengecek apakah nama dari parameter yang baru kosong atau tidak diisi. Line 46 hingga 49 adalah fungsi tambahan untuk pengecekkan isi dari parameter yang baru. Line 52 hingga 54 berfungsi untuk menghentikan proses looping ketika ternyata inputan dari parameter baru ada yang kosong. Line 56 hingga 60 berfungsi untuk mengembalikan response gagal.

Segmen Program 5.9 Fungsi Inputan kembar

1. function fungsiceksama(array $cek)
2. {
3. if (count($cek) !== count(array\_unique($cek))) {
4. return true;
5. }
6. }
7. $paramsama = false;
8. if (!$paramsama) {
9. $paramsama = fungsiceksama($param['TipeMobil']);
10. }
11. if (!$paramsama) {
12. $paramsama = fungsiceksama($param['ModelMobil']);
13. }
14. if (!$paramsama) {
15. $paramsama = fungsiceksama($param['TinggiMobil']);
16. }
17. if (!$paramsama) {
18. $paramsama = fungsiceksama($param['Departemen']);
19. }
20. if (!$paramsama) {
21. $paramsama = fungsiceksama($param['Stall']);
22. }
23. if (!$paramsama) {
24. $paramsama = fungsiceksama($param['Stock']);
25. }
26. if ($paramsama) {
27. return response()->json([
28. "success" => true,
29. "statuscode" => 402,
30. ]);

Segmen Program 5.9 (Lanjutan)

1. }
2. $paramtambahansama = false;
3. $judulparamtambahankembar = array();
4. foreach ($param['NewParameter'] as $newparam) {
5. array\_push($judulparamtambahankembar, $newparam['Newparam']);
6. if(count($newparam['Component'])!==count(array\_unique( $newparam['Component']))) {
7. $paramtambahansama = true;
8. break;
9. }
10. }
11. if(count($judulparamtambahankembar)!== count(array\_unique($judulparamtambahankembar))) {
12. $paramtambahansama = true;
13. }
14. if ($paramtambahansama) {
15. return response()->json([
16. "success" => true,
17. "statuscode" => 404,
18. ]);
19. }
20. function fungsicekparameterterdaftar(array $array1, array $array2){
21. $jumlahkesamaan = 0;
22. foreach ($array2 as $isiarray2) {
23. foreach ($array1 as $isiarray1) {
24. if(strtoupper($isiarray1)==strtoupper($isiarray2)){
25. $jumlahkesamaan++;
26. break;
27. }
28. }
29. }
30. if ($jumlahkesamaan == count($array2)) {
31. return true;
32. }
33. }

Line 1 sampai 6 berfungsi untuk membuat sebuah fungsi untuk mengecek apakah terdapat inputan yang kembar. Line 7 untuk memberikan tanda nantinya apakah terdapat inputan yang kembar. Line 8 hingga 10 berfungsi untuk mengecek inputan kembar pada parameter tipe mobil. Line 11 sampai 13 mengecek inputan kembar pada parameter model mobil. Line 14 sampai 16 mengecek inputan kembar pada parameter tinggi mobil. Line 17 sampai 19 mengecek inputan kembar pada parameter departemen. Line 20 sampai 22 mengecek inputan kembar pada parameter stall. Line 23 sampai 25 mengecek inputan kembar pada stock. Line 26 sampai 31 berfungsi untuk mengembalikan response ke user.

Line 32 berfungsi untuk penanda apakah terdapat inputan kembar pada parameter baru. Line 33 merupakan inisialisasi awal untuk membuat array kosong. Line 34 sampai 63 berfungsi untuk melakukan looping, mengecek setiap inputan pada parameter baru apakah ada inputan yang kembar. Pengecekkan kembar yang dilakukan adalah nama parameter yang baru dan isi dari parameter yang baru. dan line 44 hingga 49 apabila ditemukan kesamaan response gagal langsung diberikan.

