

LAPORAN JOBSHEET 10

RESTFUL API

MATA KULIAH PEMROGRAMAN WEB LANJUT

Dosen Pengampu : Dimas Wahyu Wibowo, S.T., M.T.



Disusun oleh :

Dahniar Davina SIB-2A / 2341760023

JURUSAN TEKNOLOGI INFORMASI

PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS

POLITEKNIK NEGERI MALANG

2025

A. PRAKTIKUM 1 : Membuat RESTful API Register

1. Sebelum memulai membuat REST API, terlebih dahulu download aplikasi Postman

<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\PHL_2025\Minggu10\Jobsheet10> composer require tymon/jwt-auth:2.1.1
./composer.json has been updated
Running composer update tymon/jwt-auth
```

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\PHL_2025\Minggu10\Jobsheet10> php artisan vendor:publish --provider="Tymon\JWTAuth\Providers\LaravelServiceProvider"
INFO Publishing assets.
Copying file [C:\laragon\www\PHL_2025\Minggu10\Jobsheet10\vendor\tymon\jwt-auth\config\config.php] to [C:\laragon\www\PHL_2025\Minggu10\Jobsheet10\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
```

```
PS C:\laragon\www\PHL_2025\Minggu10\Jobsheet10> php artisan jwt:secret
jwt-auth secret [UJCnJOHLWcvLq0ZUzDF87rpzg0Q5JjUDMBUmGbFYtS2bTmq78wSnb4LERF2m74wZ] set successfully.
```

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

```
JWT_SECRET=UJCnJOHLWcvLq0ZUzDF87rpzg0Q5JjUDMBUmGbFYtS2bTmq78wSnb4LERF2m74wZ
```

6. Selanjutnya lakukan konfigurasi guard API. Buka config/auth.php. Ubah bagian 'guards' menjadi seperti Berikut :

```

8         'guards' => [
9             'web' => [
10                 'driver' => 'session',
11                 'provider' => 'users',
12             ],
13             'api' => [
14                 'driver' => 'jwt',
15                 'provider' => 'users',
16             ],
17         ],

```

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

```

use Tymon\JWTAuth\Contracts\JWTSubject;

class UserModel extends Authenticatable implements JWTSubject
{
    public function getJWTIdentifier(){
        return $this->getKey();
    }
    public function getJWTCustomClaims(){
        return [];
    }
}

```

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

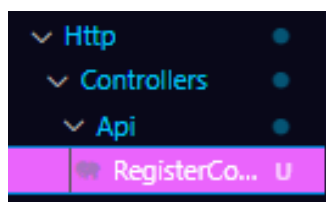
```
php artisan make:controller Api/RegisterController
```

```

PS C:\laragon\www\PWL_2025\Minggu10\Jobsheet10> php artisan make:controller Api/RegisterController
INFO Controller [C:\laragon\www\PWL_2025\Minggu10\Jobsheet10\app\Http\Controllers\Api\RegisterController.php] created successfully.

```

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



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

```

app > Http > Controllers > Api > RegisterController.php > ...
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use App\Models\UserModel;
6  use App\Http\Controllers\Controller;
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 validation 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),

```

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

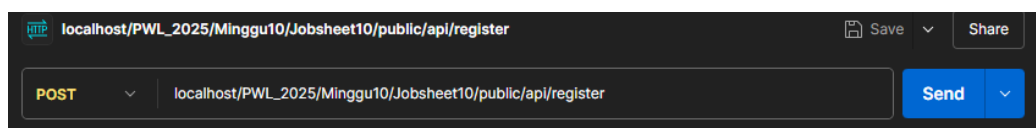
```

routes > api.php
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)->

```

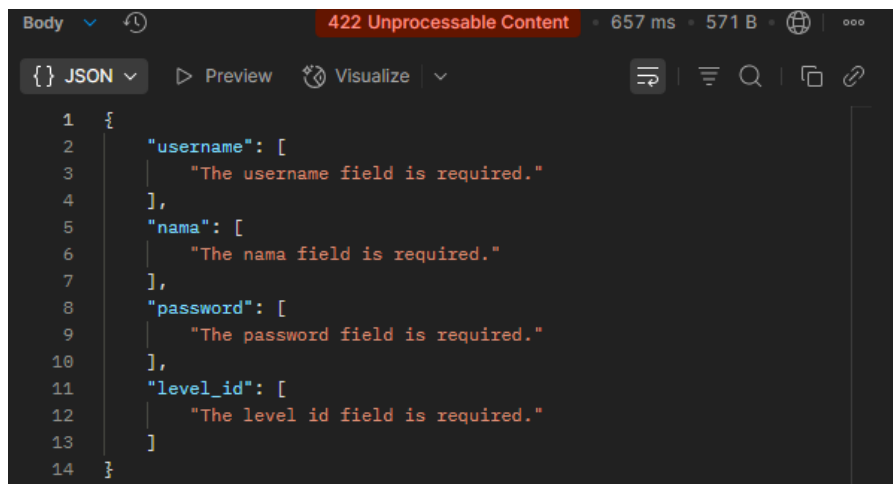
11. Jika sudah, kita akan melakukan uji coba REST API melalui aplikasi Postman.

Buka aplikasi Postman, isi URL localhost/PWL_POS/public/api/register serta method POST. Klik Send.



