

Login dan Register

1. Api

```
19 Route::post('register', 'PetugasController@register');
20 Route::post('login', 'PetugasController@login');
21 Route::get('/', function() {
22     return Auth::user()->level;
23 }->middleware('jwt.verify');
```

2. Model User

```
1 <?php
2
3 namespace App;
4
5 use Illuminate\Notifications\Notifiable;
6 use Illuminate\Contracts\Auth\MustVerifyEmail;
7 use Illuminate\Foundation\Auth\User as Authenticatable;
8 use Tymon\JWTAuth\Contracts\JWTSubject;
9
10 class User extends Authenticatable implements JWTSubject
11 {
12     use Notifiable;
13     protected $table = 'petugas';
14
15     /**
16      * The attributes that are mass assignable.
17      *
18      * @var array
19      */
20     protected $fillable = [
21         'nama_petugas', 'alamat', 'no_hp', 'username', 'password', 'level',
22     ];
```

```

29     protected $hidden = [
30         'password', 'remember_token',
31     ];
32
33     public function getJWTIdentifier()
34     {
35         return $this->getKey();
36     }
37     public function getJWTCustomClaims()
38     {
39         return [];
40     }
41 }

```

3. Petugas Controller

```

5     use App\User;
6     use Illuminate\Http\Request;
7     use Illuminate\Support\Facades\Hash;
8     use Illuminate\Support\Facades\Validator;
9     use JWTAuth;
10    use Tymon\JWTAuth\Exceptions\JWTException;
11
12    class PetugasController extends Controller
13    {
14        public function login(Request $request)
15        {
16            $credentials = $request->only('username', 'password');
17
18            try {
19                if (!$token = JWTAuth::attempt($credentials)) {
20                    return response()->json(['error' => 'invalid_credentials'], 400);
21                }
22            } catch (JWTException $e) {
23                return response()->json(['error' => 'could_not_create_token'], 500);
24            }
25
26            return response()->json(compact('token'));
27        }
28
29        public function register(Request $request)
30        {
31            $validator = Validator::make($request->all(), [
32                'nama_petugas' => 'required|string|max:255',
33                'alamat' => 'required|string|max:255',
34                'no_hp' => 'required|string|max:255',
35                'username' => 'required|string|max:255',
36                'password' => 'required|string|min:6|confirmed',
37                'level' => 'required',
38            ]);

```

```

40     if($validator->fails()){
41         return response()->json($validator->errors()->toJson(), 400);
42     }
43
44     $user = User::create([
45         'nama_petugas' => $request->get('nama_petugas'),
46         'alamat' => $request->get('alamat'),
47         'no_hp' => $request->get('no_hp'),
48         'username' => $request->get('username'),
49         'password' => Hash::make($request->get('password')),
50         'level' => $request->get('level'),
51     ]);
52
53     $token = JWTAuth::fromUser($user);
54
55     return response()->json(compact('user','token'),201);
56 }
57
58 public function getAuthenticatedUser()
59 {
60     try {
61
62         if (! $user = JWTAuth::parseToken()->authenticate()) {
63             return response()->json(['user_not_found'], 404);
64         }
65
66     } catch (Tymon\JWTAuth\Exceptions\TokenExpiredException $e) {
67
68         return response()->json(['token_expired'], $e->getStatusCode());
69
70     } catch (Tymon\JWTAuth\Exceptions\TokenInvalidException $e) {
71
72         return response()->json(['token_invalid'], $e->getStatusCode());
73
74     } catch (Tymon\JWTAuth\Exceptions\JWTException $e) {
75
76         return response()->json(['token_absent'], $e->getStatusCode());
77
78     }
79
80     return response()->json(compact('user'));
81 }
82 }

```


6. Hasil Auth

GET

http://localhost/Laravel4/public/api/

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL BETA

	KEY	VALUE
<input checked="" type="checkbox"/>	username	Taetae
<input checked="" type="checkbox"/>	password	123456
<input type="checkbox"/>	nama_petugas	Kim Taehyung
<input type="checkbox"/>	no_hp	1234567890
<input type="checkbox"/>	password_confirmation	123456
<input type="checkbox"/>	alamat	Daegu
<input type="checkbox"/>	level	petugas
	Key	Value

Body

Cookies

Headers (11)

Test Results

Pretty

Raw

Preview

Visualize BETA

HTML

1 petugas

Nama : Fara Nisha Sukma Gustika

Kelas : Laravel