Segmen Program 5.10 Pengecekkan Data Input Dengan Data Master

1. $allmaster = Master::all();
2. foreach ($allmaster as $master) {
3. $saved = $master->Parameter;
4. $cekTipeMobil = false;
5. $cekModelMobil = false;
6. $cekTinggiMobil = false;
7. $cekDepartemen = false;
8. $cekstall = false;
9. $cekStock = false;
10. $cekAdditionaParameter = false;
11. $cekTipeMobil=fungsicekparameterterdaftar($saved['TipeMobil'], $param['TipeMobil']);
12. $cekModelMobil=fungsicekparameterterdaftar($saved['ModelMobil'], $param['ModelMobil']);
13. $cekTinggiMobil=fungsicekparameterterdaftar($saved['TinggiMobil'], $param['TinggiMobil']);
14. $cekDepartemen=fungsicekparameterterdaftar($saved['Departemen'], $param['Departemen']);
15. $cekstall=fungsicekparameterterdaftar($saved['Stall'],$param['Stall']);
16. if(count($param['Stock'])==0&&count($saved['Stock']) == 0) {
17. $cekStock = true;
18. }else{
19. $cekStock=fungsicekparameterterdaftar($saved['Stock'], $param['Stock']);
20. }
21. if(count($param['NewParameter'])==0&&count($saved['NewParameter']) == 0){
22. $cekAdditionaParameter = true;
23. }
24. if ($cekTipeMobil && $cekModelMobil && $cekTinggiMobil && $cekDepartemen&& $cekStock && $cekstall && $cekAdditionaParameter) {
25. return response()->json([
26. "success" => true,
27. "statuscode" => 406,
28. ]);
29. }}

Line 1 bertujuan untuk mengambil semua data yang ada pada master. Kemudian pada line 2 akan dilakukan proses looping untuk mengecek apakah data yang akan diinput sudah terdaftar apa belum. Baris 3 adalah proses mengambil parameter pada database master. Line 4 sampai 10 berfungsi sebagai penanda apakah parameter yang mau diinputkan sudah ada atau belum. Kemudian pada line 11 hingga 23 proses pengecekkan parameter secara berurutan mulai dilakukan. Dimulau dari Tipe Mobil, Model Mobil, Tinggi Mobil, Departemen, Stock, dan new parameter. Lalu pada Line 24 hingga 29 berfungsi untuk mengembalikan response gagal jika memang terdapat parameter yang mau diinput sudah terdaftar.

Segmen Program 5.11 Pembuatan Hasil Result

1. $listmasterkit = Masterkit::all();
2. $masterkit = new Masterkit();
3. foreach ($listmasterkit as $kit) {
4. if(strtoupper($kit->kode\_kit) ==strtoupper($request->param)){
5. $masterkit = $kit;
6. return response()->json([
7. "success" => true,
8. "statuscode" => 201,
9. "data" =>  strtoupper($request->param),
10. "message" => $masterkit,
11. "result" => $masterkit,
12. "hasil" => $kitsudahada
13. ]);
14. }
15. }
16. if (response.data.statuscode == 201) {
17. let datanamakit = response.data.result.nama\_kit
18. let dataKodeKit = response.data.result.kode\_kit
19. let dataisikit = response.data.result.komponen
20. let kembar = false
21. this.Result.forEach(element => {
22. if(element.NamaKit.toUpperCase()==datanamakit.toUpperCase()) {
23. kembar = true
24. }
25. });
26. }
27. if (kembar) {
28. this.$swal({
29. title: 'Kode Kit ' + this.InputKodeKit + ' sudah ada',
30. icon: 'error'
31. });
32. }

Line 1 adalah proses menarik semua data kit yang ada. Line 2 berfungsi untuk membuat sebuah object kosong. Kemudian pada line 3 hingga 14 adalah proses looping yang mana mencari kecocokkan kode kit antara yang ada pada data kit dengan kode yang diinput oleh user. Ketika terjadi kecocokkan nama kode barulah server mengembalikkan response sukses.

