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'),
8. 'password' => env('DB\_PASSWORD', 'secret'),
9. 'options' => [
10. 'database'=>env('DB\_AUTHENTICATION\_DATABASE' ,'admin'),
11. ],
12. ],
13. ],

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: {
11. async getUser() {
12. this.authuser = this.authuser
13. },
14. async handleLogin(data) {
15. this.authErrors = [];
16. try {
17. await axios.post('/api/login', data).then((response) => {
18. if (response.data[0] == null) {
19. this.authuser = null
20. }
21. else {
22. this.authuser = response.data[0].user
23. }
24. });
25. } catch (error) {}
26. },
27. async handleLogout() {
28. try {
29. this.authuser = null;
30. } catch (error) {
31. }
32. },
33. },
34. persist: true
35. })

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();
3. if (!to.meta.guestPageAccess) {
4. store.getUser();
5. const routes = [
6. if (store.user) {
7. if (store.user.account\_privileges.title == "Super Admin Role") {
8. next()
9. }
10. else if (store.user.account\_privileges.title == "Admin Role") {
11. if (to.meta.levelAccess == "Admin Role" || to.meta.levelAccess == "Staff Role") {
12. next()
13. }
14. }
15. else if (store.user.account\_privileges.title == "Staff Role") {
16. if (to.meta.levelAccess == "Staff Role") {
17. next()
18. }
19. }else{
20. next({
21. name: 'NotFound'
22. })
23. }
24. }
25. else {
26. next({
27. name: 'Login'
28. })}
29. }
30. else {
31. next()
32. }
33. })

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. 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.

Segmen Program 5.6 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 5.7 Cek Inputan Kosong

1. function FungsicekKosong(array $cek)
2. {
3. if (count($cek) == 0) {
4. return true;
5. }
6. }
7. $paramkosong = false;
8. if (!$paramkosong) {
9. $paramkosong = FungsicekKosong($param['TipeMobil']);
10. }
11. if (!$paramkosong) {
12. $paramkosong = FungsicekKosong($param['ModelMobil']);
13. }
14. if (!$paramkosong) {
15. $paramkosong = FungsicekKosong($param['TinggiMobil']);
16. }
17. if (!$paramkosong) {
18. $paramkosong = FungsicekKosong($param['Departemen']);
19. }
20. if (!$paramkosong) {
21. $paramkosong = FungsicekKosong($param['Stall']);
22. }
23. if ($paramkosong) {
24. return response()->json([
25. "success" => true,
26. "statuscode" => 401,
27. ]);
28. }
29. $paramtambahankosong = false;
30. foreach ($param['NewParameter'] as $newparam) {
31. if ($newparam['Newparam'] == "" || $newparam['Newparam'] == null) {
32. $paramtambahankosong = true;
33. break;
34. }
35. foreach ($newparam['Component'] as $komponen) {
36. if ($komponen == "" || $komponen == null) {
37. $paramtambahankosong = true;
38. break;
39. }
40. }
41. if ($paramtambahankosong) {
42. break;
43. }
44. }
45. if ($paramtambahankosong) {
46. return response()->json([
47. "success" => true,
48. "statuscode" => 403,
49. ]);
50. }

Segmen Program 5.7 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. ]);
31. }
32. $paramtambahansama = false;
33. $judulparamtambahankembar = array();
34. foreach ($param['NewParameter'] as $newparam) {
35. array\_push($judulparamtambahankembar, $newparam['Newparam']);
36. if (count($newparam['Component']) !== count(array\_unique($newparam['Component']))) {
37. $paramtambahansama = true;
38. break;
39. }
40. }
41. if (count($judulparamtambahankembar) !== count(array\_unique($judulparamtambahankembar))) {
42. $paramtambahansama = true;
43. }
44. if ($paramtambahansama) {
45. return response()->json([
46. "success" => true,
47. "statuscode" => 404,
48. ]);
49. }
50. function fungsicekparameterterdaftar(array $array1, array $array2)
51. {
52. $jumlahkesamaan = 0;
53. foreach ($array2 as $isiarray2) {
54. foreach ($array1 as $isiarray1) {
55. if (strtoupper($isiarray1) == strtoupper($isiarray2)) {
56. $jumlahkesamaan++;
57. break;
58. }
59. }
60. }
61. if ($jumlahkesamaan == count($array2)) {
62. return true;
63. }
64. }

Segmen Program 5.7 Fungsi Menambah kembar

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. $cekAdditionaParameter = true;
23. } elseif (count($saved['NewParameter']) == count($param['NewParameter'])) {
24. $judulparamsama = 0;
25. foreach ($saved['NewParameter'] as $item) {
26. foreach ($param['NewParameter'] as $item2) {
27. if (strtoupper($item['Newparam']) == strtoupper($item2['Newparam'])) {
28. $komponensama = 0;
29. foreach ($item['Component'] as $komponendatabase) {
30. foreach ($item2['Component'] as $komponennew) {
31. if (strtoupper($komponendatabase) == strtoupper($komponennew)) {
32. $komponensama++;
33. }
34. }
35. }
36. if ($komponensama == count($item2['Component'])) {
37. $judulparamsama++;
38. }
39. }
40. }
41. }
42. if ($judulparamsama == count($saved['NewParameter'])) {
43. $cekAdditionaParameter = true;
44. }
45. }
46. if ($cekTipeMobil && $cekModelMobil && $cekTinggiMobil && $cekDepartemen && $cekStock && $cekstall && $cekAdditionaParameter) {
47. return response()->json([
48. "success" => true,
49. "statuscode" => 406,
50. ]);
51. }
52. }