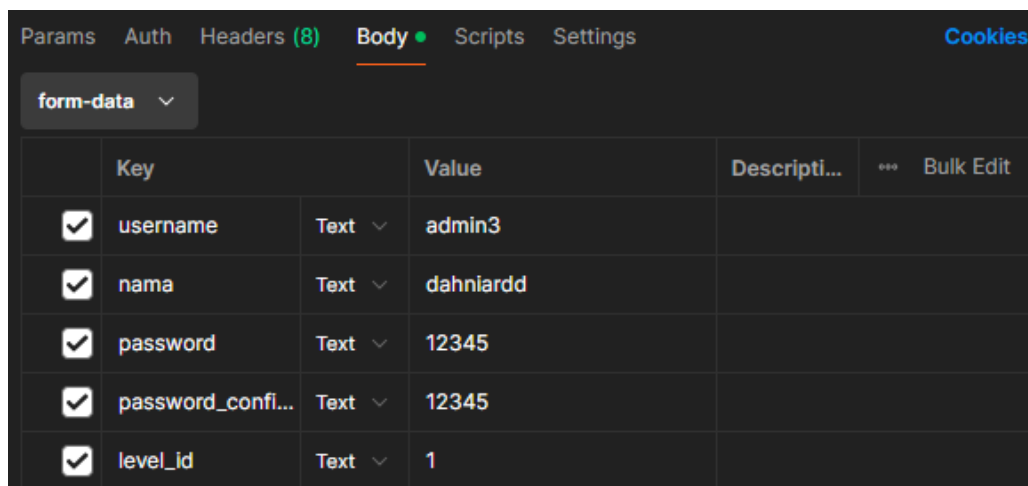
Jika berhasil akan muncul error validasi seperti gambar di atas.

Lakukan percobaan yang sama dan berikan screenshot hasil percobaan Anda.



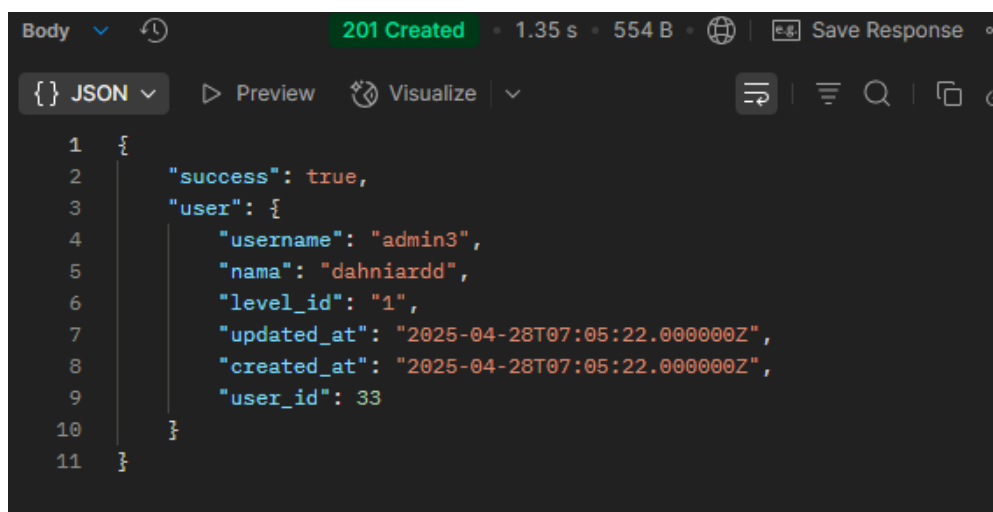
```
1 {
2   "username": [
3     "The username field is required."
4   ],
5   "nama": [
6     "The nama field is required."
7   ],
8   "password": [
9     "The password field is required."
10  ],
11  "level_id": [
12    "The level id field is required."
13  ]
14 }
```

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.



Params Auth Headers (8) Body Scripts Settings Cookies				
form-data				
	Key		Value	Descripti...
<input checked="" type="checkbox"/>	username	Text	admin3	
<input checked="" type="checkbox"/>	nama	Text	dahniardd	
<input checked="" type="checkbox"/>	password	Text	12345	
<input checked="" type="checkbox"/>	password_confir...	Text	12345	
<input checked="" type="checkbox"/>	level_id	Text	1	

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



```
1 {
2   "success": true,
3   "user": {
4     "username": "admin3",
5     "nama": "dahniardd",
6     "level_id": "1",
7     "updated_at": "2025-04-28T07:05:22.000000Z",
8     "created_at": "2025-04-28T07:05:22.000000Z",
9     "user_id": 33
10  }
11 }
```

13. Lakukan commit perubahan file pada Github.

B. PRAKTIKUM 2 : Membuat RESTful API Login

1. Kita buat file controller dengan nama LoginController.

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

```
PS C:\laragon\www\PWL_2025\Minggu10\Jobsheet10> php artisan make:controller Api/LoginController

INFO Controller [C:\laragon\www\PWL_2025\Minggu10\Jobsheet10\app\Http\Controllers\Api>LoginController.php] created successfully.
```

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

2. Buka file tersebut, dan ubah kode menjadi seperti Berikut :

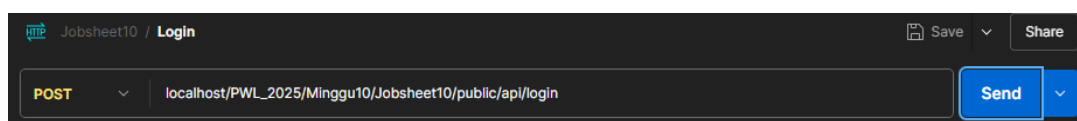
```
app > Http > Controllers > Api > LoginController.php > ...
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use App\Http\Controllers\Controller;
6  use Illuminate\Http\Request;
7  use Illuminate\Support\Facades\Validator;
8
9  class LoginController extends Controller
10 {
11     public function __invoke(Request $request)
12     {
13         // Set validation
14         $validator = Validator::make($request->all(), [
15             'username' => 'required',
16             'password' => 'required',
17         ]);
18
19         // If validation fails
20         if ($validator->fails()) {
21             return response()->json($validator->errors(), 422);
22         }
23     }
24 }
```