Line 16 kebawah adalah bahasa javascript. Line 16 berfungsi untuk melakukan proses cek status code dari response. Ketika cocok maka fungsi dibawahnya akan dijalankan. Selanjutnya pada line 17 sampai 25 adalah proses melakukan cek apakah kode kit yang dipakai sudah pernah dipakai. Jika ternyata kembar maka akan muncul peringatan yang menunjjukan error bahwa pemakaian kode kit ini sudah pernah dipakai line 27 sampai 32.

Segmen Program 5.12 Pengecekkan result

1. $kosongkit = false;
2. if (count($kit) > 0) {
3. foreach ($kit as $subkit) {
4. $subkit["NamaKit"] = ucwords($subkit["NamaKit"]);
5. if (count($subkit['IsiKit']) > 0) {
6. $j = 0;
7. foreach ($subkit['IsiKit'] as $komponen) {
8. $komponen[$j]['nama\_komponen']=ucwords($komponen['nama\_komponen']);
9. $j++;
10. }
11. }
12. foreach ($subkit['IsiKit'] as $komponen) {
13. if ($komponen['nama\_komponen'] == null || $komponen['qty'] == null) {
14. $kosongkit = true;
15. break;
16. }
17. }
18. }
19. } else {
20. return response()->json([
21. "success" => true,
22. "statuscode" => 405,
23. ]);
24. }
25. if ($kosongkit) {
26. return response()->json([
27. "success" => true,
28. "statuscode" => 408,
29. ]);
30. }

Line 1 berfungsi sebagai penanda apakah kit ada atau tidak. Selanjutnya dilakukan proses pengecekkan apakah kit ada atau kosong termasuk isi dari kit tersebut Line 2 sampai 18. Line 19 sampai 24 adalah pengembalian response gagal jika ternyata ada isi dari komponen yang kosong. Line 25 sampai 30 adalah pengembalian response gagal untuk kit yang kosong sama sekali.

Segmen Program 5.13 Input Data Master

1. $Newmaster = Master::create([
2. "Kit" => $kit,
3. "Parameter" => $param,
4. ]);
5. return response()->json([
6. "success" => true,
7. "statuscode" => 200,
8. "kit" => $kit
9. ]);

Segmen ini berfungsi untuk melakukan penambahan data pada database master setelah berhasil melalui semua pengecekkan. Line 1 sampai 4 adalah proses penambahan data baru. Line 5 sampai 9 adalah bentuk pengembalian response berhasil. Line 7 status code menunjukkan kode keberhasilan.

1. Konversi

Konversi adalah fitur yang mencocokkan data parameter yang berada pada SPK dengan data parameter yang telah diisikan di master. Nantinya SPK yang berhasil dalam proses kecocokkan akan menampilkan data komponen – komponen yang diperlukan. Konversi yang tidak berhasil akan menampilkan data komponen kosong dan sekiranya permasalahan dari alasan mengapa tidak lolos pada proses cek. Hal ini akan ditampilkan ketika user menekan tombol problem pada SPK yang tidak lolos proses pengecekkan.

Segmen Program 5.14 Validasi Input SPK

1. if (this.SPKfield == "" || this.stall == "" || this.stall == 0 || this.NamaStall == "" || this.Departemen == "") {
2. this.$swal({
3. title: 'pengisian SPK tidak Valid',
4. icon: 'error'
5. });}

Segmen ini bertugas untuk mengecek apakah input SPK yang mau dicek sudah memenuhi kriteria. Pada line 1 hingga 5 adalah proses pengecekkan apakah ada field yang kosong. Di line 2 hingga 5 adalah proses memunculkan penringatan dialog box. Dan di line 3 adalah isi judul pesan error tersebut. Swal ini adalah fungsi yang berasal dari library Sweetalert.

