



Nama : Dea Putri Nastiti
NIM : 2241720117
Kelas : TI 2H

JOBSHEET 10

RESTFUL API

Sebelumnya kita sudah membahas mengenai *authentication*, *authorization*, dan *middleware* pada Laravel. Dimana kita telah membuat fungsi login, register, logout, serta pemilihan role dan penerapan session pada halaman web. Pada pertemuan kali ini, kita akan mempelajari penerapan RESTFUL API di dalam project Laravel.

Sebelum kita masuk materi, kita buat dulu project baru yang akan kita gunakan untuk membangun aplikasi sederhana dengan topik *Point of Sales (PoS)*, sesuai dengan **Studi Kasus PWL.pdf**. Jadi kita bikin project Laravel 10 dengan nama **PWL_POS**.

Project PWL_POS akan kita gunakan sampai pertemuan 12 nanti, sebagai project yang akan kita pelajari

A. RESTFUL API

Representational State Transfer (REST) adalah gaya arsitektur perangkat lunak yang mendefinisikan seperangkat prinsip untuk merancang jaringan aplikasi terdistribusi. RESTful API adalah aplikasi pemrograman antarmuka yang mengikuti prinsip-prinsip REST untuk mentransfer data antara klien dan server.

RESTful API adalah salah satu arsitektur dalam API (*Application Program Interface*) yang menggunakan request HTTP untuk mengakses data. Data diakses dengan menggunakan HTTP method GET, PUT, POST dan DELETE yang merujuk pada operasi pembacaan, pembaruan, pembuatan dan penghapusan pada resource. Selain HTTP method, dalam RESTful atau REST digunakan juga HTTP response untuk mendefinisikan respon data yang dikembalikan. Format respon yang umum digunakan berupa JSON (Javascript Object Notation).



B. JSON Web Token (JWT)

JWT adalah singkatan dari JSON Web Token. Ini adalah standar terbuka (RFC 7519) yang mendefinisikan format token yang kompak dan mandiri untuk mentransfer klaim antara dua pihak. JWT sering digunakan dalam otentikasi dan pertukaran informasi yang aman di lingkungan yang tidak terpercaya, seperti internet.

JWT terdiri dari tiga bagian yang dipisahkan oleh titik ("."): header, payload, dan signature. Setiap bagian ini terdiri dari data JSON yang dienkripsi menggunakan algoritma tertentu dan kemudian disatukan untuk membentuk token yang lengkap. Header berisi jenis token dan tipe algoritma yang digunakan untuk enkripsi. Payload berisi klaim atau informasi yang ingin disampaikan. Signature digunakan untuk memverifikasi bahwa token belum berubah dan datanya berasal dari sumber yang dipercayai.

JWT sering digunakan dalam sistem otentikasi dan otorisasi modern, seperti aplikasi web dan layanan web API, karena fleksibilitasnya dalam menyampaikan informasi terenkripsi secara ringkas.

Kita dapat menggunakan JWT untuk:

- **Authentication**
Ketika pengguna melakukan authentication dan mendapatkan token, maka setiap permintaan berikutnya akan menyertakan token tersebut, dan memungkinkan pengguna untuk mengakses route, service, dan resources yang diizinkan.
- **Pertukaran informasi**
JSON Web Token adalah cara yang baik untuk mengirimkan informasi antar pihak dengan aman. Dengan token yang sudah ditandatangani dengan algoritma RSA, maka kita bisa tahu siapa yang melakukan request tersebut.

Berikut adalah cara kerja JWT :

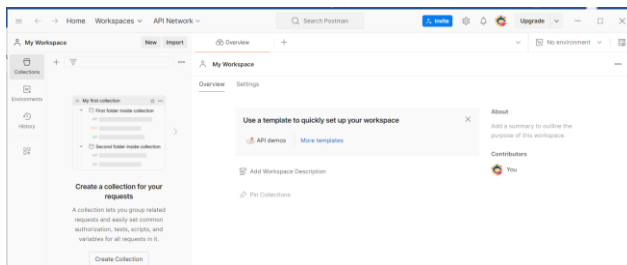
JWT (JSON Web Token) adalah cara untuk mentransfer informasi antara dua pihak secara aman sebagai objek JSON. Ini terdiri dari tiga bagian: header, payload, dan signature. Setelah pengguna berhasil autentikasi, server menghasilkan token JWT yang disematkan dalam permintaan HTTP. Server kemudian memvalidasi token untuk memberikan akses ke sumber daya yang diminta. Ini memberikan autentikasi yang aman dan stateless tanpa memerlukan penyimpanan status sesi di server.



Praktikum 1 – Membuat RESTful API Register

1. Sebelum memulai membuat REST API, terlebih dahulu download aplikasi Postman di <https://www.postman.com/downloads>.

Aplikasi ini akan digunakan untuk mengerjakan semua tahap praktikum pada Jobsheet ini.



2. Lakukan instalasi JWT dengan mengetikkan perintah berikut: `composer require tymon/jwt-auth:2.1.1` Pastikan Anda terkoneksi dengan internet.

```
PS C:\laragon\www\PWL_POS> composer require tymon/jwt-auth:2.1.1
https://repo.packagist.org could not be fully loaded (curl error 28 while downloading https://repo.packagist.org/packages.json: Failed to connect to repo.packagist.org port 443 after 10004 ms: Timeout was reached), package information was loaded from the local cache and may be out of date
./composer.json has been updated
Running composer update tymon/jwt-auth
Loading composer repositories with package information
Updating dependencies
Lock file operations: 4 installs, 0 updates, 0 removals
- Locking lcobucci/clock (2.2.0)
- Locking lcobucci/jwt (4.0.4)
- Locking stella-maris/clock (0.1.7)
```

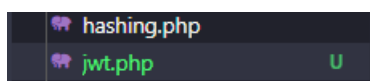
3. Setelah berhasil menginstall JWT, lanjutkan dengan publish konfigurasi file dengan perintah berikut:

```
php artisan vendor:publish --
provider="Tymon\JWTAuth\Providers\LaravelServiceProvider"
```

```
PS C:\laragon\www\PWL_POS> php artisan vendor:publish --
Which provider or tag's files would you like to publish?
All providers and tags ..... 0
Provider: illuminate\Foundation\Providers\FoundationServiceProvider ..... 1

INFO Publishing assets.
Copying file [C:\laragon\www\PWL_POS\vendor\tymon\jwt-auth\config\config.php] to [C:\laragon\www\PWL_POS\config\jwt.php] DONE
```

4. Jika perintah di atas berhasil, maka kita akan mendapatkan 1 file baru yaitu config/jwt.php. Pada file ini dapat dilakukan konfigurasi jika memang diperlukan.



