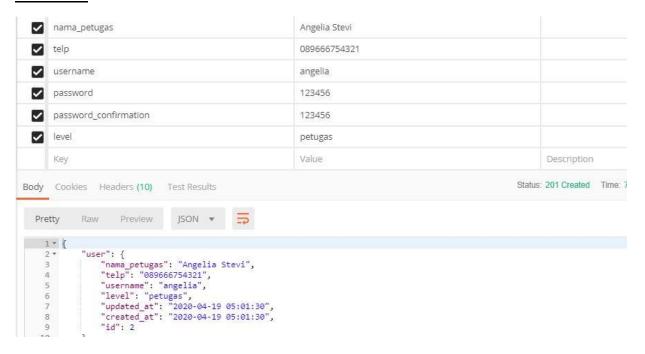
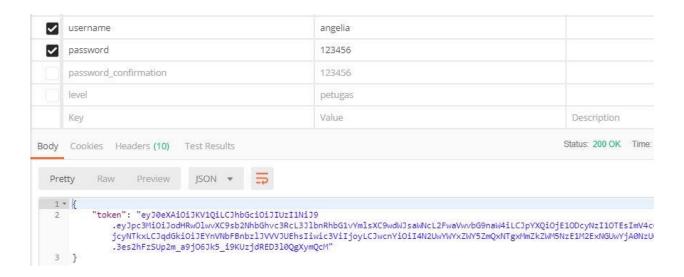
TUGAS RENTAL MOBIL

REGISTER



LOGIN



Scricpt

API

```
Route::post('register', 'PetugasController@register');
Route::post('login', 'PetugasController@login');
```

CONTROLLER

```
<?php
 2
     namespace App\Http\Controllers;
     use App\User;
 6
     use Illuminate\Http\Request;
     use Illuminate\Support\Facades\Hash;
     use Illuminate\Support\Facades\Validator;
 8
 9
     use JWTAuth;
10
     use Tymon\JWTAuth\Exceptions\JWTException;
11
12
     class PetugasController extends Controller
13
         public function login(Request $request)
14
15
16
             $credentials = $request->only('username', 'password');
17
18
             try {
                 if (! $token = JWTAuth::attempt($credentials)) {
19
                     return response()->json(['error' => 'invalid_credentials'], 400);
20
21
             } catch (JWTException $e) {
22
                 return response()->json(['error' => 'could_not_create_token'], 500);
23
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',
                  'telp' => 'required|string|max:255',
33
                  'username' => 'required|string|max:255',
34
35
                  'password' => 'required|string|min:6|confirmed',
                  'level' => 'required',
36
37
              ]);
38
              if($validator->fails()){
39
40
                  return response()->json($validator->errors()->toJson(), 400);
41
42
43
              $user = User::create([
44
                  'nama_petugas' => $request->get('nama_petugas'),
                  'telp' => $request->get('telp'),
45
46
                  'username' => $request->get('username'),
47
                  'password' => Hash::make($request->get('password')),
                  'level' => $request->get('level'),
48
49
              1);
50
51
             $token = JWTAuth::fromUser($user);
52
53
             return response()->json(compact('user', 'token'), 201);
         }
54
55
56
         public function getAuthenticatedUser()
57
58
             try {
59
                 if (! $user = JWTAuth::parseToken()->authenticate()) {
60
61
                      return response()->json(['user_not_found'], 404);
                 }
62
63
             } catch (Tymon\JWTAuth\Exceptions\TokenExpiredException $e) {
64
65
                 return response()->json(['token_expired'], $e->getStatusCode());
66
67
             } catch (Tymon\JWTAuth\Exceptions\TokenInvalidException $e) {
68
69
                 return response()->json(['token_invalid'], $e->getStatusCode());
70
71
             } catch (Tymon\JWTAuth\Exceptions\JWTException $e) {
72
73
74
                 return response()->json(['token_absent'], $e->getStatusCode());
75
76
77
             return response()->json(compact('user'));
78
79
80
```

MODEL

```
<?php
2
     namespace App;
4
     use Illuminate\Notifications\Notifiable;
     use Illuminate\Contracts\Auth\MustVerifyEmail;
6
     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
          * The attributes that are mass assignable.
16
17
          * @var array
18
         protected $fillable = [
19
             'nama_petugas', 'telp', 'username', 'password', 'level',
20
21
         ];
22
23
24
          * The attributes that should be hidden for arrays.
25
          * @var array
26
27
28
         protected $hidden = [
29
             'password', 'remember_token',
30
         ];
31
32
         public function getJWTIdentifier()
33
         {
34
             return $this->getKey();
35
         }
36
         public function getJWTCustomClaims()
37
38
             return [];
         }
39
40
41
```