Segmen Program 5.15 Penambahan Input SPK

1. $newdata = SavedConversionResult::create([
2. "NOSPK" => $spk->NOSPK,
3. "stall" => $stall,
4. "namastall" => $namastall,
5. "Departemen" => $Departemen,
6. "checked" => false,
7. "status" => "Pending",
8. "parameter" => $parameter,
9. "created\_at" => Carbon::now()->format('Y-m-d H:i:s'),
10. "updated\_at" => Carbon::now()->format('Y-m-d H:i:s'),
11. ]);
12. return response()->json([
13. "success" => true,
14. "status" => 200,
15. "spk" => $spk,
16. "namastall" => $namastall,
17. "Departemen" => $Departemen,
18. "newdata" => $parameter,
19. "allsaved" => $allsaved
20. ]);

Segmen ini bertugas untuk mengecek apakah input SPK yang mau dicek sudah memenuhi kriteria. Pada line 1 hingga 6 adalah proses pengecekkan apakah ada field yang kosong. Di line 2 hingga 5 adalah proses memunculkan penringatan dialog box. Dan di line 3 adalah isi judul pesan error tersebut. Swal ini adalah fungsi yang berasal dari library Sweetalert.

Segmen Program 5.16 Konversi SPK

1. foreach ($master as $item2) {
2. $data = SPK::where('NOSPK', $item1["NOSPK"])->first();
3. $ModelMobilterdaftar = false;
4. $TinggiMobilterdaftar = false;
5. $TipeMobilTerdaftar = false;
6. $DepartemenTerdaftar = false;
7. $StallTerdaftar = false;
8. $newparameterTerdaftar = false;
9. foreach ($item2["Parameter"]["ModelMobil"] as $subitem2) {
10. if(strtoupper($subitem2)==strtoupper($data["parameter"][

Segmen Program 5.16 (Lanjutan)

1. "ModelMobil"])){
2. $ModelMobilterdaftar = true;
3. $errorModelMobil = false;
4. break;
5. }
6. }
7. foreach ($item2["Parameter"]["TinggiMobil"] as $subitem2) {
8. if(strtoupper($subitem2)==strtoupper($data["parameter"]["TinggiMobil"])) {
9. $TinggiMobilterdaftar = true;
10. $errorTinggiMobil = false;
11. break;
12. }
13. }
14. if($ModelMobilterdaftar && $TinggiMobilterdaftar && $TipeMobilTerdaftar && $DepartemenTerdaftar && $StallTerdaftar && $newparameterTerdaftar) {
15. array\_push($result, $item2["Kit"]);
16. $i++;
17. }
18. if ($i > 0) {
19. array\_push($results, [
20. 'kit' => $result,
21. 'NoSPK' => $item1->NOSPK,
22. ]);
23. $item1["status"] = "berhasil";
24. $item1["kit"] = $results;
25. $item1->save();
26. } else {
27. $item1["status"] = "Pending";
28. $item1->save();
29. }

Line 1 adalah proses melakukan looping data master untuk di crosscheck dengan parameter SPK.di line 2 adalah proses pengambilan data SPK. Line 3 sampai 8 adalah pemberian tanda yang nanti akan berguna untuk memberikan informasi apakah parameter SPK *matching* dengan parameter pada database master. Kemudian di baris 9 sampai 16 adalah proses pengecekkan parameter model mobil. Lalu pada baris 17 sampai 23 adalah proses pengecekkan untuk parameter tinggi mobil. Seharusnya masih ada untuk pengecekkan yang lain hanya saja segmen akan menjadi kepanjangan karena secara garis besar fungsi yang dijalankan sama.

Line 24 sampai 27 adalah proses pengelompokkan data kit yang berhasil didapatkan. Kemudian pada baris 28 hingga 32 adalah proses merapikan data sebelum dilakukan proses save pada database. Kemudian pada baris 33 sampai 35 adalah fungsi untuk menyimpan data yang telah diatur tadi. Lalu pada line 36 sampai 39 adalah untuk menyimpan perubahan status untuk data SPK yang tidak berhasil ditemukan kitnya. Nantinya data yang tidak ditemukan tadi akan membuat munculnya tobol problem pada halaman input SPK yang mana bisa dicek untuk melihat pesan errornya.