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

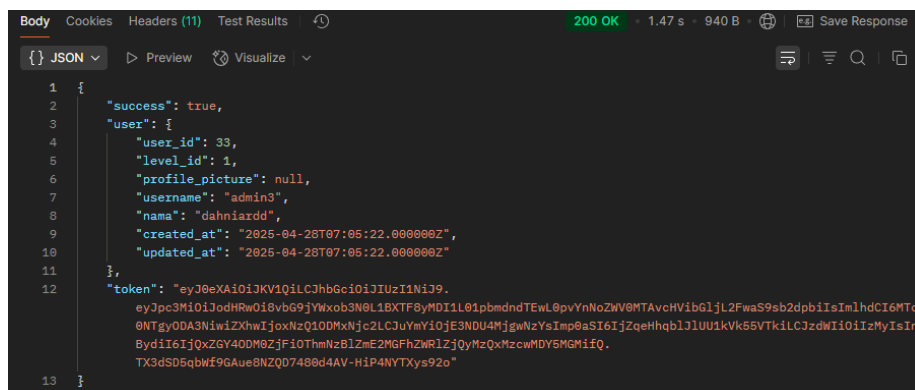
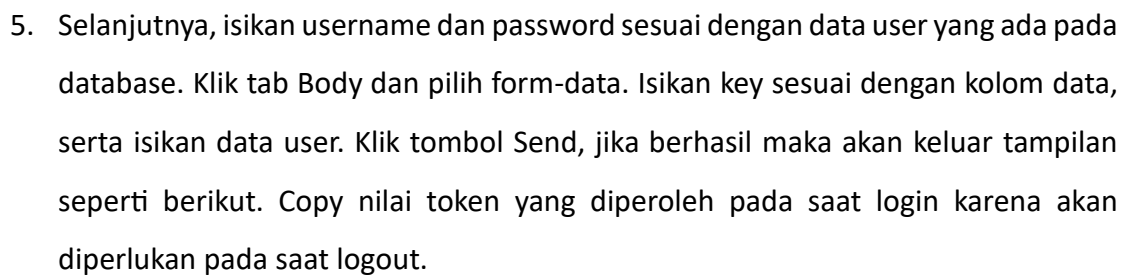
```
Route::post('/register', App\Http\Controllers\Api\RegisterController::class)->name('register');
Route::post('/login', App\Http\Controllers\Api>LoginController::class)->name('login');

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

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.

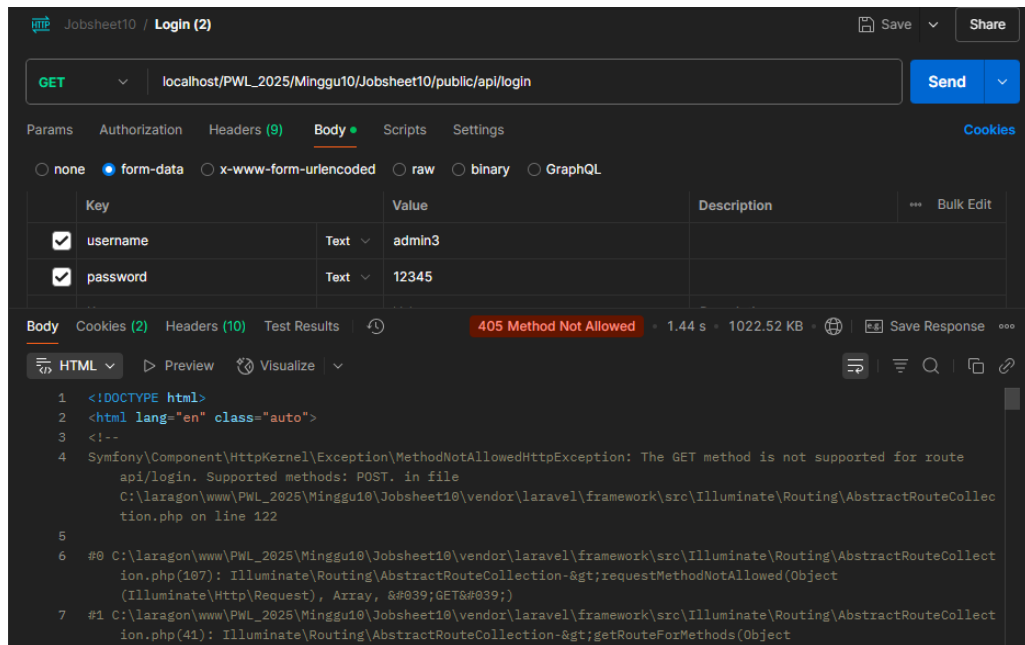


Jika berhasil akan muncul error validasi seperti gambar di atas.



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

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.



- **error 405 Method Not Allowed**, karena kita mencoba mengakses endpoint `/api/login` menggunakan method GET, padahal endpoint tersebut hanya menerima method POST.
- **GET:** Hanya untuk mengambil data (tidak boleh mengubah server). Contohnya ambil daftar produk, ambil profil user, dll. Sedangkan
- **POST:** Untuk mengirimkan data baru atau melakukan aksi ke server. Contohnya registrasi user, login user, buat postingan, dll.
- Sehingga login itu termasuk mengirim data username & password ke server, lalu server melakukan pengecekan, jadi WAJIB pakai POST, tidak bisa GET.

8. Lakukan commit perubahan file pada Github.

C. PRAKTIKUM 3 : Membuat RESTful API Logout

1. Tambahkan kode berikut pada file `.env`

```
JWT_SHOW_BLACKLIST_EXCEPTION=true
```

```
JWT_SHOW_BLACKLIST_EXCEPTION=true
```


2. Buat Controller baru dengan nama LogoutController.

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