5. Setelah itu jalankan perintah berikut untuk membuat secret key JWT. `php artisan jwt:secret`



Jika berhasil, maka pada file .env akan ditambahkan sebuah baris berisi nilai key JWT_SECRET.

```
PS C:\laragon\www\PWL_POS> php artisan jwt:secret  
jwt-auth secret [ppoZkUPYLq52m0cVa7dcRxfu5KXMaDUtezKNUb7xm1ELYbC91VLPb9q2LQtFNFq  
w] set successfully.
```

```
60  
61 JWT_SECRET=ppoZkUPYLq52m0cVa7dcRxfu5KXMaDUtezKNUb7xm1ELYbC91VLPb9q2LQtFNFqw  
62
```

6. Selanjutnya lakukan konfigurasi guard API. Buka config/auth.php. Ubah bagian ‘guards’ menjadi seperti berikut.

```
37  
38 'guards' => [  
39     'web' => [  
40         'driver' => 'session',  
41         'provider' => 'users',  
42     ],  
43     'api' => [  
44         'driver' => 'jwt',  
45         'provider' => 'users',  
46     ],  
47 ],
```

7. Kita akan menambahkan kode di model UserModel, ubah kode seperti berikut:

```
app > Models > UserModel.php > UserModel  
5 use Illuminate\Database\Eloquent\Factories\HasFactory;  
6 use Illuminate\Database\Eloquent\Model;  
7 use Illuminate\Database\Eloquent\Relations\BelongsTo;  
8 use Illuminate\Database\Eloquent\Relations\HasMany;  
9 use Tymon\JWTAuth\Contracts\JWTSubject;  
10 use Illuminate\Foundation\Auth\User as Authenticatable;  
11  
12 class UserModel extends Authenticatable implements JWTSubject  
13 {  
14     public function getJWTIdentifier()  
15     {  
16         return $this->getKey();  
17     }  
18  
19     public function getJWTCustomClaims()  
20     {  
21         return[];  
22     }  
23  
24     use HasFactory;  
25  
26     protected $table = 'm_user'; //Mendefinisikan nama tabel yang digunakan oleh model ini  
27     protected $primaryKey = 'user_id'; //mendefinisikan primary key dari tabel yang digunakan  
28     //protected $fillable = ['level_id', 'username', 'nama', 'password'];  
29     protected $fillable = ['level_id', 'username', 'nama', 'password'];
```

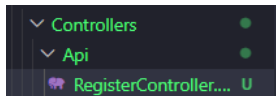
8. Berikutnya kita akan membuat controller untuk register dengan menjalankan perintah berikut.

```
php artisan make:controller Api/RegisterController
```

Jika berhasil maka akan ada tambahan controller pada folder Api dengan nama RegisterController.



```
PS C:\laragon\www\PWL_POS> php artisan make:controller Api/RegisterController  
INFO: Controller [C:\laragon\www\PWL_POS\app\Http\Controllers\Api/RegisterController.php] created successfully.
```



9. Buka file tersebut, dan ubah kode menjadi seperti berikut.

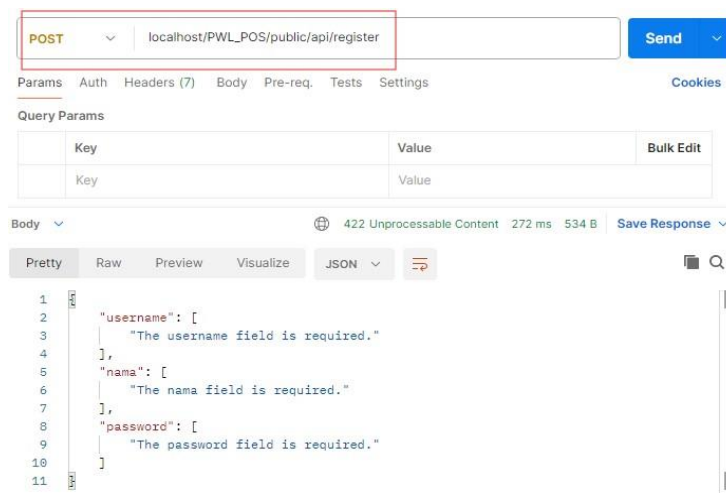
```
app > Http > Controllers > Api > RegisterController.php > RegisterController > __invoke  
1  <?php  
2  
3  namespace App\Http\Controllers\Api;  
4  
5  use App\Http\Controllers\Controller;  
6  use App\Models\UserModel;  
7  use Illuminate\Http\Request;  
8  use Illuminate\Support\Facades\Validator;  
9  
10 class RegisterController extends Controller  
11 {  
12     public function __invoke(Request $request)  
13     {  
14         //set validation  
15         $validator = Validator::make($request->all(), [  
16             'username' => 'required',  
17             'nama' => 'required',  
18             'password' => 'required|min:5|confirmed',  
19             'level_id' => 'required',  
20         ]);  
21  
22         //if validations fails  
23         if($validator->fails()){  
24             return response()->json($validator->errors(), 422);  
25         }  
26  
27         //create user  
28         $user = UserModel::create([  
29             'username' => $request->username,  
30             'nama' => $request->nama,  
31             'password' => bcrypt($request->password),  
32             'level_id' => $request->level_id,  
33         ]);  
34  
35         //return response JSON user is created  
36         if($user){  
37             return response()->json([  
38                 'success' => true,  
39                 'user' => $user,  
40             ], 201);  
41         }  
42  
43         //return JSON process insert failed  
44         return response()->json([  
45             'succes' => false,  
46         ], 409);  
47     }  
48 }
```

10. Selanjutnya buka routes/api.php, ubah semua kode menjadi seperti berikut.



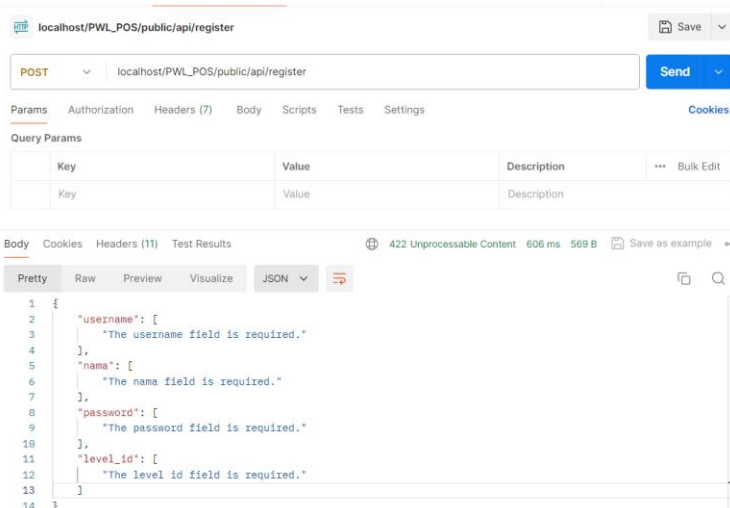
```
routes > apiphp
1  <?php
2
3  use App\Http\Controllers\Api\RegisterController;
4  use Illuminate\Http\Request;
5  use Illuminate\Support\Facades\Route;
6
7  /*
8  |-----
9  | API Routes
10 |-----
11 |
12 | Here is where you can register API routes for your application. These
13 | routes are loaded by the RouteServiceProvider and all of them will
14 | be assigned to the "api" middleware group. Make something great!
15 |
16 | */
17
18 Route::post('/register', App\Http\Controllers\Api\RegisterController::class)->name('register');
```

11. Jika sudah, kita akan melakukan uji coba REST API melalui aplikasi Postman. Buka aplikasi Postman, isi URL localhost/PWL_POS/public/api/register serta method POST. Klik Send.