1. Datatable

Datatable adalah fitur yang diperlukan pada kebanyakan web yang banyak berhubungan dengan laporan. Disini datatable berperan dalam menampilkan data yang banyak dalam bentuk yang rapi. Selain itu beberapa fungsi datatable disini juga adalah sebagai media untuk melakukan aksi terhadap data seperti edit, insert, dan delete. Datatable juga memiliki fitur untuk melakukan filter tanpa merusak data aslinya.

Segmen Program 5.17 Inisialisasi Datatable

1. headerstable: [
2. {text: 'Nomor SPK',
3. align: 'start',
4. sortable: false,
5. value: 'NOSPK',
6. class: "title text-uppercase font-weight-black black--text light-blue lighten-5"
7. },
8. { text: 'Nama Stall', value: 'namastall', class: "title text-uppercase font-weight-black black--text light-blue lighten-5" },
9. { text: 'Stall', value: 'stall', class: "title text-uppercase font-weight-black black--text light-blue lighten-5" },
10. { text: 'Departemen', value: 'Departemen', class: "title text-uppercase font-weight-black black--text light-blue lighten-5" },
11. { text: 'Status', value: 'status', width: '150px', class: "title text-uppercase font-weight-black black--text light-blue lighten-5" },
12. { text: 'Waktu Update Terakhir', width: '250px', value: 'updated\_at', class: "title text-uppercase font-weight-black black--text light-blue lighten-5" },
13. { text: 'Action', value: 'actions', width: '300px', class: "title text-uppercase font-weight-black black--text light-blue lighten-5" },
14. ],

Line 1 adalah line pembuka untuk memberikan judul tabel. Line 2 sampai 7 adalah fungsi untuk membuat sebuah kolom. Line 3 adalah untuk memberikan alignment pada tabel. Line 4 berfungsi untuk memberikan fitur pada kolom yang bersangkutan apakah bisa dilakukan sortir atau tidak. Line 5 adalah tempat isi datanya nanti. Line 6 adalah untuk memberikan css inline pada html. Sama halnya pada line 8, 9, 10, 11, 12, dan 13 masing masing line adalah untuk membuat sebuah kolom.

Segmen Program 5.18 Pengisian Datatable

1. Await axios.post('/api/getdatatable',

{Role:this.authStore.user.account\_privileges.title, Departemen:this.authStore.user.account\_privileges.account\_dept}).then((response) => {

1. this.datatable = []
2. this.datatable = response.data.reverse()
3. this.datatable.forEach(element => {
4. element["updated\_at"]
5. this.converttime(element["updated\_at"])
6. });
7. })

Line 1 berfungsi untuk melakukan request untuk mendapatkan data yang akan diisi pada datatable. Selanjutnya pada baris 16 datatable dikosongkan terlebih dahulu sebelum diisi. Line 17 berfungsi untuk membalik data dari posisi baru ke lama sekaligus melakukan pengisian pada datatable. Selanjutnya pada line 19 pada masing masing baris yang berasal dari data object yang sama ditambahkan data updated\_at. Kemudian pada baris 20 format data diubah. Data datatable akan secara otomatis melakukan update data sesuai pada value this.datatable.

1. Excel

Excel adalah kebutuhan yang sudah tidak dapat terpisahkan lagi. Excel sendiri sudah lazim ditemui pada lingkungan perkantoran. Pada program ini nantinya excel dipakai untuk melakukan proses download datatable yang telah menampilkan komponen – komponen SPK. Proses download akan dimulai ketika user menekan tombol download excel pada halaman show result. Secara otomatis file yang telah di download tadi akan tersimpan pada folder download (default) atau di folder lain sesuai dengan pengaturan download pada masing – masing komputer klien.

