Tugas 4 JWT Laravel

Nama : Chatlea Cinta Putri Widyanto

Kelas : XI Laravel

No. Absen: 09

• SS Make Controller, Model, Middleware Make controller petugas

::\xampp\htdocs\perpustakaan>php artisan make:controller PetugasController

Make middleware

c:\xampp\htdocs\perpustakaan>php artisan make:middleware JwtMiddleware

Make model

::\xampp\htdocs\perpustakaan>php artisan make:model Petugas_model

• SS api.php, controller, model api.php

```
use Illuminate\Http\Request;
     API Routes
     | routes are loaded by the RouteServiceProvider within a group which
     | is assigned the "api" middleware group. Enjoy building your API!
     Route::middleware('auth:api')->get('/user', function (Request $request) {
     return $request->user();
     });
     Route::post('register', 'PetugasController@register');
19
     Route::post('login', 'PetugasController@login');
20
21
     Route::get('/', function(){
         return Auth::user()->level;
     })->middleware('jwt.verify');
23
     Route::get('user', 'PetugasController@getAuthenticatedUser')->middleware('jwt.verify');
```

Petugas controller

```
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Petugas_model;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Validator;
use JWTAuth:
use Tymon\JWTAuth\Exceptions\JWTException;
class PetugasController extends Controller
    public function login(Request $request)
        $credentials = $request->only('username', 'password');
        try {
            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',
            'alamat' => 'required|string|max:255',
            'telp' => 'required|string|max:255',
            'username' => 'required|string|max:255',
            'level' => 'required|string|max:255',
            'password' => 'required|string|min:6|confirmed',
        1);
        if($validator->fails()){
            return response()->json($validator->errors()->toJson(), 400);
```

```
1);
             if($validator->fails()){
                 return response()->json($validator->errors()->toJson(), 400);
             $user = Petugas_model::create([
                 'nama_petugas' => $request->get('nama_petugas'),
                 'alamat' => $request->get('alamat'),
                 'telp' => $request->get('telp'),
                 'username' => $request->get('username'),
                 'level' => $request->get('level'),
                 'password' => Hash::make($request->get('password')),
             1);
             $token = JWTAuth::fromUser($user);
             return response()->json(compact('user', 'token'),201);
         public function getAuthenticatedUser()
             try {
57
                 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'));
```

Model User

```
<?php
   namespace App;
   use Illuminate\Notifications\Notifiable;
6   use Illuminate\Contracts\Auth\MustVerifyEmail;
     use Illuminate\Foundation\Auth\User as Authenticatable;
8  use Tymon\JWTAuth\Contracts\JWTSubject;
10 class User extends Authenticatable implements JWTSubject
        use Notifiable;
        protected $table = "petugas";
         * The attributes that are mass assignable.
         protected $fillable = [
         'nama_petugas', 'alamat', 'telp', 'username', 'password', 'level'
20
         ];
         protected $hidden = [
         'password', 'remember_token',
         ];
         public function getJWTIdentifier()
32
        return $this->getKey();
        public function getJWTCustomClaims()
           return [];
```

Model Petugas

```
namespace App;
    use Illuminate\Notifications\Notifiable;
    use Illuminate\Contracts\Auth\MustVerifyEmail;
     use Illuminate\Foundation\Auth\User as Authenticatable;
     use Tymon\JWTAuth\Contracts\JWTSubject;
10 class Petugas_model extends Authenticatable implements JWTSubject
12
         use Notifiable;
         protected $table = "petugas";
14
         * The attributes that are mass assignable.
         * @var array
18
         protected $fillable = [
         'nama_petugas', 'alamat', 'telp', 'username', 'password', 'level'
20
         ];
         * @var array
         protected $hidden = [
29
         'password', 'remember_token',
         ];
32
         public function getJWTIdentifier()
            return $this->getKey();
         public function getJWTCustomClaims()
            return [];
```

• SS Hasil Postman

Hasil registrasi petugas

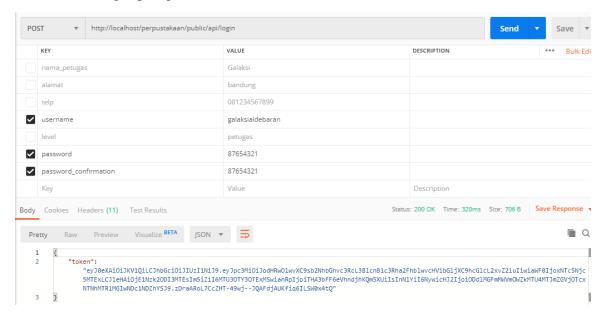
Hasil registrasi admin

```
Post white://localhost/perpustakaan/public/api/register

Body Cookies Headers (11) Test Results

Status: 201 Created Time: 422ms Size: 906 B Save Response of the control o
```

Hasil login petugas



Hasil login admin