Jika berhasil akan muncul error validasi seperti gambar di atas.

Lakukan percobaan yang sama dan berikan screenshot hasil percobaan Anda.





12. Sekarang kita coba masukkan data. Klik tab Body dan pilih form-data. Isikan key sesuai dengan kolom data, serta isikan data registrasi menggunakan nilai yang Anda inginkan.

POST localhost/PWL_POS/public/api/register

Params Auth Headers (8) **Body** Pre-req. Tests Settings Cookies

form-data

Key	Value
username	penggunasatu
nama	Pengguna 1
password	12345
password_confirmation	12345
level_id	2

Body Cookies Headers (11) Test Results 201 Created 624 ms 645 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "success": true,
3   "user": {
4     "username": "penggunasatu",
5     "name": "Pengguna 1",
6     "password": "$2y$12$Eb2SzV1sykIhYtYGTzH100VAKcK5p6EgnZnmbChkPcIu750Q3Ju",
7     "level_id": "2",
8     "updated_at": "2024-04-22T15:56:04.000000Z",
9     "created_at": "2024-04-22T15:56:04.000000Z",
10    "user_id": 17
11  }
12 }
```

Setelah klik tombol Send, jika berhasil maka akan keluar pesan sukses seperti gambar di atas.

Lakukan percobaan yang sama dan berikan screenshoot hasil percobaan Anda.

POST localhost/PWL_POS/public/api/register

Params Auth Headers (8) **Body** Scripts Settings Cookies

form-data

Key	Type	Value
username	Text	deaputri
nama	Text	dea putri
password	Text	123456
password_confirmation	Text	123456
level_id	Text	1
Key	Text	Value

201 Created 930 ms 630 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "success": true,
3   "user": {
4     "username": "deaputri",
5     "name": "dea putri",
6     "password": "$2y$12$sj5lfq98ZfrcJTt6KrlFa.nRmLCo37w6d1Y/v9/jyIIPqyKYUvov.",
7     "level_id": "1",
8     "updated_at": "2024-05-11T16:05:27.000000Z",
9     "created_at": "2024-05-11T16:05:27.000000Z",
10    "user_id": 18
11  }
12 }
```

13. Lakukan commit perubahan file pada Github.



Praktikum 2 – Membuat RESTful API Login

1. Kita buat file controller dengan nama LoginController. `php artisan make:controller Api/LoginController`

Jika berhasil maka akan ada tambahan controller pada folder Api dengan nama LoginController.

```
PS C:\laragon\www\PMI_POS>
php artisan make:controller Api/LoginController
[INFO] Controller [C:\laragon\www\PMI_POS\app\Http\Controllers\Api>LoginController.php] created successfully
```

2. Buka file tersebut, dan ubah kode menjadi seperti berikut.

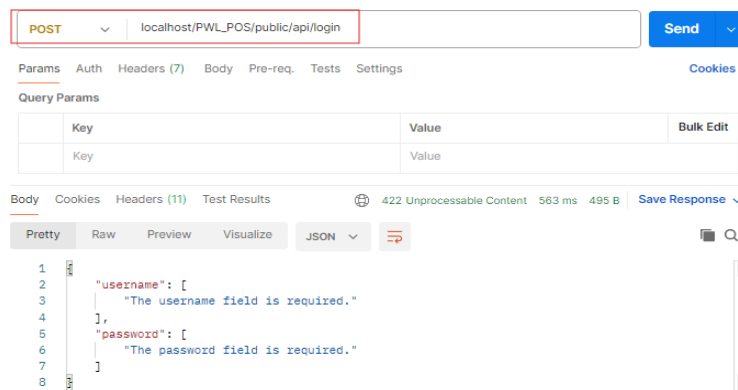
```
app > Http > Controllers > Api > LoginController.php > LoginController > __invoke
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use App\Http\Controllers\Controller;
6  use Illuminate\Http\Request;
7  use Illuminate\Support\Facades\Validator;
8
9  class LoginController extends Controller
10 {
11     public function __invoke(Request $request)
12     {
13         //set validation
14         $validator = Validator::make($request->all(),[
15             'username' => 'required',
16             'password' => 'required',
17         ]);
18
19         //if validations fails
20         if($validator->fails()){
21             return response()->json($validator->errors(), 422);
22         }
23
24         //get credentials from request
25         $credentials = $request->only('username', 'password');
26
27         //if auth failed
28         if(!$token = auth()->guard('api')->attempt($credentials)){
29             return response()->json([
30                 'succes' => false,
31                 'message' => 'Username atau Password anda salah'
32             ], 401);
33         }
34
35         //if auth succes
36         return response()->json([
37             'succes' => true,
38             'user' => auth()->guard('api')->user(),
39             'token' => $token
40         ], 200);
41     }
42 }
```




3. Berikutnya tambahkan route baru pada file api.php yaitu /login dan /user.

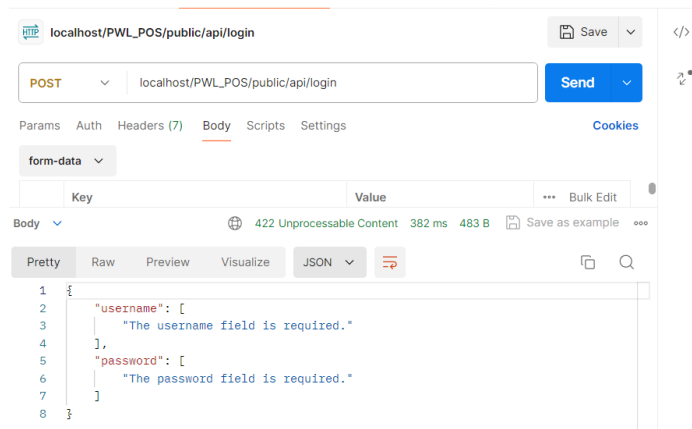
```
18 Route::post('/register', App\Http\Controllers\Api\RegisterController::class)->name('register');
19 Route::post('/login', App\Http\Controllers\Api>LoginController::class)->name('login');
20 Route::middleware('auth:api')->get('/user', function(Request $request){
21     return $request->user();
22 });
```

4. Jika sudah, kita akan melakukan uji coba REST API melalui aplikasi Postman. Buka aplikasi Postman, isi URL localhost/PWL_POS/public/api/login serta method POST. Klik Send.



Jika berhasil akan muncul error validasi seperti gambar di atas.

Lakukan percobaan yang sama dan berikan screenshoot hasil percobaan Anda.



5. Selanjutnya, isikan username dan password sesuai dengan data user yang ada pada database. Klik tab Body dan pilih form-data. Isikan key sesuai dengan kolom data, serta isikan data user. Klik tombol Send, jika berhasil maka akan keluar tampilan seperti berikut. Copy nilai token yang diperoleh pada saat login karena akan diperlukan pada saat logout.



POST localhost/PWL_POS/public/api/login Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary

Key	Value
username	penggunasatu
password	12345

body Cookies Headers (11) Test Results Status: 200 OK Time: 1501 ms Size: 986 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "success": true,
3   "user": {
4     "user_id": 17,
5     "level_id": 2,
6     "username": "penggunasatu",
7     "nama": "Pengguna 1",
8     "password": "$2y$12$Eb2Sv1j5yKINytYGTzHi00VAKcK5p6EgnZnmBChkP1cIu750Q1Ju",
9     "created_at": "2024-04-22T15:56:04.000000Z",
10    "updated_at": "2024-04-22T15:56:04.000000Z"
11  },
12  "token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOi8vbG9jYXRob3N0L1BXTF9QT1MvchVibGljL2FwaS9sb2dpbiIsImhhdCI6MTcxNTU0MTcwMTwzX2h1IjoxNzE1NTg1MzAyLCJyYmYiOiJlbnR1eSIsImVudCI6IjIwMjQwNDIyMTU1NTU0MTcwMTwzX2h1Ij09.p3VgYJD0e2iqF94Y-jLyeUBC9VTiwG67qTCJnu3T0goI"
13 }
```

Lakukan percobaan yang sama dan berikan screenshoot hasil percobaan Anda.

POST localhost/PWL_POS/public/api/login Send

Params Auth Headers (8) Body Scripts Settings Cookies

form-data

Key	Value
username	deaputri
password	123456

Body 200 OK 7.05 s 964 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "succes": true,
3   "user": {
4     "user_id": 18,
5     "level_id": 1,
6     "username": "deaputri",
7     "nama": "dea putri",
8     "password": "$2y$12$sjlfq98ZFrcJTt6KrlFa.nRmLC037w6d1V/v9/jyIIPqYKYUvov.",
9     "created_at": "2024-05-11T16:05:27.000000Z",
10    "updated_at": "2024-05-11T16:05:27.000000Z"
11  },
12  "token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOi8vbG9jYXRob3N0L1BXTF9QT1MvchVibGljL2FwaS9sb2dpbiIsImhhdCI6MTcxNTU0MTcwMTwzX2h1IjoxNzE1NTg1MzAyLCJyYmYiOiJlbnR1eSIsImVudCI6IjIwMjQwNDIyMTU1NTU0MTcwMTwzX2h1Ij09.p3VgYJD0e2iqF94Y-jLyeUBC9VTiwG67qTCJnu3T0goI"
13 }
```

Token:

"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOi8vbG9jYXRob3N0L1BXTF9QT1MvchVibGljL2FwaS9sb2dpbiIsImhhdCI6MTcxNTU0MTcwMTwzX2h1IjoxNzE1NTg1MzAyLCJyYmYiOiJlbnR1eSIsImVudCI6IjIwMjQwNDIyMTU1NTU0MTcwMTwzX2h1Ij09.p3VgYJD0e2iqF94Y-jLyeUBC9VTiwG67qTCJnu3T0goI"



6. Lakukan percobaan yang untuk data yang salah dan berikan screenshoot hasil percobaan Anda.

POST localhost/PWL_POS/public/api/login Send

Params Auth Headers (8) Body Scripts Settings Cookies

form-data

	Key		Value	...	Bulk Edit
<input checked="" type="checkbox"/>	username	Text	dea		
<input checked="" type="checkbox"/>	password	Text	12345		
	Key	Text	Value		

ody 401 Unauthorized 845 ms 441 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "succes": false,
3   "message": "Username atau Password anda salah"
4 }
```

7. Coba kembali melakukan login dengan data yang benar. Sekarang mari kita coba menampilkan data user yang sedang login menggunakan URL localhost/PWL_POS/public/api/user dan method GET. Jelaskan hasil dari percobaan tersebut.

GET localhost/PWL_POS/public/api/user?token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2Vybm9udGVudCI6ImRlYSB1dG8iLCJle... Send

Params Auth Headers (7) Body Scripts Settings Cookies

none

This request does not have a body

ody 200 OK 750 ms 599 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "user_id": 18,
3   "level_id": 1,
4   "username": "deaputri",
5   "nama": "dea putri",
6   "password": "$2y$12$SjSlfq98ZFrCJTt6KxIFa.nRmLCo37w6d1Y/v9/jyIIPQyKYUvov.",
7   "created_at": "2024-05-11T16:05:27.000000Z",
8   "updated_at": "2024-05-11T16:05:27.000000Z"
9 }
```

Pada url diberi tambahan token yang di dapat dari percobaan 5, sehingga akan memunculkan data user yang sedang login.

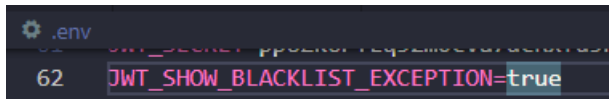
8. Lakukan commit perubahan file pada Github.