Segmen Program 5.19 Pengisian Datatable

1. import JsonExcel from "vue-json-excel"
2. json\_fields: {
3. "NO SPK": "NoSPK",
4. "Kode Kit": "kode",
5. "Nama Kit": "namakit",
6. "Nama Komponen": "namakomponen",
7. "Kebutuhan": "Qty",
8. "Siteid": "siteID",
9. "Dari Rak": "Dari",
10. "Ke Rak": "Kerak",
11. },
12. <JsonExcel class="btn btn-primary"
13. :data="datatable" :fields="json\_fields"
14. worksheet="My Worksheet" name="filename.xls" style="margin-right: 20px;">
15. Download Excel <font-awesome-icon icon="fa-solid fa-download" />
16. </JsonExcel>

Line 1 adalah fungsi untuk melakukan impor fungsi Jsonexcel dari library vue-json-excel. Line 12 bergungsi untuk melakukan menampilkan tombol download pada html. Nanti tombol bisa diberikan css inline. Di line 13 data bertugas sebagai penyedia data dan fields bertugas untuk menyiapkan format data yang akan didownloa. Lalu di line 14 worksheet bertugas untuk memberikan nama sheet pada excel, lalu name untuk nama file excelnya. Line 15 bertugas untuk memberikan icon download. Line 16 adalah tutup tag htmlnya.

1. Print

Print adalah fitur yang tidak kalah penting. Bagi perkantoran fitur ini hal yang cukup penting. Penggunaannya sangatlah sederhana hanya dengan klik tombol user akan diarahkan ke halaman untuk melakukan proses print. Tidak ada yang special dari print disini karena sudah jelas dengan fungsinya.

Segmen Program 5.20 Print

1. <div id="printMe">
2. <div id="image"></div>
3. <v-data-table dense :headers="headerstable" :items="datatable" :items-per-page="30" :search="search"
4. class="elevation-1 font-weight-bold">
5. <template v-slot:top>
6. <v-toolbar flat>
7. <v-toolbar-title>List daftar komponen</v-toolbar-title>

Segmen Program 5.20 (Lanjutan)

1. <h5 class="tanggal">tanggal{{ new Date().toLocaleString() }}</h5>
2. </v-toolbar>
3. </template>
4. </v-data-table>
5. </div>
6. const options = {
7. styles: [
8. '/css/print.css' // <- inject here
9. ]
10. }
11. async print() {
12. $("#image").append(`<img src='/images/Logo\_Adi\_Putro.svg' alt='' srcset=''>`)
13. $(".v-data-footer\_\_select").html('')
14. $(".v-data-footer\_\_pagination").html('')
15. $(".v-data-footer\_\_icons-before").html('')
16. $(".v-data-footer\_\_icons-after").html('')
17. await this.$htmlToPaper('printMe', options);
18. $("#image").html(``);
19. },

Line 1 hingga 12 adalah html untuk memasang id print pada data table. Hal ini harus dilakukan untuk menandai area pada html yang mau di print. Line 13 sampai 17 adalah fungsi untuk memberikan css pada halaman print nantinya. Ini diperuntukkan untuk memberikan kerapian. Kemudian di line 18 sampai 26 adalah fungsi untuk melakukan fitur print, nanti ada beberapa hal yang akan dihilangkan dan ada yang ditambah, seperti memberikan logo gambar perusahaan.

1. Halaman Setting

Halaman Setting berfungsi bagi super admin untuk melakukan managemen websitenya. Dimulai dari proses manajemen akun, manajemen departemen, dan manajemen stall. Namun proses codingnya dibuat dalam satu kesatuan karena memiliki kemiripan walaupun terpisahkan oleh path yang berbeda. Selain super admin halaman setting tidak dapat diakses.