```
PS C:\laragon\www\PWL_2025\Minggu10\Jobsheet10> php artisan make:controller Api/LogoutController

INFO Controller [C:\laragon\www\PWL_2025\Minggu10\Jobsheet10\app\Http\Controllers\Api\LogoutController.php] created successfully.
```

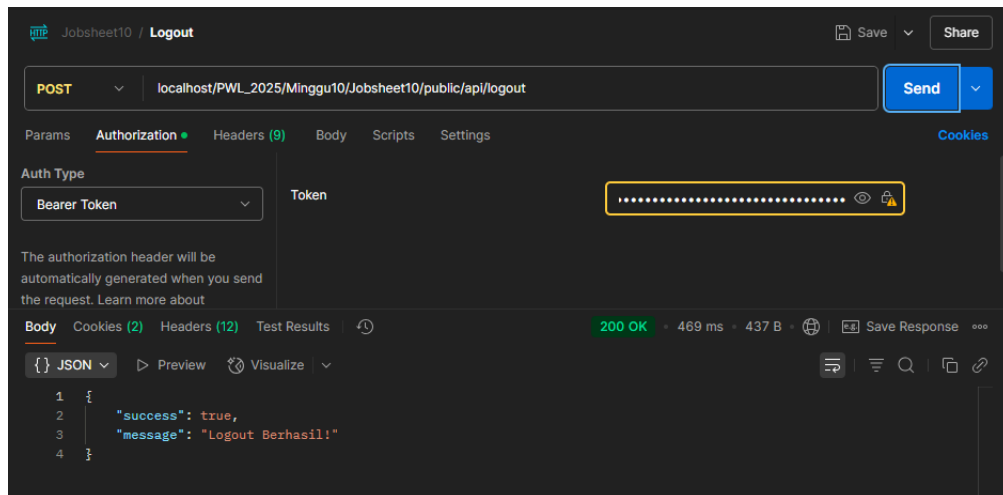
3. Buka file tersebut dan ubah kode menjadi seperti Berikut :

```
app > Http > Controllers > Api > LogoutController.php > ...
1  <?php
2
3  namespace App\Http\Controllers\Api;
4
5  use Illuminate\Http\Request;
6  use App\Http\Controllers\Controller;
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     {
16         // remove token
17         $removeToken = JWTAuth::invalidate(JWTAuth::getToken());
18
19         if ($removeToken) {
20             // return response JSON
21             return response()->json([
22                 'success' => true,
23                 'message' => 'Logout Berhasil!',
24             ]);
25         }
26     }
27 }
28
```

4. Lalu kita tambahkan routes pada api.php

```
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.



7. Lakukan commit perubahan file pada Github.

D. 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_2025\Minggu10\Jobsheet10> php artisan make:controller Api/LevelController

INFO: Controller [C:\laragon\www\PWL_2025\Minggu10\Jobsheet10\app\Http\Controllers\Api\LevelController.php] created successfully.
```

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

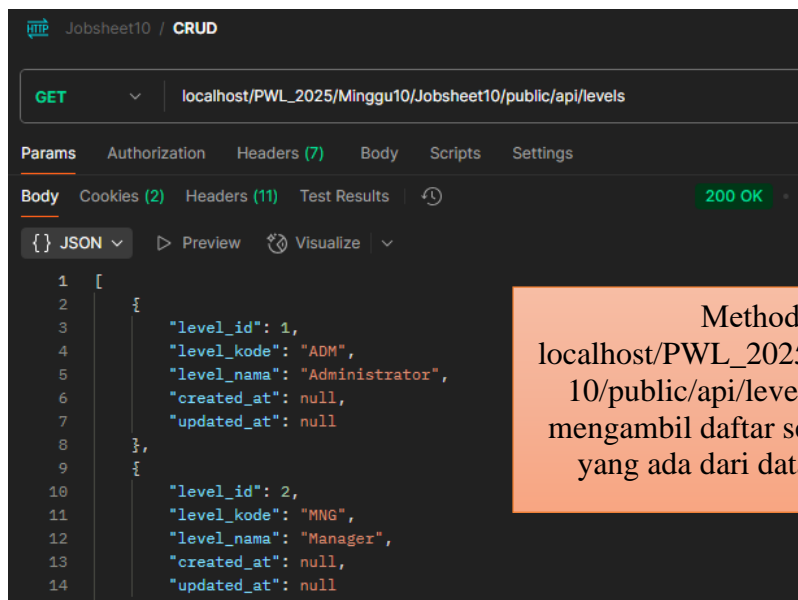
```
app > Http > Controllers > Api > LevelController.php > LevelController > update
1  <?php
2  namespace App\Http\Controllers\API;
3
4  use App\Http\Controllers\Controller;
5  use Illuminate\Http\Request;
6  use App\Models\LevelModel;
7
8  class LevelController extends Controller
9  {
10     public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