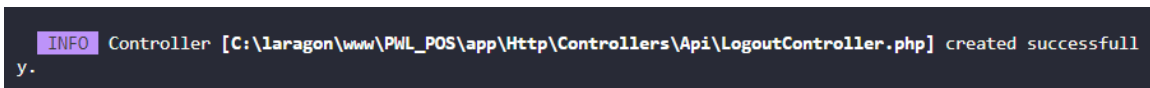
Praktikum 3 – Membuat RESTful API Logout

1. Tambahkan kode berikut pada file .env

`JWT_SHOW_BLACKLIST_EXCEPTION=true`



2. Buat Controller baru dengan nama LogoutController. `php artisan make:controller Api/LogoutController`



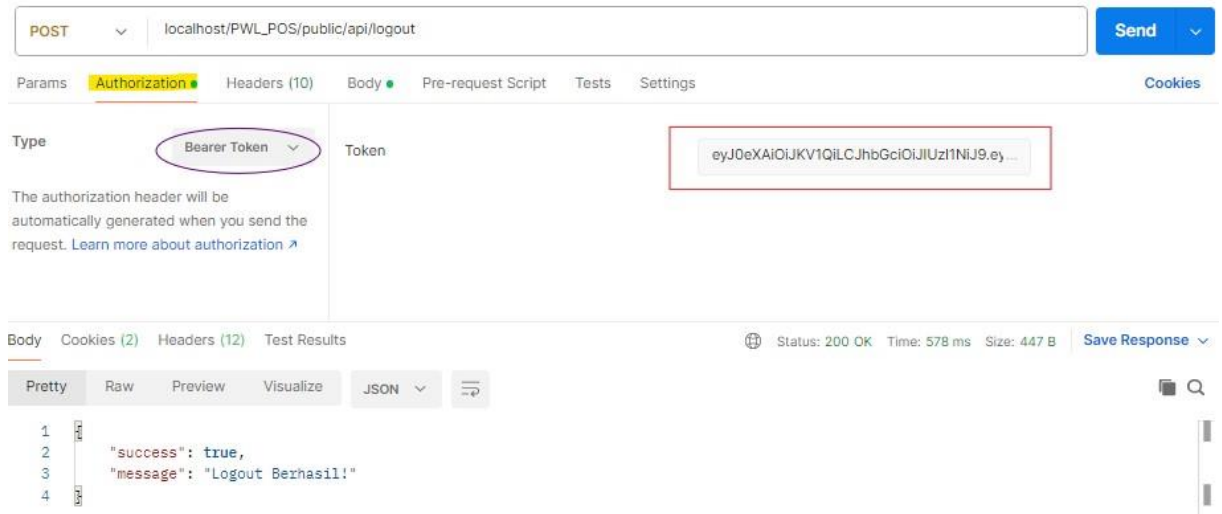
3. Buka file tersebut dan ubah kode menjadi seperti berikut.

```
app > Http > Controllers > Api > LogoutController.php > LogoutController > _invoke
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use App\Http\Controllers\Controller;
6  use Illuminate\Http\Request;
7  use Tymon\JWTAuth\Facades\JWTAuth;
8  use Tymon\JWTAuth\Exceptions\JWTException;
9  use Tymon\JWTAuth\Exceptions\TokenExpiredException;
10 use Tymon\JWTAuth\Exceptions\TokenInvalidException;
11
12 class LogoutController extends Controller
13 {
14     public function _invoke(Request $request){
15         //remove token
16         $removeToken = JWTAuth::invalidate(JWTAuth::getToken());
17
18         if($removeToken){
19             //return response JSON
20             return response()->json([
21                 'succes' => true,
22                 'message' => 'Logout Berhasil',
23             ]);
24         }
25     }
26 }
```

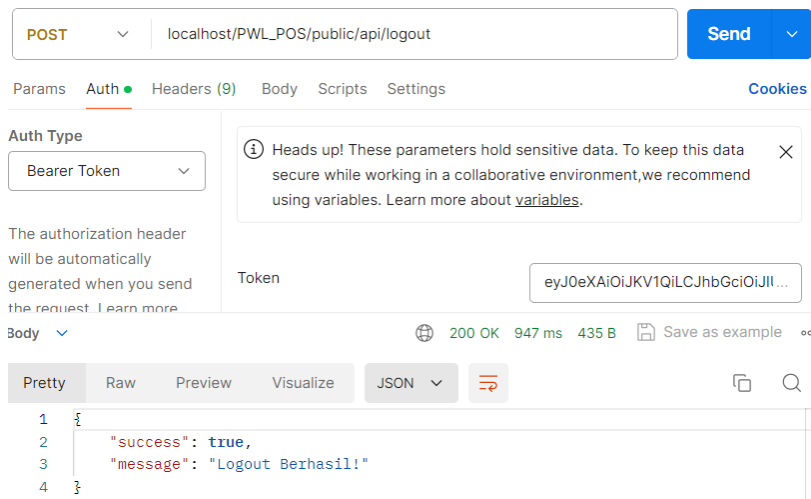
4. Lalu kita tambahkan routes pada api.php

```
routes > api.php > ...
23 | Route::post('/logout', App\Http\Controllers\Api\LogoutController::class)->name('logout');
```

5. Jika sudah, kita akan melakukan uji coba REST API melalui aplikasi Postman. Buka aplikasi Postman, isi URL `localhost/PWL_POS/public/api/logout` serta method POST.
6. Isi token pada tab Authorization, pilih Type yaitu Bearer Token. Isikan token yang didapat saat login. Jika sudah klik Send.



Lakukan percobaan yang sama dan berikan screenshot hasil percobaan Anda.



7. Lakukan commit perubahan file pada Github.

Praktikum 4 – Implementasi CRUD dalam RESTful API

Pada praktikum ini kita akan menggunakan tabel m_level untuk dimodifikasi menggunakan RESTful API.

1. Pertama, buat controller untuk mengolah API pada data level.

`php artisan make:controller Api/LevelController`

```
PS C:\laragon\www\PWL_POS> php artisan make:controller Api/LevelController  
  
INFO Controller [C:\laragon\www\PWL_POS\app\Http\Controllers\Api\LevelController.php] created successfully
```



- Setelah berhasil, buka file tersebut dan tuliskan kode seperti berikut yang berisi fungsi CRUDnya.

```
app > Http > Controllers > Api > LevelController.php > LevelController > destroy
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use App\Http\Controllers\Controller;
6  use Illuminate\Http\Request;
7  use App\Models\LevelModel;
8
9  class LevelController extends Controller
10 {
11     public function index(){
12         return LevelModel::all();
13     }
14
15     public function store(Request $request){
16         $level = LevelModel::create($request->all());
17         return response()->json($level, 201);
18     }
19
20     public function show(LevelModel $level){
21         return LevelModel::find($level);
22     }
23
24     public function update(Request $request, LevelModel $level){
25         $level->update($request->all());
26         return LevelModel::find($level);
27     }
28
29     public function destroy(LevelModel $level){
30         $level->delete();
31
32         return response()->json([
33             'succes' => true,
34             'message' => 'Data terhapus',
35         ]);
36     }
37 }
```

- Kemudian kita lengkapi routes pada api.php.

```
26 Route::get('levels', [LevelController::class, 'index']);
27 Route::post('levels', [LevelController::class, 'store']);
28 Route::get('levels/{level}', [LevelController::class, 'show']);
29 Route::put('levels/{level}', [LevelController::class, 'update']);
30 Route::delete('levels/{level}', [LevelController::class, 'destroy']);
```