Segmen Program 5.21 Ganti tampilan

1. import User from './User.vue';
2. import Departemen from './Departemen.vue';
3. import Stall from './Stall.vue';
4. <v-container fill-height fluid>
5. <v-row justify="center">

Segmen Program 5.21 (Lanjutan)

1. <v-btn-toggle >
2. <v-btn

@click="changetoggle('Account')" :class="[toggleAccount=='Account'? 'aa' : 'white']">

1. <v-icon>Account</v-icon>
2. </v-btn>
3. <v-btn @click="changetoggle('Departemen')"
4. :class="[toggleAccount=='Departemen'?'aa':'white']>
5. <v-icon>Departemen</v-icon>
6. </v-btn>
7. <v-btn @click="changetoggle('Stall')":class="[toggleAccount=='Stall'?'aa':'white']">
8. <v-icon>Stall</v-icon>
9. </v-btn>
10. </v-btn-toggle>
11. <div v-if="toggleAccount === 'Account'">
12. <User />
13. </div>
14. <div v-if="toggleAccount === 'Departemen'">
15. <Departemen />
16. </div>
17. <div v-if="toggleAccount === 'Stall'">
18. <Stall />
19. </div>
20. </v-row>
21. </v-container>

Line 1 hingga 3 adalah proses melakukan impor komponen. Line 4 adalah tag pembuka untuk mengisi. Line 5 adalah untuk membuat tampilan memiliki align justify. Line 6 sampai 17 adalah tempat untuk memberikan tombol yang nantinya akan digunakan untuk melakukan routing Line 18 sampai 28 adalah tempat untuk melakukan aktivasi routing. Jika true maka tampilkan komponen ini.

1. Insert, Update dan Delete pada Model

Insert update dan delete pastinya diperlukan dalam proses melakukan update data pada database. di sub bab ini akan menjelaskan sekiranya gambaran ringkas terkait dengan cara melakukan insert, update, dan delete. Hal ini bertujuan untuk memberikan gambaran singkat mengenai cara codenya. Karena segmen – segmen code ini merupakan hal yang sangat penting dan pastinya memiliki sifat yang sangat berulang.

Segmen Program 5.22 Insert Code Singkat

1. $user = Account::create([
2. "account\_username" => $request->username,
3. "account\_name" => $request->name,
4. "password" => $request->password,
5. 'account\_privileges' => [
6. 'title' => $request->role,
7. 'account\_dept' => $request->departemen,
8. ],
9. "account\_active" => false,
10. ]);

Pada Line 1 kita membuat objek kosong dan menentukan model data yang mau di lakukan proses penambahan data (insert). Dalam hal ini account adalah model data yang kita pakai sebagai contoh, kode create artinya kita mau melakukan insert pada objek model. Selanjutnya line 2 hingga 10 adalah kode singkat terkait data apa saja yang mau dimasukkan. Menggunakan mongodb membisakan user untuk melakukan insert data dalam bentuk object lihat line 5 sampai 8.

Segmen Program 5.23 Update Code Singkat

1. $departemenupdate=Departemen::where('\_id',$request>id)>first();
2. $departemenupdate->Nama\_Departemen= $request->namadepartemen;
3. $departemenupdate->AksesTipeDatabase=$request->databaseakses;
4. $departemenupdate->save();

Objek contoh kali ini adalah untuk model data departemen. Langkah pertama yang harus dilakukan adalah mengambil atau mencari id data mana yang mau diubah, hal ini dilakukan pada line 1. Selanjutnya pada line 2 data yang kita dapatkan kita ganti valuenya. Begitu juga pada line 3 kita mengganti data akses tipe database. Selanjutnya pada line 4 kita melakukan proses save untuk mengupdate data yang telah kita update. Perlu diingat nama panah yang diisikan disesuaikan dengan area fillable pada model data. Jika tidak disebutkan maka data tidak akan masuk dan diberikan peringatan bahwa perubahan yang dilakukan tidak akan berdampak apa – apa.

