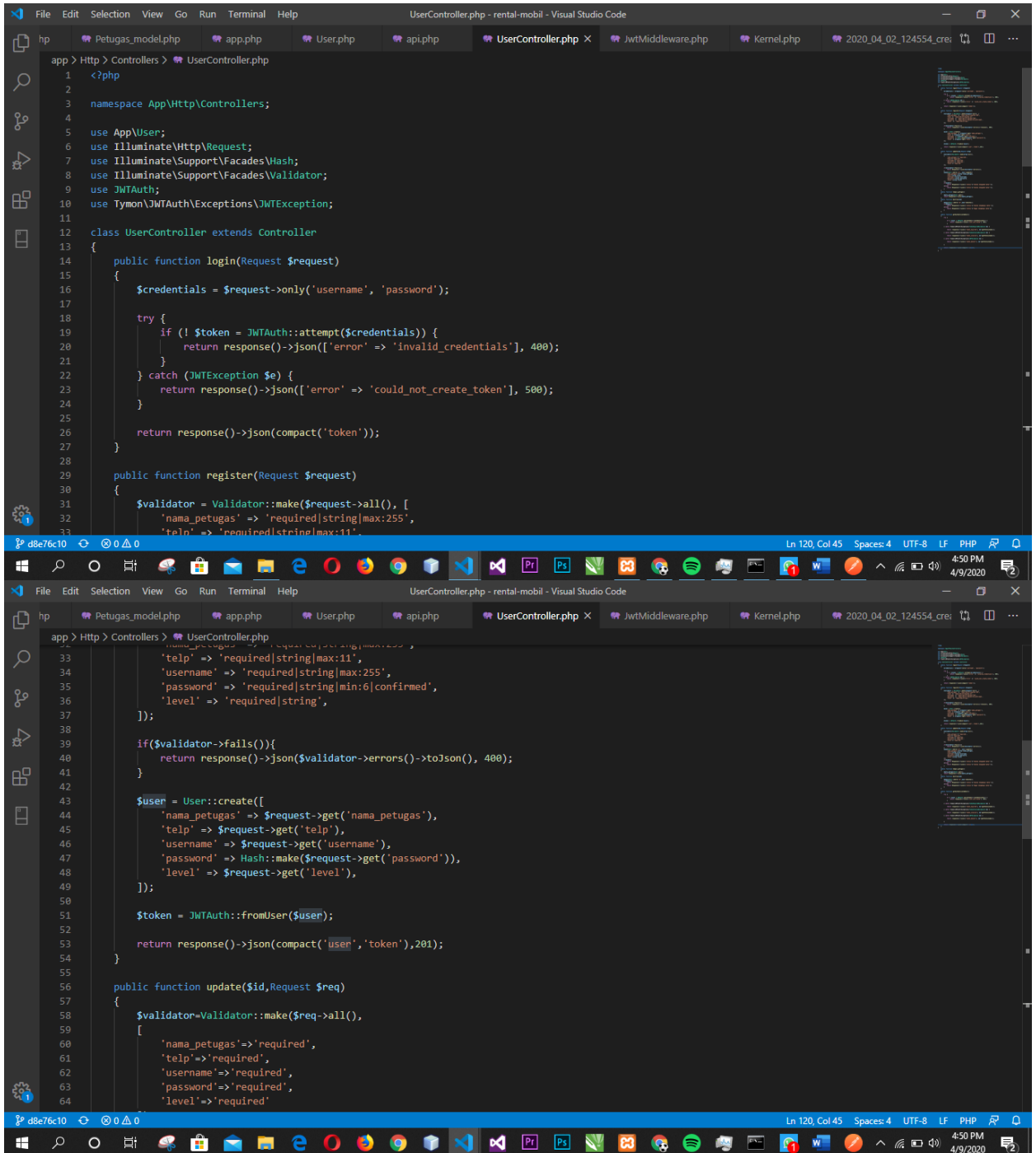


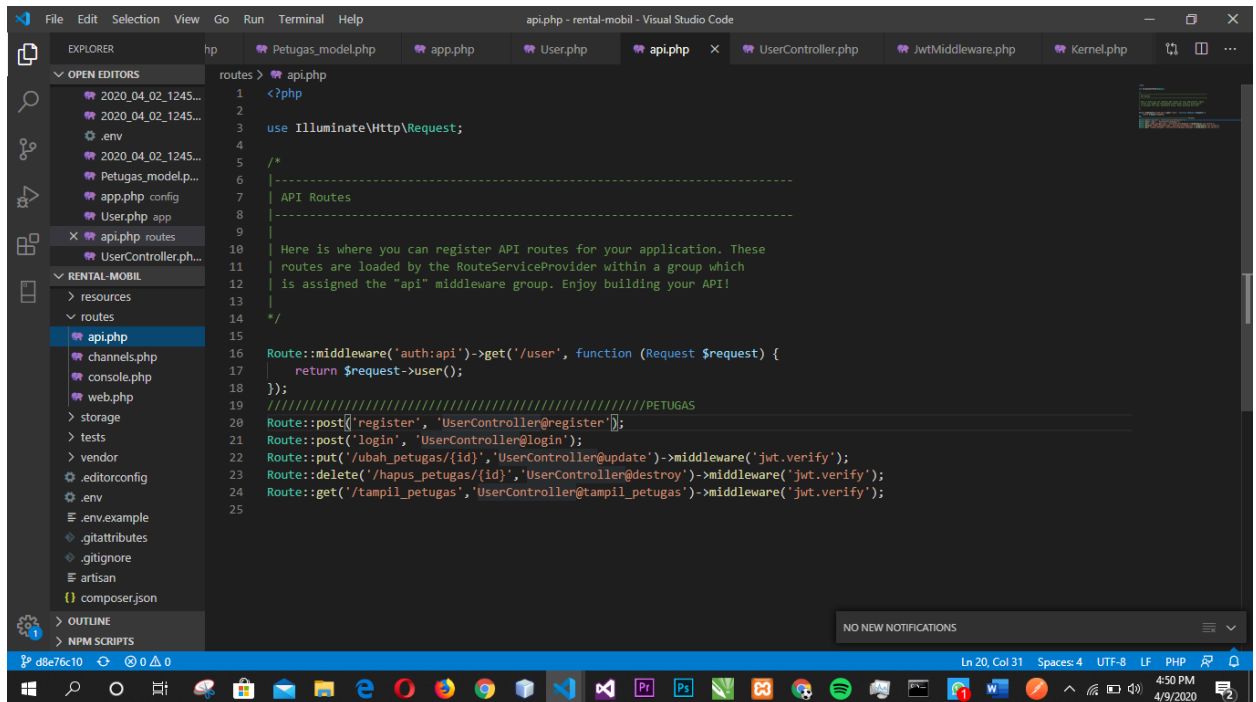
- Script Controller UserController.php



```
1 <?php
2
3 namespace App\Http\Controllers;
4
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 UserController 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             'telp' => 'required|string|max:11',
34             'username' => 'required|string|max:11',
35             'password' => 'required|string|min:6|confirmed',
36             'level' => 'required|string',
37         ]);
38
39         if($validator->fails()){
40             return response()->json($validator->errors()->toJson(), 400);
41         }
42
43         $user = User::create([
44             'nama_petugas' => $request->get('nama_petugas'),
45             'telp' => $request->get('telp'),
46             'username' => $request->get('username'),
47             'password' => Hash::make($request->get('password')),
48             'level' => $request->get('level'),
49         ]);
50
51         $token = JWTAuth::fromUser($user);
52
53         return response()->json(compact('user', 'token'), 201);
54     }
55
56     public function update($id, Request $req)
57     {
58         $validator = Validator::make($req->all(), [
59             'nama_petugas' => 'required',
60             'telp' => 'required',
61             'username' => 'required',
62             'password' => 'required',
63             'level' => 'required'
64         ]);
65     }
```