3. Kemudian kita lengkapi routes pada api.php

```
use App\Http\Controllers\API\LevelController;
```

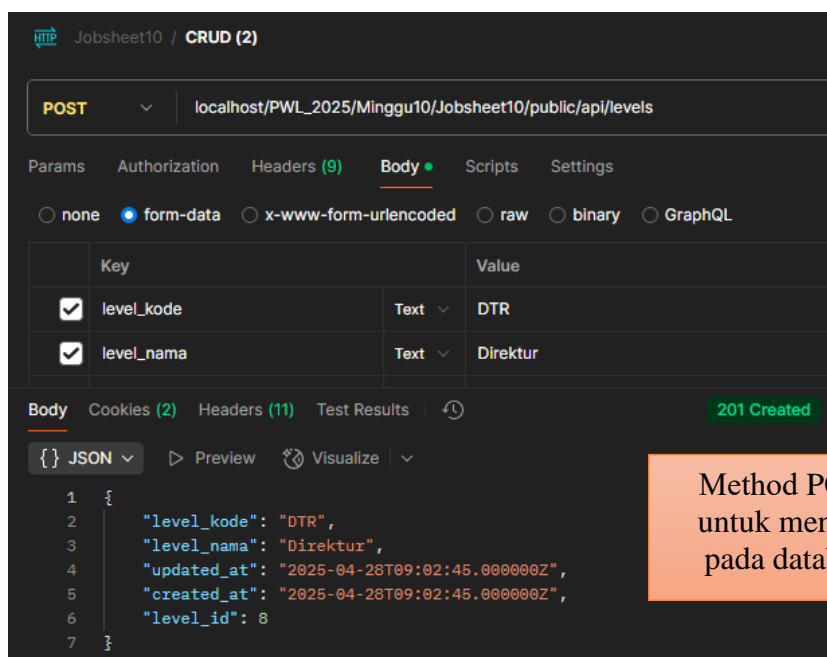
```
Route::get('levels', [LevelController::class, 'index']);
Route::post('levels', [LevelController::class, 'store']);
Route::get('levels/{level}', [LevelController::class, 'show']);
Route::put('levels/{level}', [LevelController::class, 'update']);
Route::delete('levels/{level}', [LevelController::class,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

4. Jika sudah. Lakukan uji coba API mulai dari fungsi untuk menampilkan data. Gunakan URL: localhost/PWL_POS-main/public/api/levels dan method GET.



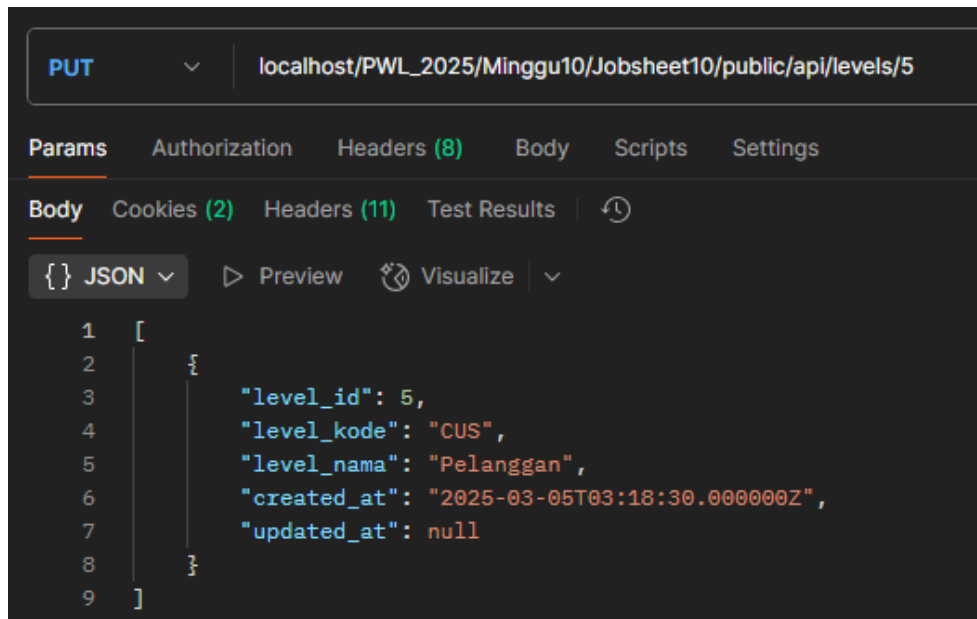
Method GET ke localhost/PWL_2025/Minggu10/Jobsheet10/public/api/levels digunakan untuk mengambil daftar semua level pada role yang ada dari database melalui API

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

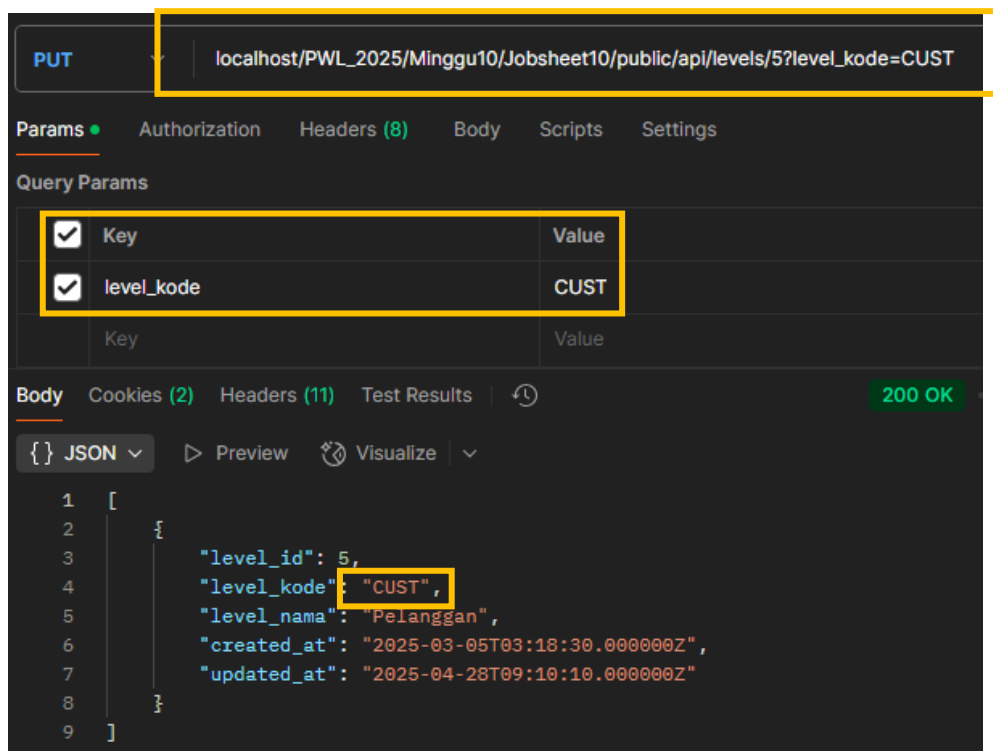


Method POST digunakan untuk menambahkan data pada database lewat API

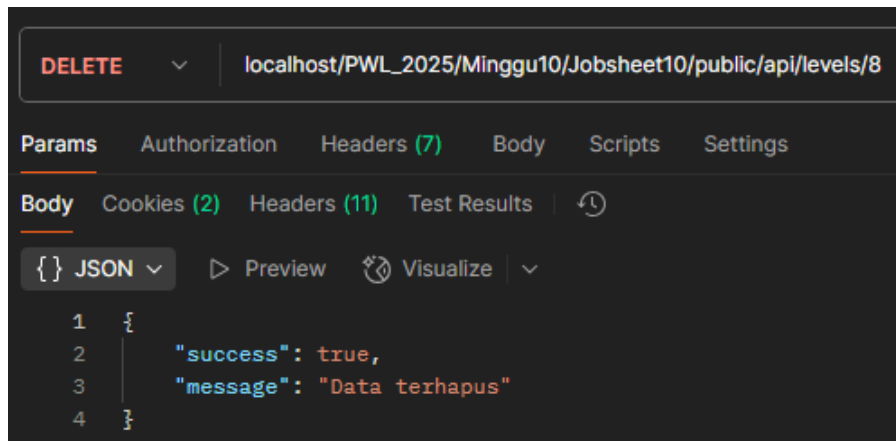
6. Berikutnya lakukan percobaan menampilkan detail data. Jelaskan dan berikan screenshot hasil percobaan Anda.



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.



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



9. Lakukan commit perubahan file pada Github.

E. TUGAS

- Implementasi CRUD API table lainnya yaitu table :

a) m_user

- Buat controller