Segmen Program 5.24 Delete

1. $departemenupdate=Departemen::where('\_id',$request>id)>first();
2. $hapusdepartemen->delete();

Masih objek contoh yang sama disini objek contohnya adalah data departemen. Begitu pula halnya dengan proses update, delete juga perlu menentuka id mana yang mau dilakukan penghapusan. Hal ini didapatkan dengan code pada line nomor 1. Selanjutnya user yang telah mendapatkan idnya bisa melakukan panah delete yang ada pada line 2 untuk melakukan proses penghapusan. Jika model yang kita berikan menerapkan softdelete maka data tidak dihapus melainkan diberikan atribut tambahan berupa delete\_at yang berfungsi untuk menandai bahwa data ini sudah pernah dihapus. Jangan pernah membuat atribut dengan nama itu ketika menerapkan soft delete. Sebaliknya jika model tidak menerapkan soft delete maka data langsung akan dihapus.

Segmen Program 5.25 Model

1. <?php
2. namespace App\Models;
3. use Illuminate\Database\Eloquent\Factories\HasFactory;
4. use Jenssegers\Mongodb\Eloquent\Model;
5. use Jenssegers\Mongodb\Auth\User as Authenticatable;
6. use Illuminate\Notifications\Notifiable;
7. use Jenssegers\Mongodb\Eloquent\SoftDeletes;
8. use Laravel\Sanctum\HasApiTokens;
9. class Account extends Authenticatable
10. {
11. use HasApiTokens, HasFactory, Notifiable, SoftDeletes;
12. protected $connection = 'mongodb';
13. protected $collection = 'Accounts\_DB';
14. public $timestamps      = true;
15. protected $fillable = [
16. 'account\_name',
17. 'account\_username',
18. 'account\_password',
19. 'account\_privileges',
20. 'account\_picture',
21. 'account\_desc',
22. 'account\_active',
23. 'api\_token'
24. ];
25. protected $hidden = [
26. 'password',
27. 'remember\_token',
28. ];
29. protected $casts = [
30. 'email\_verified\_at' => 'datetime',
31. ];
32. }

Model merupakan hal yang sangat penting dalam framework laravel. Sebenarnya proses melakukan input bisa dilakukan tanpa harus melakukan model, namun model memberikan kepastian bagi programmer untuk memastikan bahwa tujuan data benar menuju objek yang bersangkutan. Pada Line 1 adalah tag pembuka php yang menandai bahwa program ini adalah program PHP. Selanjutnya pada line 2 hingga 8 adalah proses mengimpor library yang diperlukan. Khusus yang memiliki “\mongodb\” hal itu harus diperhatikan, karena tidak bisa menggunakan bawaan model dari versi laravel sendiri.

Selanjutnya di line 9 terdapat line “extends authenticatable” yang artinya bahwa model ini dapat melakukan fungsi autentikasi seperti proses login dan logout. Lalu di line 11 berfungsi untuk memberikan token untuk proses cookie, has factory untuk memberi tau bahwa model dapat digunakan dalam pembuatan banyak data dummy, notifiable untuk proses pemberian notifikasi, dan soft delete adalah untuk memberikan penanda bahwa data yang bersangkutan dihapus pada datetime sekian. Line 12 adalah proses penentuan koneksi yang dipakai. Line 13 adalah nama tabel yang akan diakses pada database koneksi. Line 14 untuk proses pencatatan dengan pemberian dua field baru yaitu created\_at dan updated\_at. Dilanjutkan dengan line 15 sampai 24 adalah proses pengisian nama field yang bisa diisi, artinya jika nanti data yang diinput tidak memiliki data yang sama maka data tidak akan masuk. Hal ini bertujuan untuk mencegah data lain yang tidak disebut masuk ke dalam database. Line 25 hingga 28 adalah untuk pemberian field di database tapi disembunyikan sehingga tidak ketahuan, biasanya data yang penting seperti password atau token. Line 29 hingga 31 bertugas untuk mengubah tipe data sesuai yang diinginkan dalam contoh ini verifikasi email yang diubah ke dalam tipe data datetime.

Dalam contoh kali ini hanya menggunakan data account. Padahal pada kenyataan pengembangannya akan banyak sekali penggunaan model dan sering kali code ini akan berulang. model data ini juga digunakan pada data departemen, master, stall, konversi. Tujuannya tidak lain adalah untuk memudahkan proses insert, update dan delete data.