- Jika sudah. Lakukan uji coba API mulai dari fungsi untuk menampilkan data. Gunakan URL: `localhost/PWL_POS-main/public/api/levels` dan method GET. **Jelaskan dan berikan screenshoot hasil percobaan Anda.**



GET localhost/PWL_POS/public/api/levels

Params Auth Headers (8) Body Scripts Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
-----	-------	-------------	-----------

body 200 OK 1443 ms 681 B Save as example

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "level_id": 1,
4     "level_kode": "ADM",
5     "level_nama": "Administrator",
6     "created_at": null,
7     "updated_at": null
8   },
9   {
10    "level_id": 2,
11    "level_kode": "MNG",
12    "level_nama": "Manager",
13    "created_at": null,
14    "updated_at": null
15  },
16  {
17    "level_id": 3,
18    "level_kode": "STF",
19    "level_nama": "Staff/Kasir",
20    "created_at": null,
21    "updated_at": null
22  }
23 ]
```

Mengambil dan menampilkan data yang ada pada level di database.

5. Kemudian, lakukan percobaan penambahan data dengan URL :
localhost/PWL_POSmain/public/api/levels dan method POST seperti di bawah ini.

POST localhost/PWL_POSmain/public/api/levels

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

☐ none ☒ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

Key	Value	Description	Bulk Edit
<input checked="" type="checkbox"/> level_kode	Text SPV		
<input checked="" type="checkbox"/> level_nama	Text Supervisor		
Key	Text Value	Description	

Body Cookies Headers (11) Test Results Status: 201 Created Time: 276 ms Size: 531 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "level_kode": "SPV",
3   "level_nama": "Supervisor",
4   "updated_at": "2024-04-22T21:40:32.000000Z",
5   "created_at": "2024-04-22T21:40:32.000000Z",
6   "level_id": 4
7 }
```

Jelaskan dan berikan screenshot hasil percobaan Anda.



POST localhost/PWL_POS/public/api/levels Send

Params Auth Headers (10) Body Scripts Settings Cookies

form-data

Key	Value	...	Bulk Edit
<input checked="" type="checkbox"/> level_kode	Text SPV		
<input checked="" type="checkbox"/> level_nama	Text Supervisor		
Key	Text Value		

Body 201 Created 20.86 s 540 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "level_kode": "SPV",
3   "level_nama": "Supervisor",
4   "updated_at": "2024-05-13T07:19:36.000000Z",
5   "created_at": "2024-05-13T07:19:36.000000Z",
6   "level_id": 7
7 }
```

☐ Edit Copy Delete 7 SPV Supervisor 2024-05-13 07:19:36 2024-05-13 07:19:36

Melakukan penambahan data pada level

6. Berikutnya lakukan percobaan menampilkan detail data. **Jelaskan dan berikan screenshot hasil percobaan Anda.**

GET localhost/PWL_POS/public/api/levels/7 Send

Params Auth Headers (10) Body Scripts Settings Cookies

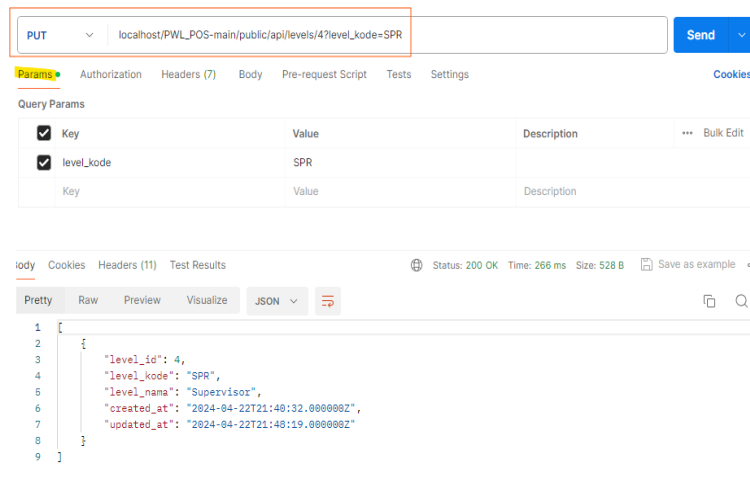
body 200 OK 932 ms 537 B Save as example

Pretty Raw Preview Visualize JSON

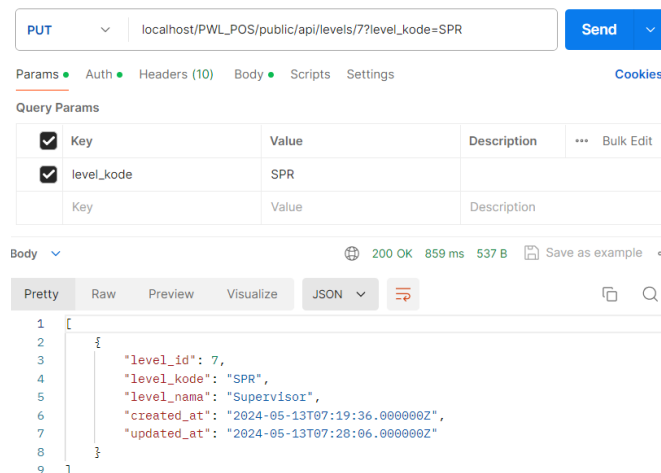
```
1 [
2   {
3     "level_id": 7,
4     "level_kode": "SPV",
5     "level_nama": "Supervisor",
6     "created_at": "2024-05-13T07:19:36.000000Z",
7     "updated_at": "2024-05-13T07:19:36.000000Z"
8   }
9 ]
```

Menampilkan detail data dengan level_id 7 menggunakan method get untuk mengambil/mendapatkan data

7. Jika sudah, kita coba untuk melakukan edit data menggunakan localhost/PWL_POSmain/public/api/levels/{id} dan method PUT. Isikan data yang ingin diubah pada tab Param.

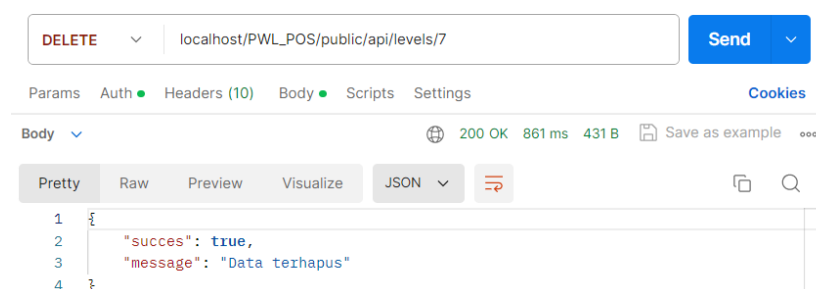


Jelaskan dan berikan screenshoot hasil percobaan Anda.



Melakukan perubahan / update data menggunakan method PUT.

8. Terakhir lakukan percobaan hapus data. Jelaskan dan berikan screenshoot hasil percobaan Anda.



Menghapus data dengan level_id 7 menggunakan method delete

9. Lakukan commit perubahan file pada Github.



TUGAS

Implementasikan CRUD API pada tabel lainnya yaitu tabel m_user, m_kategori, dan m_barang

1. m_user