```

PS C:\laragon\www\PWL_2025\Minggu10\Jobsheet10> php artisan make:controller Api/UserController

INFO Controller [C:\laragon\www\PWL_2025\Minggu10\app\Http\Controllers\Api\UserController.php] created successfully.

```

- Tambahkan isinya kedalam file controller

```

app > Http > Controllers > Api > UserController.php > UserController
1 <?php
2 namespace App\Http\Controllers\API;
3
4 use App\Http\Controllers\Controller;
5 use Illuminate\Http\Request;
6 use App\Models\UserModel;
7
8 class UserController extends Controller
9 {
10     public function index()
11     {
12         return UserModel::all();
13     }
14     public function store(Request $request)
15     {

```

- Lengkapi routesnya pada api.php

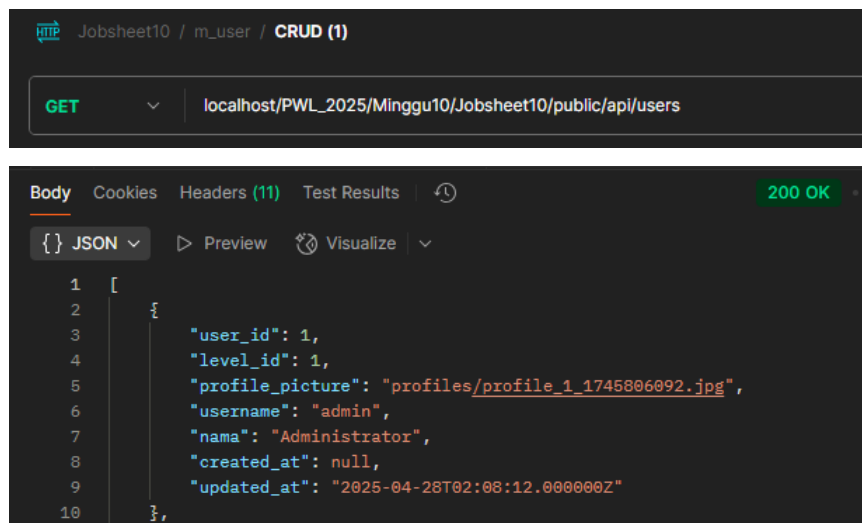
```
use App\Http\Controllers\API\UserController;
```

```

Route::get('users', [UserController::class, 'index']);
Route::post('users', [UserController::class, 'store']);
Route::get('users/{user}', [UserController::class, 'show']);
Route::put('users/{user}', [UserController::class, 'update']);
Route::delete('users/{user}', [UserController::class, 'destroy']);

```

4. Uji coba API method GET



HTTP Jobsheet10 / m_user / CRUD (1)

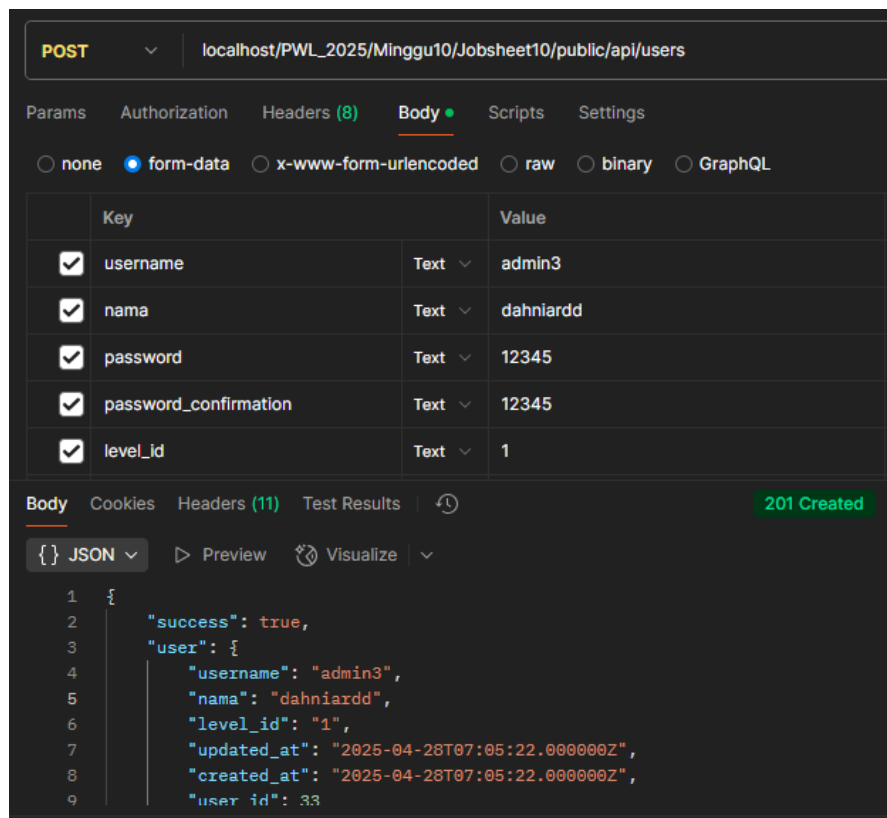
GET localhost/PWL_2025/Minggu10/Jobsheet10/public/api/users

Body Cookies Headers (11) Test Results 200 OK

{ } JSON Preview Visualize

```
1 [
2   {
3     "user_id": 1,
4     "level_id": 1,
5     "profile_picture": "profiles/profile_1_1745806092.jpg",
6     "username": "admin",
7     "nama": "Administrator",
8     "created_at": null,
9     "updated_at": "2025-04-28T02:08:12.000000Z"
10  },
```

5. Uji coba API method POST



POST localhost/PWL_2025/Minggu10/Jobsheet10/public/api/users

Params Authorization Headers (8) Body Scripts Settings

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

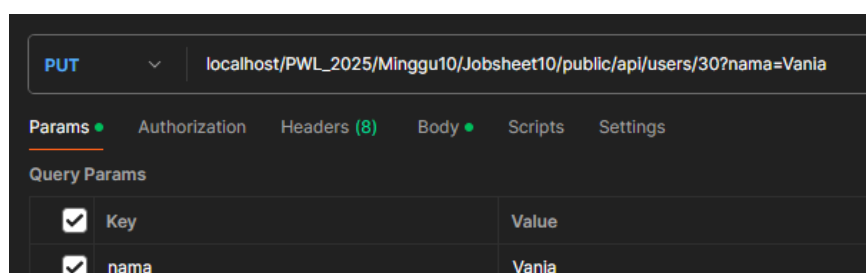
Key	Value
username	admin3
nama	dahniardd
password	12345
password_confirmation	12345
level_id	1

Body Cookies Headers (11) Test Results 201 Created

{ } JSON Preview Visualize

```
1 {
2   "success": true,
3   "user": {
4     "username": "admin3",
5     "nama": "dahniardd",
6     "level_id": "1",
7     "updated_at": "2025-04-28T07:05:22.000000Z",
8     "created_at": "2025-04-28T07:05:22.000000Z",
9     "user_id": 33
```

6. Uji coba API method PUT



PUT localhost/PWL_2025/Minggu10/Jobsheet10/public/api/users/30?nama=Vania

Params Authorization Headers (8) Body Scripts Settings

Query Params

Key	Value
nama	Vania

```
Body Cookies Headers (11) Test Results 200 OK
[{} JSON] [Preview] [Visualize]
10 },
11 {
12     "user_id": 30,
13     "level_id": 3,
14     "profile_picture": null,
15     "username": "staff2",
16     "nama": "Vania",
17     "created_at": "2025-04-15T21:22:45.000000Z",
18     "updated_at": "2025-04-29T02:09:23.000000Z"
19 }
20 ]
```

7. Uji coba API method DELETE

```
Jobsheet10 / m_user / CRUD (delete)
DELETE localhost/PWL_2025/Minggu10/Jobsheet10/public/api/users/31
Params Authorization Headers (6) Body Scripts Settings
Body Cookies Headers (11) Test Results 200 OK
[{} JSON] [Preview] [Visualize]
1 {
2     "success": true,
3     "message": "Data terhapus"
4 }
```

b) m_kategori

1. Buat controller

```
PS C:\laragon\www\PWL_2025\Minggu10\Jobsheet10> php artisan make:controller Api/KategoriController
INFO Controller [C:\laragon\www\PWL_2025\Minggu10\Jobsheet10\app\Http\Controllers\Api\KategoriController.php] created successfully.
```

2. Tambahkan isinya kedalam file controller

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

3. Lengkapi routesnya pada api.php

```
use App\Http\Controllers\API\KategoriController;
```

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

4. Uji coba API method GET

GET localhost/PWL_2025/Minggu10/Jobsheet10/public/api/kategoris

Params Authorization Headers (6) Body Scripts Settings

Body Cookies Headers (11) Test Results 200 OK

{ } JSON Preview Visualize

```
1 [
2   {
3     "kategori_id": 1,
4     "kategori_kode": "KTG001",
5     "kategori_nama": "Elektronik",
6     "created_at": null,
7     "updated_at": null
8   },
9 ]
```

5. Uji coba API method POST

POST localhost/PWL_2025/Minggu10/Jobsheet10/public/api/kategoris

Params Authorization Headers (8) Body Scripts Settings

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

Key	Value
kategori_id	14
kategori_kode	KTG0014
kategori_nama	Barang pecah Belah

Body Cookies Headers (11) Test Results 201 Created

{ } JSON Preview Visualize

```
1 {
2   "kategori_kode": "KTG0014",
3   "kategori_nama": "Barang pecah Belah",
4   "updated_at": "2025-04-29T02:25:22.000000Z",
5   "created_at": "2025-04-29T02:25:22.000000Z",
6   "kategori_id": 14
7 }
```

6. Uji coba API method PUT

PUT localhost/PWL_2025/Minggu10/Jobsheet10/public/api/kategoris/6?kategori_nama=Snack Ringan

Params Authorization Headers (7) Body Scripts Settings

Query Params

Key	Value	Description
kategori_nama	Snack Ringan	


```
Body Cookies Headers (11) Test Results 200 OK • 595 ms • 5
[{} JSON] Preview Visualize
1 [
2   {
3     "kategori_id": 6,
4     "kategori_kode": "KTG006",
5     "kategori_nama": "Snack Ringan",
6     "created_at": "2025-03-17T02:02:14.000000Z",
7     "updated_at": "2025-04-29T02:29:08.000000Z"
8   }
]
```

7. Uji coba API method DELETE

```
DELETE localhost/PWL_2025/Minggu10/Jobsheet10/public/api/kategoris/14
Params Authorization Headers (6) Body Scripts Settings
Body Cookies Headers (11) Test Results 200 OK
[{} JSON] Preview Visualize
1 {
2   "success": true,
3   "message": "Data terhapus"
4 }
```

c) m_barang

1. Buat controller

```
PS C:\laragon\www\PWL_2025\Minggu10\Jobsheet10> php artisan make:controller Api/BarangController
INFO Controller [C:\laragon\www\PWL_2025\Minggu10\Jobsheet10\app\Http\Controllers\Api\BarangController.php] created successfully.
```

2. Tambahkan isinya kedalam file controller

```
app > Http > Controllers > Api > BarangController.php > BarangController > update
1 <?php
2 namespace App\Http\Controllers\API;
3
4 use App\Http\Controllers\Controller;
5 use Illuminate\Http\Request;
6 use App\Models\BarangModel;
7
8 class BarangController extends Controller
9 {
10     public function index()
11     {
12         return BarangModel::all();
13     }
14     public function store(Request $request)
15     {
16         $barang = BarangModel::create($request->all());
17         return response()->json($barang, 201);
18     }
19     public function show(BarangModel $barang)
20     {
21         return BarangModel::find($barang);
22     }
23     public function update(Request $request, BarangModel $barang)
```

3. Lengkapi routesnya pada api.php

```
use App\Http\Controllers\API\BarController;
```

```
Route::get('barangs', [BarController::class, 'index']);
Route::post('barangs', [BarController::class, 'store']);
Route::get('barangs/{barang}', [BarController::class, 'show']);
Route::put('barangs/{barang}', [BarController::class, 'update']);
Route::delete('barangs/{barang}', [BarController::class, 'destroy']);
```

4. Uji coba API method GET

GET localhost/PWL_2025/Minggu10/Jobsheet10/public/api/barangs

Params Authorization Headers (6) Body Scripts Settings

Body Cookies Headers (11) Test Results 200 OK

{ } JSON Preview Visualize

```
1 [
2   {
3     "barang_id": 1,
4     "kategori_id": 1,
5     "barang_kode": "BRG001",
6     "barang_nama": "Laptop ASUS",
7     "harga_beli": 16000000,
8     "harga_jual": 17500000,
9     "created_at": null,
10    "updated_at": null
11  },
12 ]
```

5. Uji coba API method POST

POST localhost/PWL_2025/Minggu10/Jobsheet10/public/api/barangs

Params Authorization Headers (8) Body Scripts Settings

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

	Key	Value
<input checked="" type="checkbox"/>	barang_id	26
<input checked="" type="checkbox"/>	kategori_id	9
<input checked="" type="checkbox"/>	barang_kode	BRG021
<input checked="" type="checkbox"/>	barang_nama	Purbasari Body Scrub
<input checked="" type="checkbox"/>	harga_beli	15000
<input checked="" type="checkbox"/>	harga_jual	20000

Body Cookies Headers (11) Test Results 201 Created

{ } JSON Preview Visualize

```
1 {
2   "kategori_id": "9",
3   "barang_kode": "BRG021",
4   "barang_nama": "Purbasari Body Scrub",
5   "harga_beli": "15000",
6   "harga_jual": "20000",
7   "updated_at": "2025-04-29T02:47:59.000000Z",
8 }
```

6. Uji coba API method PUT

PUT

localhost/PWL_2025/Minggu10/Jobsheet10/public/api/barangs/9?barang_nama=Indomie Goreng Rendang

Params

Authorization

Headers (7)

Body

Scripts

Settings

Query Params

<input checked="" type="checkbox"/> Key	Value	Description
<input checked="" type="checkbox"/> barang_nama	Indomie Goreng Rendang	

Body

Cookies

Headers (11)

Test Results

200 OK

{ } JSON

Preview

Visualize

```
12 {
13   "barang_id": 9,
14   "kategori_id": 3,
15   "barang_kode": "BRG009",
16   "barang_nama": "Indomie Goreng Rendang",
17   "harga_beli": 3500,
18   "harga_jual": 5000,
19   "created_at": null,
20   "updated_at": "2025-04-29T02:52:32.000000Z"
21 }
22 ]
```

7. Uji coba API method DELETE

DELETE

localhost/PWL_2025/Minggu10/Jobsheet10/public/api/barangs/23

Params

Authorization

Headers (6)

Body

Scripts

Settings

Body

Cookies

Headers (11)

Test Results

200 OK

{ } JSON

Preview

Visualize

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

lete	22	9 BRG017	BB Cream Wardah	65000
lete	24	9 BRG019	G2G Moisturizer	45000