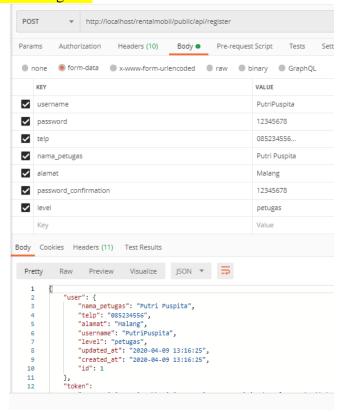
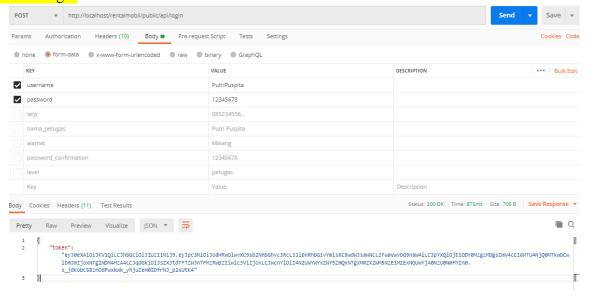
API Register



API Login



• Script api.php

```
Route::middleware('auth:api')->get('/user', function (Request $request) {
return $request->user();
});

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

Route::get('petugas', 'petugascontroller@getAuthenticatedUser')->middleware('jwt.verify');
```

Script petugas controller

```
app > Http > Controllers > ♥ petugascontroller.php
      namespace App\Http\Controllers;
      use App\User;
      use Illuminate\Http\Request;
      use Illuminate\Support\Facades\Hash;
      use Illuminate\Support\Facades\Validator;
      use Tymon\JWTAuth\Exceptions\JWTException;
       class petugascontroller extends Controller
           public function login(Request $request)
               $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',
                    'telp' => 'required|string|max:255',
                    'username' => 'required|string|max:255',
```

```
if($validator->fails()){
                return response()->json($validator->errors()->toJson(), 400);
                'nama_petugas' => $request->get('nama_petugas'),
                'telp' => $request->get('telp'),
                 'alamat' => $request->get('alamat'),
                '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()
            try {
                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());
73
             } catch (Tymon\JWTAuth\Exceptions\JWTException $e) {
                 return response()->json(['token_absent'], $e->getStatusCode());
             return response()->json(compact('user'));
```

Script Model