Passport API dengan LUMEN

1. Langkah awal kita harus menginstal package Lumen passport di dalam project kita dengan mengetikan perintah berikut di terminal:

|  |
| --- |
| composer require dusterio/lumen-passport |

1. Selanjutnya jika ada error pada proses instalasi kita dapat menginstlanya secara manual dengan cara menambahkan perintah berikut didalam file composer.json :

|  |
| --- |
| {  "require": {  "dusterio/lumen-passport": "^0.3.6"  }  } |

Jika sudah maka ketikan perintah “composer update” pada terminal

1. Pergi ke bootstrap/app.php untuk mengaktifkan penyedia Laravel Passport dan penyedia khusus Lumen Passport :

|  |
| --- |
| /\*\* @file bootstrap/app.php \*/  // hilangkan komentar ada pada baris ke 26  $app->withFacades();  // hilangkan komentar ada pada baris ke 28  $app->withEloquent();  // hilangkan komentar ada pada baris ke 82-85  $app->routeMiddleware([  'auth' => App\Http\Middleware\Authenticate::class,  ]);  // hilangkan komentar ada pada baris ke 98-100  $app->register(App\Providers\AppServiceProvider::class);  $app->register(App\Providers\AuthServiceProvider::class);  $app->register(App\Providers\EventServiceProvider::class); |

1. Selanjutnya tambahkkan sintaks berikut pada baris ke 101

|  |
| --- |
| // untuk Register dua service providers, Laravel Passport and Lumen adapter  $app->register(Laravel\Passport\PassportServiceProvider::class);  $app->register(Dusterio\LumenPassport\PassportServiceProvider::class); |

1. Berikutnya masih berada di file bootstrap/app.php tambahkan kelas aplikasi seperti berikut :

|  |
| --- |
| /\*\* @file bootstrap/app.php \*/  $app = new Laravel\Lumen\Application(  dirname(\_\_DIR\_\_)  );  // tambahkan kelas dibawah ini di line 26  $app = new \Dusterio\LumenPassport\Lumen7Application(  dirname(\_\_DIR\_\_)  ); |

1. Buatlah folder config dan didalam folder buat file auth.php “Config/auth.php” isi kan dengan sintaks berikut:

|  |
| --- |
| <?php  return [  'defaults' => [  'guard' => 'api',  'passwords' => 'users',  ],  'guards' => [  'api' => [  'driver' => 'passport',  'provider' => 'users',  ],  ],  'providers' => [  'users' => [  'driver' => 'eloquent',  'model' => \App\Models\User::class  ]  ]  ]; |

1. Setelah itu tambahkan sintak ini di dalam botstrap/app.php:

|  |
| --- |
| /\*\* @file bootstrap/app.php \*/  $app->configure('auth');  //Atau bisa pada line 69 kita dapat mengganti sintak ini :  $app->configure('app');  //menjadi  $app->configure('auth'); |

1. Selanjutnya, kita harus memanggil LumenPassport::routes metode dalam bootmetode aplikasi . Metode ini akan mendaftarkan rute yang diperlukan untuk mengeluarkan token akses dan mencabut token akses, klien, dan token akses pribadi:

|  |
| --- |
| /\*\* @file app/Providers/AuthServiceProvider.php \*/  use Dusterio\LumenPassport\LumenPassport;  class AuthServiceProvider extends ServiceProvider  {  public function boot()  {  LumenPassport::routes($this->app);  /\* rest of boot \*/  }  } |

1. Pada folder app/models/user.php kita isi tambahkan dengan sintaks berikut ini :

|  |
| --- |
| // pada line 11  use Laravel\Passport\HasApiTokens;  //pada line 15  use Authenticatable, Authorizable, HasFactory, **HasApiTokens**; |

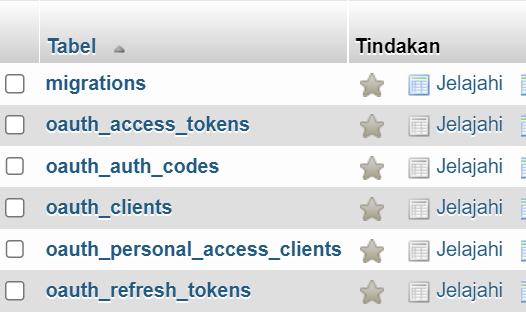
1. Setelah itu kita akan melakukan migrate, ketikan perintah berikut pada terminal

|  |
| --- |
| # Create new tables for Passport  php artisan migrate |

1. Pada kasus ini saya menemukan error “Call to undefined method Laravel\Lumen\Routing\Router::middleware()”. Cara mengatasi error tersebut kita pergi ke file composer.json dan tambahkan perintah berikut di dalam *“require”: {};*

|  |
| --- |
| "laravel/passport": "^10.4" |

Kita migrate kembali maka akan berhasil dan kita akan mendapatkan struktur tabel berikut ini:



1. Langkah selanjutnya kita ketikan perintah berikut pada terminal :

|  |
| --- |
| // perintah ini untuk Install encryption keys and other stuff for Passport  **php artisan passport:install** |

1. Langkah selanjutnya membuat tabel user\_acces\_api “tabel ini untuk akun user/pesonahub”. Kita akan membuat tabel dengan migration. Pertama ketikan perintah membuat migration:

|  |
| --- |
| Php artisan make:migration user\_acces\_api |

1. Jika suda buka file migration user\_acces\_api. Masukan sintaks berikut pada function up()

|  |
| --- |
| public function up()  {  Schema::create('user\_acces\_api', function (Blueprint $table) {  $table->increments('id');  $table->string('username');  $table->string('password');  $table->timestamps();  });  } |

1. Mengubah model users dengan sintak berikut ini

|  |
| --- |
| <?php  namespace App\Models;  use Illuminate\Auth\Authenticatable;  use Illuminate\Contracts\Auth\Access\Authorizable as AuthorizableContract;  use Illuminate\Contracts\Auth\Authenticatable as AuthenticatableContract;  use Illuminate\Database\Eloquent\Factories\HasFactory;  use Illuminate\Database\Eloquent\Model;  use Laravel\Lumen\Auth\Authorizable;  use Laravel\Passport\HasApiTokens;  class User extends Model implements AuthenticatableContract, AuthorizableContract  {  use Authenticatable, Authorizable, HasFactory, HasApiTokens;  protected $table = 'user\_acces\_api';  protected $primaryKey = 'id';  protected $fillable = [  'id', 'username', 'password',  ];  protected $hidden = [  // 'password',  ];  } |

1. Menseting router di folder routes->web.php seting dengan perintah berikut:

|  |
| --- |
| $router->post('api/login', ['uses' => 'AuthController@login']); |

1. Buat file AuthController dengan cara ketikan di terminal:

|  |
| --- |
| $router->post('api/login1', ['uses' => 'AuthController@login']);  $router->group(['prefix' => 'api', 'middleware' => 'auth'], function () use ($router) {  $router->get('/kups', ['uses' => 'KupsController@index']);  }); |

1. Lalu setelah berhasil membuat fle tersebut isi dengan sintak berikut ini :

|  |
| --- |
| <?php  namespace App\Http\Controllers;  use Illuminate\Http\Request;  use Illuminate\Support\Facades\Auth;  use Illuminate\Support\Facades\Hash;  use Illuminate\Support\Facades\Validator;  use App\models\User;  class AuthController extends Controller  {  public function login(Request $request)  {  $username = $request->input('username');  $password = $request->input('password');  $user = User::where('username', $username)->first();  if (!$user) {  return response()->json(['message' => 'Login gagal'], 401);  } else if ($user->password === $password) {  return response()->json([  'message' => 'login berhasil',  'api\_token' => $user->createToken('User')->accessToken,  'data' => $user  ]);  } else {  return response()->json([  'message' => 'login gagal',  ]);  }  }  } |

Penjelasan :

|  |
| --- |
| Untuk keamanan api ini menggunakan Passport yang dimana jika user ingin mengakses data api yang disediakan harus login terlebih pada postmant dengan cara memasukan url dengan method post lalu memasukan “*username dan password*” pada body yang sudah terdaftar pada database tabel yang telah dibuat. Jika user sudah melakukan login maka user akan mendapatkan token yang nantinya akan digunakan untuk melihat data api yang telah disediakan untuk token harus dimasukan di bagian Authorization pada bagian type ubah menjadi **barer token** pada postmant. Ketika user sudah memasukan token yang didapat jika benar maka akan berhasil menampilkan data yang telah disediakan. “*untuk setiap request user harus memasukan token yang telah didapat*” |

Contoh request yang berhasil menampilkan token ketika berhasil login:

