## REGISTER LOGIN RENTAL MOBIL

1. API Register Login

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

2. Model

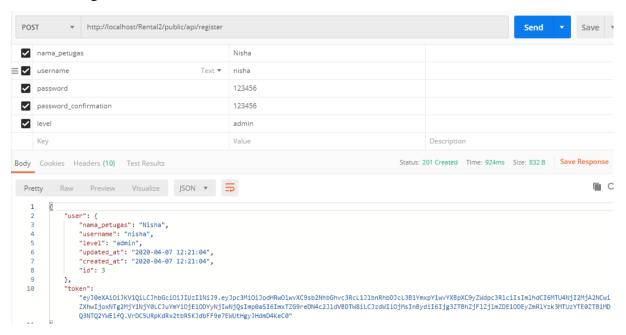
```
namespace App;
use Illuminate\Notifications\Notifiable;
use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Tymon\JWTAuth\Contracts\JWTSubject;
class User extends Authenticatable implements JWTSubject
    use Notifiable;
    protected $table = 'petugas';
       'nama_petugas', 'username', 'password', 'level',
        'password', 'remember_token',
    public function getJWTIdentifier()
    public function getJWTCustomClaims()
```

## 3. Controller

```
use App\User;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Validator;
use JWTAuth;
use Tymon\JWTAuth\Exceptions\JWTException;
class PetugasController extends Controller
        $credentials = $request->only('username', 'password');
            if (! $token = JWTAuth::attempt($credentials)) {
               return response()->json(['error' => 'invalid_credentials'], 400);
        } catch (JWTException $e) {
           return response()->json(['error' => 'could_not_create_token'], 500);
        return response()->json(compact('token'));
    public function register(Request $request)
        $validator = Validator::make($request->all(), [
            'nama_petugas' => 'required|string|max:255',
            'username' => 'required|string|max:255',
            'password' => 'required|string|min:6|confirmed',
            'level' => 'required',
        if($validator->fails()){
            return response()->json($validator->errors()->toJson(), 400);
```

```
$user = User::create([
       'nama_petugas' => $request->get('nama_petugas'),
        'alamat' => $request->get('alamat'),
        'no_hp' => $request->get('no_hp'),
        'username' => $request->get('username'),
        'password' => Hash::make($request->get('password')),
       'level' => $request->get('level'),
    $token = JWTAuth::fromUser($user);
   return response()->json(compact('user', 'token'), 201);
public function getAuthenticatedUser()
       if (! $user = JWTAuth::parseToken()->authenticate()) {
           return response()->json(['user_not_found'], 404);
   } catch (Tymon\JWTAuth\Exceptions\TokenExpiredException $e) {
       return response()->json(['token_expired'], $e->getStatusCode());
   } catch (Tymon\JWTAuth\Exceptions\TokenInvalidException $e) {
       return response()->json(['token_invalid'], $e->getStatusCode());
   } catch (Tymon\JWTAuth\Exceptions\JWTException $e) {
       return response()->json(['token_absent'], $e->getStatusCode());
   return response()->json(compact('user'));
```

## 4. Hasil Register



## 5. Hasil Login