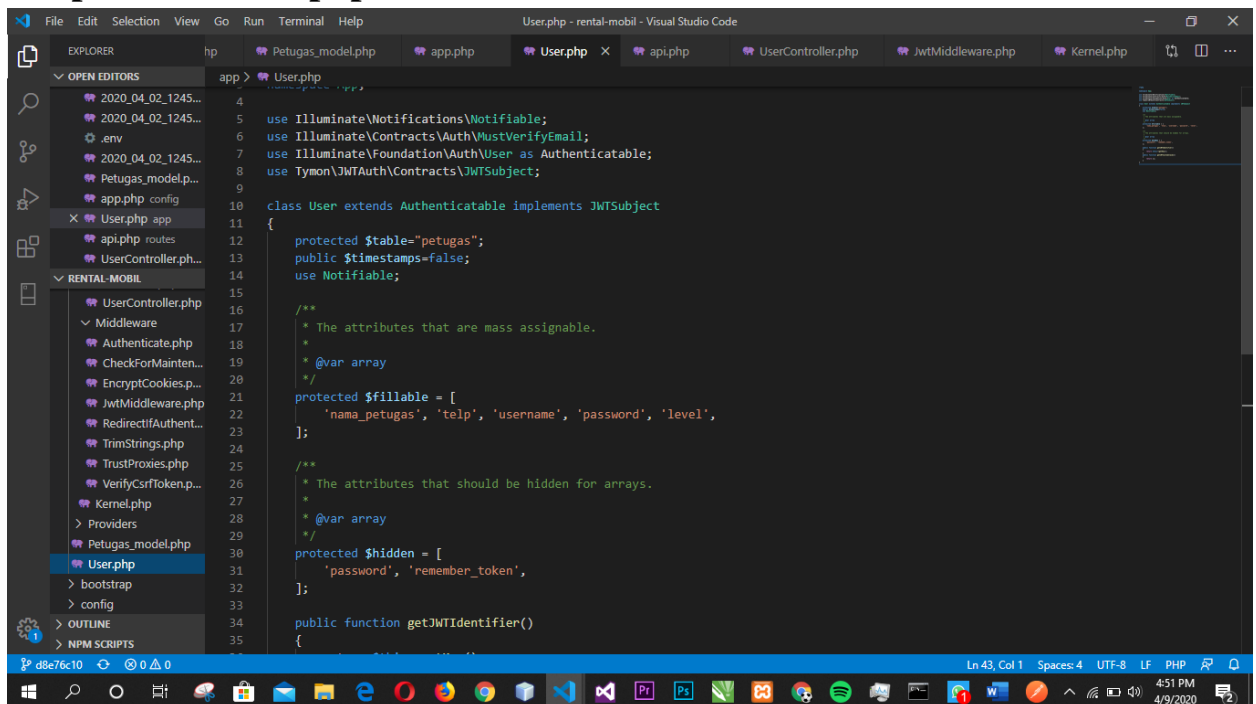
```
File Edit Selection View Go Run Terminal Help UserController.php - rental-mobil - Visual Studio Code
hp Petugas_model.php app.php User.php api.php UserController.php x JwtMiddleware.php Kernel.php 2020_04_02_124554_crei
app > Http > Controllers > UserController.php
62         'username'=>'required',
63         'password'=>'required',
64         'level'=>'required'
65     ]];
66
67     if($validator->fails()){
68         return Response()->json($validator->errors());
69     }
70     $ubah=User::where('id', $id)->update([
71         'nama_petugas'=>$req->nama_petugas,
72         'telp'=>$req->telp,
73         'username'=>$req->username,
74         'password'=>$req->password,
75         'level'=>$req->level
76     ]);
77     if($ubah){
78         return Response()->json(['status'=>'Sukses mengubah data!']);
79     }else{
80         return Response()->json(['status'=>'Sukses mengubah data!']);
81     }
82 }
83 public function tampil_petugas()
84 {
85     $data_petugas=User::get();
86     return Response()->json($data_petugas);
87 }
88 public function destroy($id)
89 {
90     $hapus=User::where('id',$id)->delete();
91     if($hapus){
92         return Response()->json(['status'=>'Sukses menghapus data!']);
93     }else{
94         return Response()->json(['status'=>'Gagal menghapus data!']);
95     }
96 }
97
98 public function getAuthenticatedUser()
99 {
100     try {
101
102         if (!$user = JWTAuth::parseToken()->authenticate()) {
103             return response()->json(['user_not_found'], 404);
104         }
105     } catch (Tymon\JWTAuth\Exceptions\TokenExpiredException $e) {
106
107         return response()->json(['token_expired'], $e->getStatusCode());
108     } catch (Tymon\JWTAuth\Exceptions\TokenInvalidException $e) {
109
110         return response()->json(['token_invalid'], $e->getStatusCode());
111     } catch (Tymon\JWTAuth\Exceptions\JWTException $e) {
112
113         return response()->json(['token_absent'], $e->getStatusCode());
114     }
115
116     return response()->json(compact('user'));
117 }
118
119
120
121
122
123
```

## • Script Routes Api.php



```
1 <?php
2
3 use Illuminate\Http\Request;
4
5 /*
6 |-----
7 | API Routes
8 |-----
9 |
10 | Here is where you can register API routes for your application. These
11 | routes are loaded by the RouteServiceProvider within a group which
12 | is assigned the "api" middleware group. Enjoy building your API!
13 |
14 | */
15
16 Route::middleware('auth:api')->get('/user', function (Request $request) {
17     return $request->user();
18 });
19
20 ///////////////////////////////////////////////////PETUGAS
21 Route::post('register', 'UserController@register');
22 Route::post('login', 'UserController@login');
23 Route::put('/ubah_petugas/{id}', 'UserController@update')->middleware('jwt.verify');
24 Route::delete('/hapus_petugas/{id}', 'UserController@destroy')->middleware('jwt.verify');
25 Route::get('/tampil_petugas', 'UserController@tampil_petugas')->middleware('jwt.verify');
```

## • Script Model User.php



```
4
5 use Illuminate\Notifications\Notifiable;
6 use Illuminate\Contracts\Auth\Authenticatable;
7 use Illuminate\Foundation\Auth\User as Authenticatable;
8 use Tymon\JWTAuth\Contracts\JWTSubject;
9
10 class User extends Authenticatable implements JWTSubject
11 {
12     protected $table='petugas';
13     public $timestamps=false;
14     use Notifiable;
15
16     /**
17      * The attributes that are mass assignable.
18      *
19      * @var array
20      */
21     protected $fillable = [
22         'nama_petugas', 'telp', 'username', 'password', 'level',
23     ];
24
25     /**
26      * The attributes that should be hidden for arrays.
27      *
28      * @var array
29      */
30     protected $hidden = [
31         'password', 'remember_token',
32     ];
33
34     public function getJWTIdentifier()
35     {
36     }
```

- Hasil Input Register dan Login