```
PS C:\laragon\www\PWL_POS> php artisan make:controller Api/UserController
```

```
INFO Controller [C:\laragon\www\PWL_POS\app\Http\Controllers\Api\UserController.php] created successfully.
```

```
app > Http > Controllers > Api > UserController.php > UserController > store
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use App\Http\Controllers\Controller;
6  use Illuminate\Http\Request;
7  use App\Models\UserModel;
8
9  class UserController extends Controller
10 {
11     public function index(){
12         return UserModel::all();
13     }
14
15     public function store(Request $request){
16         $user = UserModel::create([
17             'level_id' => $request->level_id,
18             'username' => $request->username,
19             'nama' => $request->nama,
20             'password' => bcrypt($request->password),
21         ]);
22         return response()->json($user, 201);
23     }
24
25     public function show(UserModel $user){
26         return UserModel::find($user);
27     }
28
29     public function update(Request $request, UserModel $user){
30         $user->update($request->all());
31         return UserModel::find($user);
32     }
33
34     public function destroy(UserModel $user){
35         $user->delete();
36
37         return response()->json([
38             'succes' => true,
39             'message' => 'Data terhapus',
40         ]);
41     }
42 }
```

```
33 //user
34 Route::get('users', [UserController::class, 'index']);
35 Route::post('users', [UserController::class, 'store']);
36 Route::get('users/{user}', [UserController::class, 'show']);
37 Route::put('users/{user}', [UserController::class, 'update']);
38 Route::delete('users/{user}', [UserController::class, 'destroy']);
```



- Menampilkan data

GET localhost/PWL_POS/public/api/users Send

Params Auth Headers (8) Body Scripts Settings Cookies

Body 200 OK 547 ms 3.37 KB Save as example

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "user_id": 1,
4     "level_id": 1,
5     "username": "admin",
6     "nama": "Administrator",
7     "password": "$2y$12$uRlgTycCq1YK4.
8       kLTs3TteeihTs0PjEe0p8mUtuDWiWAGGHmx46",
9     "created_at": null,
10    "updated_at": null
11  },
12  {
13    "user_id": 2,
14    "level_id": 2,
15    "username": "manager",
16    "nama": "Manager",
17    "password": "$2y$12$4MSb1/4wLM/oNALc8dzZB09w1SXVge0gNyGnaQNkkw7wKUpv/
18      i9i0",
19    "created_at": null,
20    "updated_at": null
21  },
22  {
23    "user_id": 3,
24    "level_id": 3,
25    "username": "staff",
26    "nama": "Staff/Kasir",
27    "password": "$2y$12$FFtILtLca9k3XoTIwCda/u15XLfo9nRBGLdPswMny2Hcs.
28      qnubx2q",
29    "created_at": null,
```

- Menambah data

POST localhost/PWL_POS/public/api/users Send

Params Auth Headers (10) Body Scripts Settings Cookies

form-data

Key	Value
<input checked="" type="checkbox"/> level_id	2
<input checked="" type="checkbox"/> username	managerdea
<input checked="" type="checkbox"/> nama	Manager Dea
<input checked="" type="checkbox"/> password	12345
Key	Value

Body 201 Created 1391 ms 629 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "level_id": "2",
3   "username": "managerdea",
4   "nama": "Manager Dea",
5   "password": "$2y$12$WoQH.Y039XJYI6MJZpPN5.FYfllzeFj3qC1RAY.joN6gXj7NsQVna",
6   "updated_at": "2024-05-13T07:47:50.000000Z",
7   "created_at": "2024-05-13T07:47:50.000000Z",
8   "user_id": 20
9 }
```



- Mengedit data

PUT localhost/PWL_POS/public/api/users/20?level_id=1&nama=Admin Dea Send

Params Auth Headers (9) Body Scripts Settings Cookies

Query Params

<input checked="" type="checkbox"/>	Key	Value	Description	...	Bulk Edit
<input checked="" type="checkbox"/>	level_id	1			
<input checked="" type="checkbox"/>	nama	Admin Dea			

```
{
  "user_id": 20,
  "level_id": 1,
  "username": "manageridea",
  "nama": "Admin Dea",
  "password": "$2y$12$WQH.Y039XJYI6MJZpPN5.FYfillzeFj3qCiRAY.
  joN6gXj7NsQVna",
  "created_at": "2024-05-13T07:47:50.000000Z",
  "updated_at": "2024-05-13T07:51:21.000000Z"
}
```

- Menghapus data

DELETE localhost/PWL_POS/public/api/users/20 Send

Params Auth Headers (8) Body Scripts Settings Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
--	-----	-------	-------------	-----	-----------

Body 200 OK 842 ms 431 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "succes": true,
3   "message": "Data terhapus"
4 }
```

2. m_kategori

```
PS C:\laragon\www\PWL_POS> php artisan make:controller Api/KategoriController

INFO Controller [C:\laragon\www\PWL_POS\app\Http\Controllers\Api\KategoriController.php] created successfully.

app > Http > Controllers > Api > KategoriController.php > ...
1 <?php
2
3 namespace App\Http\Controllers\Api;
4
5 use App\Http\Controllers\Controller;
6 use Illuminate\Http\Request;
7 use App\Models\KategoriModel;
8
9 class KategoriController extends Controller
10 {
11     public function index(){
12         return KategoriModel::all();
13     }
14
15     public function store(Request $request){
16         $kategori = KategoriModel::create($request->all());
17         return response()->json($kategori, 201);
18     }
19 }
```



```
19
20 public function show(KategoriModel $kategori){
21     return KategoriModel::find($kategori);
22 }
23
24 public function update(Request $request, KategoriModel $kategori){
25     $kategori->update($request->all());
26     return KategoriModel::find($kategori);
27 }
28
29 public function destroy(KategoriModel $kategori){
30     $kategori->delete();
31
32     return response()->json([
33         'succes' => true,
34         'message' => 'Data terhapus',
35     ]);
36 }
37 }

42 //kategori
43 Route::get('kategoris', [KategoriController::class, 'index']);
44 Route::post('kategoris', [KategoriController::class, 'store']);
45 Route::get('kategoris/{kategori}', [KategoriController::class, 'show']);
46 Route::put('kategoris/{kategori}', [KategoriController::class, 'update']);
47 Route::delete('kategoris/{kategori}', [KategoriController::class, 'destroy']);
```

- Menampilkan data

GET localhost/PWL_POS/public/api/kategoris Send

Params Auth Headers (8) Body Scripts Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
-----	-------	-------------	-----------

200 OK 879 ms 1.04 KB Save as example

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "kategori_id": 1,
4     "kategori_kode": "k1",
5     "kategori_nama": "Makanan",
6     "created_at": null,
7     "updated_at": null
8   },
9   {
10    "kategori_id": 2,
11    "kategori_kode": "k2",
12    "kategori_nama": "Minuman",
13    "created_at": null,
14    "updated_at": null
15  },
16  {
17    "kategori_id": 3,
18    "kategori_kode": "k3",
19    "kategori_nama": "Sembako",
20    "created_at": null,
21    "updated_at": null
22  },
23 ]
```



- Menambah data

POST localhost/PWL_POS/public/api/kategoris **Send**

Params Auth Headers (10) **Body** Scripts Settings Cookies

form-data

Key	Value
<input checked="" type="checkbox"/> kategori_kode	Text k8
<input checked="" type="checkbox"/> kategori_nama	Text Skincare
Key	Value

201 Created 736 ms 547 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "kategori_kode": "k8",
3   "kategori_nama": "Skincare",
4   "updated_at": "2024-05-13T08:11:14.000000Z",
5   "created_at": "2024-05-13T08:11:14.000000Z",
6   "kategori_id": 13
7 }
```

- Mengedit data

PUT localhost/PWL_POS/public/api/kategoris/13?kategori_nama=Lego **Send**

Params Auth Headers (10) **Body** Scripts Settings Cookies

Query Params

Key	Value	Description
<input checked="" type="checkbox"/> kategori_nama	Lego	
Key	Value	Description

200 OK 575 ms 540 B Save as example

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "kategori_id": 13,
4     "kategori_kode": "k8",
5     "kategori_nama": "Lego",
6     "created_at": "2024-05-13T08:11:14.000000Z",
7     "updated_at": "2024-05-13T08:12:21.000000Z"
8   }
9 ]
```

- Menghapus data

DELETE localhost/PWL_POS/public/api/kategoris/13 **Send**

Params Auth Headers (10) **Body** Scripts Settings Cookies

Query Params

Key	Value	Description
Key	Value	Description

200 OK 694 ms 431 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "succes": true,
3   "message": "Data terhapus"
4 }
```



3. m_barang

```
PS C:\laragon\www\PWL_POS> php artisan make:controller Api\BarangController

[INFO] Controller [C:\laragon\www\PWL_POS\app\Http\Controllers\Api\BarangController.php] created successfully.
```

```
app > Http > Controllers > Api > BarangController.php > BarangController
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use App\Http\Controllers\Controller;
6  use Illuminate\Http\Request;
7  use App\Models\BarangModel;
8
9  class BarangController extends Controller
10 {
11     public function index(){
12         return BarangModel::all();
13     }
14
15     public function store(Request $request){
16         $barang = BarangModel::create($request->all());
17         return response()->json($barang, 201);
18     }
19
20     public function show(BarangModel $barang){
21         return BarangModel::find($barang);
22     }
23
24     public function update(Request $request, BarangModel $barang){
25         $barang->update($request->all());
26         return BarangModel::find($barang);
27     }
28
29     public function destroy(BarangModel $barang){
30         $barang->delete();
31         $barang->delete();
32         return response()->json([
33             'succes' => true,
34             'message' => 'Data terhapus',
35         ]);
36     }
37 }
```

```
49 //barang
50 Route::get('barangs', [BarangController::class, 'index']);
51 Route::post('barangs', [BarangController::class, 'store']);
52 Route::get('barangs/{barang}', [BarangController::class, 'show']);
53 Route::put('barangs/{barang}', [BarangController::class, 'update']);
54 Route::delete('barangs/{barang}', [BarangController::class, 'destroy']);
```



- Menampilkan data

GET localhost/PWL_POS/public/api/barangs Send

Params Auth Headers (8) Body Scripts Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
-----	-------	-------------	-----------

body 200 OK 1021 ms 1.82 KB Save as example

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "barang_id": 1,
4     "kategori_id": 1,
5     "barang_kode": "b1",
6     "barang_nama": "Sosis",
7     "harga_beli": 1000,
8     "harga_jual": 2000,
9     "created_at": null,
10    "updated_at": null
11  },
12  {
13    "barang_id": 2,
14    "kategori_id": 1,
15    "barang_kode": "b2",
16    "barang_nama": "Mi Goreng",
17    "harga_beli": 3000,
18    "harga_jual": 4000,
19    "created_at": null,
20    "updated_at": null
21  },
22  ]
```

- Menambah data

POST localhost/PWL_POS/public/api/barangs Send

Params Auth Headers (10) Body Scripts Settings Cookies

form-data

Key	Value	Bulk Edit
<input checked="" type="checkbox"/> kategori_id	Text 1	
<input checked="" type="checkbox"/> barang_kode	Text b11	
<input checked="" type="checkbox"/> barang_nama	Text Basreng	
<input checked="" type="checkbox"/> harga_beli	Text 4000	
<input checked="" type="checkbox"/> harga_jual	Text 5000	
Key	Text Value	

body 201 Created 769 ms 599 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "kategori_id": "1",
3   "barang_kode": "b11",
4   "barang_nama": "Basreng",
5   "harga_beli": "4000",
6   "harga_jual": "5000",
7   "updated_at": "2024-05-13T08:08:27.000000Z",
8   "created_at": "2024-05-13T08:08:27.000000Z",
9   "barang_id": 12
10 }
```




- Mengedit data

PUT localhost/PWL_POS/public/api/barangs/12?harga_jual=6000 Send

Params Auth Headers (10) Body Scripts Settings Cookies

Query Params

<input checked="" type="checkbox"/>	Key	Value	Description	...	Bulk Edit
<input checked="" type="checkbox"/>	harga_jual	6000			

```
{
  "barang_id": 12,
  "kategori_id": 1,
  "barang_kode": "b11",
  "barang_nama": "Basreng",
  "harga_beli": 4000,
  "harga_jual": 6000,
  "created_at": "2024-05-13T08:08:27.000000Z",
  "updated_at": "2024-05-13T08:13:46.000000Z"
}
```

- Menghapus data

DELETE localhost/PWL_POS/public/api/barangs/12 Send

Params Auth Headers (10) Body Scripts Settings Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
--	-----	-------	-------------	-----	-----------

body 200 OK 929 ms 431 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "succes": true,
3   "message": "Data terhapus"
4 }
```

*** Sekian, dan selamat belajar ***